University of Southampton
Faculty of Engineering
Department of Electronics and Computer Science
Southampton S017 1BJ

Fountain Codes for the Wireless Internet

by
Thanh Nguyen Dang
MEng.

A doctoral thesis submitted in partial fulfilment of the requirements for the award of Doctor of Philosophy at the University of Southampton
September 2008

SUPERVISORS:

First supervisor: Prof. Lajos Hanzo
Dipl Ing, MSc, PhD, FIEEE
DSc, FIEE, FREng
Chair of Telecommunications

Second supervisor: Dr Lie Liang Yang
BEng, MEng, PhD, SMIEEE, MIET
Reader in
Wireless Communication

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND APPLIED SCIENCE
DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

Fountain codes for the Wireless Internet

by Thanh Nguyen Dang

In this thesis, novel Foutain codes are proposed for transmission over wireless channels. The thesis concentrates on a specific version of Fountain codes, namely on Luby transform codes. More specifically, we consider their concatenation with classic error correction codes, yielding schemes, such as the concatenated Luby Transform and Bit Interleaved Coded Modulation using Iterative Decoding (LT-BICM-ID), the amalgamated Luby Transform and Generalized Low Density Parity Check (LT-GLDPC) code, or the Luby Transform coded Spatial Division Multiple Access (LT-SDMA) scenario considered. The thesis also investigates the potential of Systematic Luby Transform (SLT) codes using soft-bit decoding and analyses their Bit Error Ratio (BER) performance using EXtrinsic Information Transfer (EXIT) charts. SLT codes using different degree distributions and random integer packet index generation algorithms for creating the parity and information part of the SLT codeword are also investigated in this thesis. For the sake of improving both the BER performance and the diversity gain of Vertical Bell Laboratories Layered Space Time (V-BLAST) schemes, in this thesis a SLT coded V-BLAST system having four transmit and four receive antennas is proposed. Finally, A Hybrid Automatic Repeat reQuest (H-ARQ) SLT coded modulation scheme is designed in this thesis, where SLT codes are used both for correcting erroneous bits and for detecting as well as retransmitting erroneous Internet Protocol (IP) based packets. Erroneous IP packet detection is implemented using syndrome checking with the aid of the SLT codes' Parity Check Matrix (PCM). Optimizing the mapping of SLT-encoded bits to modulated symbols and then using iterative decoding for exchanging extrinsic information between the SLT decoder and the demapper substantially improves the achievable Bit Error Ratio (BER) performance of the scheme.

Acknowledgements

I would like to express my heartfelt gratitude to Professor Lajos Hanzo for his outstanding supervision and support throughout my research. His guidance, inspiration and encouragement have greatly benefited me not only in work but also in life. He has also managed to cultivate in me the desire to be a good researcher through his enthusiasm and perseverance in research. Most importantly, I would like to thank him for his invaluable friendship.

Many thanks also to Dr Lie Liang Yang, Dr Soon Xin Ng, my colleagues and the staff of the Communications Group for their support, help and discussions throughout my research. Special thanks are also due to Denise Harvey for her help in the administrative matters. The financial support of the EPSRC, UK and of the European Union is gratefully acknowledged. I would also like to express my appreciation to my parents, as well as to my parents in law for their love and support. Finally, to my wife, Do Thi Vuong, for her love, sacrifice for me and my daughters.

List of Publications

1. **R. Tee, T. D. Nguyen, L-L. Yang and L. Hanzo**, "Serially Concatenated Luby Transform Coding and Bit-Interleaved Coded Modulation Using Iterative Decoding for The Wireless Internet", In Proceedings of VTC 2006 Spring, Melbourne, CD ROM, May 2006, vol.138, pp. 177-182.

2. **T. D. Nguyen, F. C. Kuo, L-L. Yang and L. Hanzo**, "Amalgamated Generalized Low Density Parity Check and Luby Transform Codes for The Wireless Internet", On IEEE 65th VTC2007-Spring Vehicular Technology Conference, Dublin, April 2007, pp. 2440-2444.

3. **R. Tee, T. D. Nguyen, L-L. Yang and L. Hanzo**, "Luby Transform Coding Aided Bit-Interleaved Coded Modulation for the Wireless Internet", In Proceeding of VTC Fall 2007, Baltimore, September 2007, pp. 2025-2029.

4. **T. D. Nguyen, L-L Yang, L. Hanzo**, "Systematic Luby Transform Codes and Their Soft Decoding", In Proceedings of IEEE Workshop on Signal Processing Systems in Shanghai, China, CD ROM, October 2007, pp. 67-72.

5. **C-Y. Wei, T. D. Nguyen, N. Wu, J. Akhtman, L-L. Yang and L. Hanzo**, "Luby Transform Coding Aided Iterative Detection for Downlink SDMA Systems", In Proceedings of IEEE Workshop on Signal Processing Systems in Shanghai, China, CD ROM, October 2007, pp. 105-110.

6. **N. Wu, T. D. Nguyen, Chun-Yi Wei, L. L. Yang and L. Hanzo**, "Integrated Luby Transform Coding, Bit Interleaved Differential Space Time Coding and Sphere Packing Modulation for the Wireless Internet", In Proceeding of IEEE 67th VTC2008-Spring Vehicular Technology Conference, Singapore, CD ROM, May 2008, pp. 344-348.

7. **T. D. Nguyen, L. L. Yang, S. X. Ng and L. Hanzo**, "An Optimal Degree Distribution Design and A Conditional Random Integer Generator for The Systematic Luby Transform Coded Wireless Internet", In Proceeding of IEEE WCNC 2008 Conference, Las Vegas, Nevada, USA, CD ROM, 31 March - 3 April, pp. 243-248.

8. **T. D. Nguyen, M. El-Hajjar, L. L. Yang and L. Hanzo**, "A Systematic Luby Transform Coded V-BLAST System", In Proceeding of The 2008 IEEE International

Conference on Communications, Beijing, China, CD ROM, 19-23 May, 2008, pp. 775-779.

9. **T. D. Nguyen, L. L. Yang, S. X. Ng and L. Hanzo**, "Systematic Luby Transform Codes and Their Degree Distribution Designed for Transmission over Rayleigh Fading Contaminated Binary Erasure Channels", Submitted to the IEEE Transactions on Vehicular Technology, 2008.

10. **T. D. Nguyen, R. Tee, L. L. Yang and [1]L. Hanzo**, "Hybrid ARQ Aided Systematic Luby Transform Coded Modulation" Submitted to the IEEE Transactions on Vehicular Technology, 2008.

DECLARATION OF AUTHORSHIP

I, **Thanh Nguyen Dang**,

declare that the thesis entitled

Fountain Codes for the Wireless Internet

and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- parts of this work have been published as:*(see list of publications)*

Signed:.....

Date:.....

Contents

Chapter 1

Introduction

1.1 Introduction

A typical digital communication system providing diverse services, such as Digital Terrestial teleVision Broadcasting (DVB-T), Digital Television Broadcasting to Handhelds (DVB-H), Digital Satellite Television Broadcasting (DVB-S), mobile voice and data services, Internet Protocol based TeleVision (IPTV) broadcast etc is portrayed in Fig. 1.1. There is a proliferation of services, but the capacity of communication channels remains limit. Hence, the efficient exploitation of the available capacity is the important requirement imposed on communication systems. For the sake of correcting the received error-infested data digital communications systems invariably employ diverse Forward Error Correction (FEC) codes, such as convolution codes [2], Reed-Solomon (RS) codes [3], Bose-Chaudhuri-Hocquenghem (BCH) codes [4] [5], turbo codes [6], Low Density Parity Check (LDPC) codes [7] and etc have been developed for communicating over a wide variety of channels. When designing channel codes, there are many contradictory design factors to be considered, some of which are portrayed in Fig. 1.2. Changing any of these factors will result in a different performance. For example, reducing the code rate may reduce the Bit Error Ratio (BER), but this will also decrease the effective throughput of the system. On the other hand, increasing the length of the codeword for the sake of attaining an improved BER performance will increase the delay of both the encoding and decoding process, as well as the complexity imposed.

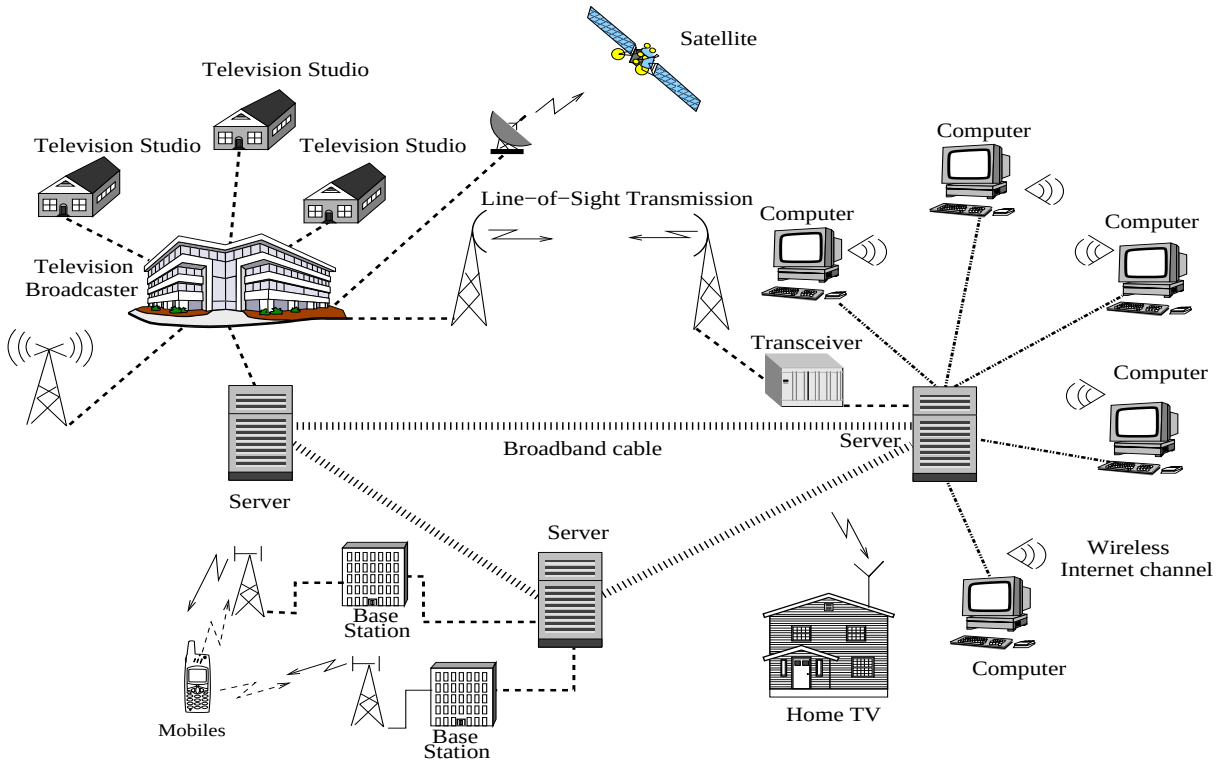


Figure 1.1: Communications system model.

There are numerous types of communication channels and an importent one closely related to the initial motivation of this thesis is the Binary Erasure Channel (BEC), which is characterized in Fig. 1.3. The BEC models Internet based networks, which may randomly

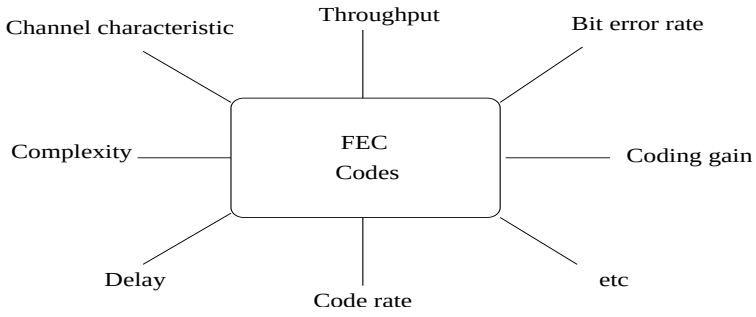


Figure 1.2: Factors affecting the design of FEC codes.

drop Internet Protocol (IP) packets owning to statistical multiplexing in the routers. For the sake of overcoming the dropping of IP packets, the classic technique used for communication over Internet channels is constituted by the Automatic Repeat-reQuest (ARQ) protocol combined with FEC codes [8] [9]. This method relies on the ACKnowledge-

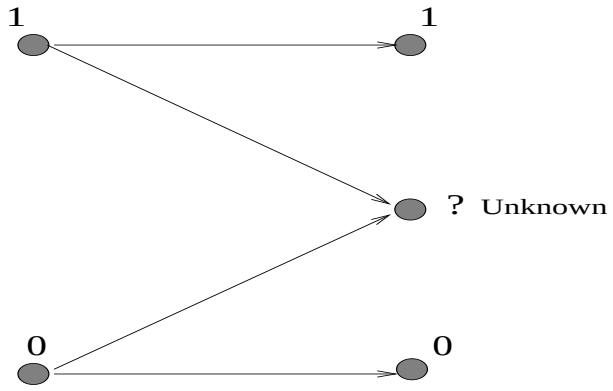


Figure 1.3: The binary erasure channel.

ment (ACK) signal fed back from the receiver to the transmitter. The receiver will send an ACK signal to the transmitter, if it successfully receives a packet and a Negative ACKnowledgement (NACK) signal, if the packet does not arrive within a specific tolerable delay limit. When transmitter 'A' receives a NACK signal instead of the ACK message, it will re-transmits this IP packet. This process will continue until the transmitter receives an ACK signal from the remote receiver or the tolerable delay expires. Recently, the Fountain codes were proposed [10] [11] for the sake of recovering the lost packets, when communicating over Internet channels, which constitutes the subject of this thesis.

1.2 Historical Overview of Forward Error Correction Codes

The history of FEC codes evolved from Shannon's pioneering work [12] published in 1948, in which he showed that it is possible to design a communication system with any desired small probability of error, whenever the rate of transmission is lower than the capacity of the channel. In the ensuing period numerous FEC codes were invented such as convolutional codes proposed by Elias [2] in 1955 or Reed-Solomon (RS) codes contrived in 1960 by Irving S. Reed and Gustave Solomon, who were then members of MIT Lincoln Laboratory. Later the efficient decoding of RS codes was invented by Elwyn Berlekamp, a professor of electrical engineering at the University of California, Berkeley. In 1962 the Low Density Parity Check (LDPC) codes were proposed by Gallager [7], but they remained dormant until the 1990s. In 1967, Viterbi [13] invented the maximum likelihood sequence estimation algorithm for efficiently decoding convolutional codes. The Maximum A-Posteriori (MAP) algorithm was presented by Bahl *et al.* [14] in 1974, which is capable

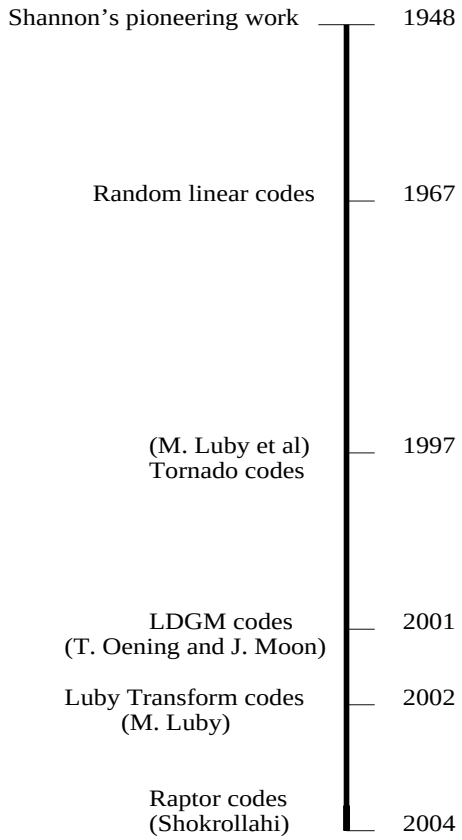


Figure 1.4: The history of Fountain Codes.

of attaining the minimum achievable BER. Berrou, Glavieux and Thitimajshima [15] proposed the witty idea of turbo codes in 1993. This invention facilitated the development of FEC codes capable of approaching the Shannon limit and has attracted intensive research efforts [16] [17]. Turbo codes were also invoked in the standardised Third-Generation (3G) mobile radio systems [18]. In 1997, Luby *et al.* introduced Tornado codes [19], which use low-complexity encoding and decoding processes for filling packet erasures with the aid of redundant packets. In 2002 Luby *et al.* proposed a novel class of Fountain codes, namely Luby Transform (LT) codes [10]. This code creates the LT-encoded packets by simply using the Exclusive OR (XOR) function of the source information packets to create parity packets based on a generator matrix. In 2004, Shokrollahi [11] invented a new class of Fountain codes referred to as Raptor codes, which were designed based on combining LT codes with LDPC codes. In this thesis we will investigate the family of LT codes and their combination with other sophisticated FEC codes. The history of Fountain codes is briefly portrayed in Fig. 1.4.

1.3 Organisation and Novel Contributions of the Thesis

The thesis is organised in three main chapters and as detailed below:

- **Chapter 2:** This chapter is divided into two main sections. Section 2.1 briefly considers Tornado code, while Section 2.2 is divided into five subsections. In Section 2.2.1, we outline the basic concept of Fountain codes. We introduce Luby transform codes in Section 2.2.2 and explore Raptor codes in Section 2.2.3 by analysing their Tanner graphs. Section 2.2.4 is dedicated to systematic Raptor codes. Finally, Section 2.2.5 details the degree distributions of LT codes and Raptor codes.

- **Chapter 3:** This chapter explores the design of Luby transform coded communication systems using hard decoding. Section 3.1 outlines the novel Improved Robust Soliton Degree Distribution (IRSDD) and investigates a sophisticated Serially Concatenated LT Coding and Bit-Interleaved Coded Modulation scheme using Iterative Decoding (BICM-ID) designed for transmission over the Wireless Internet. Our investigations are conducted for transmission over uncorrelated Rayleigh fading channels and using 16-level Quadrature Amplitude Modulation (16QAM). In Section 3.2 amalgamated Generalized Low Density Parity Check (G-LDPC) and LT Codes are designed for the Wireless Internet. The G-LDPC code words are mapped to LT-encoded packets and a Log-Likelihood Ratio (LLR) based packet reliability metric is defined. This allows the LT decoding process to erase the gravely contaminated received LT packets in order to avoid the avalanche-like propagation of errors, when attempting to recover other source packets from these contaminated packets. The achievable performance is characterized in Section 3.2.2. In Section 3.3 a novel random integer index generator is proposed for the degree distribution of the LT source packets. Furthermore, bit-by-bit hard LT decoding aided BICM-ID is investigated.

- **Chapter 4:** This chapter presents a soft bit decoding algorithm designed for LT codes. Section 4.1 introduces the Systematic Luby Transform (SLT) encoding process using a soft-bit decoding algorithm, while Section 4.2 characterizes their performance. Section 4.3 presents a novel Truncated Degree Distribution (TDD) for determining the specific degree of SLT parity packets, complemented by a novel Conditional Random Integer Generator (CRIG) used for designing the degree of the SLT source packets. The analysis of SLT codes by EXtrinsic Information Transfer (EXIT) chart is also presented in this section, while the coresponding BER sim-

ulation results are provided in Section 4.3.5. Section 4.4 characterizes the attainable BER performance of SLT codes using different degree distributions for the SLT-parity packets as well as the random integer generating algorithms of Section 4.3 used for the SLT-source packets. A novel SLT coded Vertical Bell Labs Layered Space-Time (V-BLAST) system is proposed in Section 4.5.

- **Chapter 5:** Sophisticated Hybrid Automatic Repeat reQuest (H-ARQ) aided SLT coded modulation schemes are proposed in Section. 5.1. The phylosophy of ARQ protocols is introduced in Section 5.1.1, while in Section 5.1.2 the proposed system architecture is detailed. In this section different H-ARQ aided SLT coded modulation schemes using the Gray- as well as Set-partitioning based mapping are proposed, while the corresponding EXIT chart analysis is carried out in Section 5.1.3. The achievable system performance is characterized in Section 5.1.4.

- **Chapter 6**: This chapter concludes the thesis and provides ideas for future research.

The novel contributions of the thesis are as follows:

- The improved robust soliton degree distribution was proposed for LT codes and it was amalgamated with FEC scheme [20], [21].

- An LLR-based LT packet reliability estimation technique was proposed for avoiding error propagation and hence for improving the achievable performance [22], [23], [24].

- The novel family of SLT codes and their soft bit decoding was contrived [25].

- The novel truncated degree distribution and a conditional random integer generator were designed for SLT codes [26], [27].

- An SLT coded V-BLAST scheme exchanging extrinsic information between the QPSK demapper and the SLT decoder was designed [28].

- A novel H-ARQ aided SLT scheme assisted by syndrome checking based packet reliability estimation was designed [29].

Having presented an overview of the thesis, let us now commence our detailed discourse on Fountain codes in the following chapters.

Chapter 2

Fountain Code Theory

2.1 Traditional Erasure Codes

There are many kinds of FEC codes which are used to protect data transmitted over the BEC channel. These codes include RS codes [30], LDPC codes [7] [31] [32], Low Density Generator Matrix (LDGM) codes [33] [34] and Tornado codes [19] [1] [35]. Below we analyse the structure of Tornado codes, which constitute a member of the family of erasure correcting codes.

2.1.1 Tornado Codes [1]

Tornado codes are erasure-filling block codes based on irregular sparse graphs [1] and designed for transmission over the Internet. These codes can be designed using an arbitrary alphabet size and are generated by cascading a sequence of irregular random bipartite graphs. The operation of such a graph is shown in Fig. 2.1, where the nodes at the left represent the original information packets, while those at the right are computed by performing an XOR operation of the appropriately selected input packets. Let us define a code $C(\xi)$ having k input packets and βk redundant packets, where $0 < \beta \leq 1$. The code words of a code $C(\xi)$ are generated by associating the input packets and output packets with each other using a bipartite graph ξ, as seen in Fig. 2.1. The resultant redundant packets are also referred to as parity packets. Again, the graph ξ has k nodes at the left and βk nodes at the right, corresponding to the original information input packets and the parity packets, respectively. The encoding process of $C(\xi)$ generates the parity

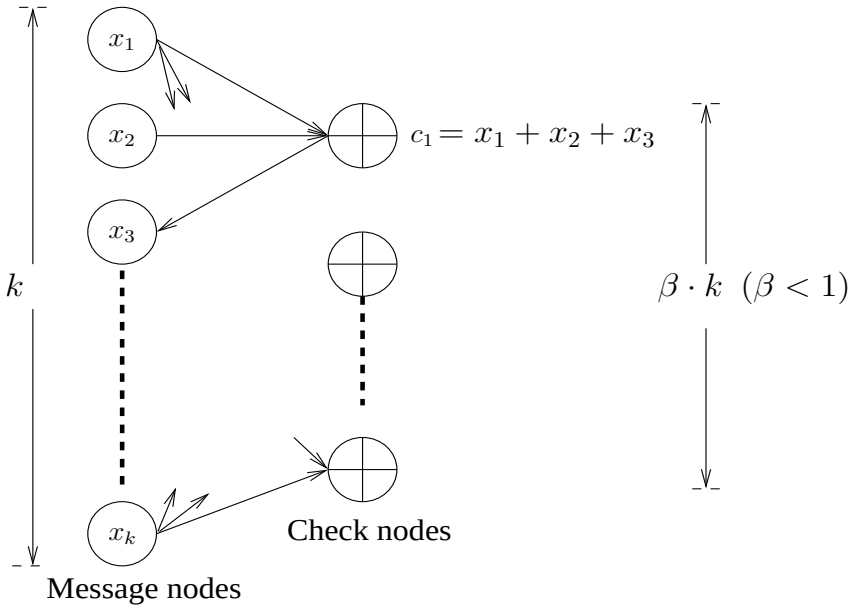


Figure 2.1: The bipartite-graph.

packets by modulo two combining according to the connections seen in Fig. 2.1. Therefore, the encoding complexity is proportional to the total number of modulo-2 connections in Fig. 2.1, which are also referred to as edges. The encoding process is implemented as follows:

- Firstly, partitioning the entire file to be transmitted into a number of k-packet source sequences and copying them to the encoded packet sequence, where again, each packet may contain a single or indeed an arbitrary number of source bits.
- Generating the first parity block B_0 of βk number of parity packets from the k-packet source information, as seen in Fig. 2.2.
- Generating the second parity block B_1 of $\beta^2 k$ number of parity packets from the βk parity packets of the first parity block B_0, as seen in Fig. 2.2. The number of parity packets of a new block equals the product of β and the number of parity packets existing in the immediately preceeding adjacent block.
- Similarly, continue generating parity blocks, until the parity block B_m of the cascade graph seen in Fig. 2.2 was created.
- At the last encoding level, the $\beta^{m+1} k$ number of parity packets of the parity block B_m are encoded once again by a conventional erasure filling code C_0 having a code rate

of $(1-\beta)$, for which the random loss of a fraction β of its packets can be recovered with a high probability [19]. Alternatively, we can continue with the creation of the cascade graph, until we generated about $\sqrt[4]{k}$ check nodes and then use erasure-filling decoding for the resultant code [19].

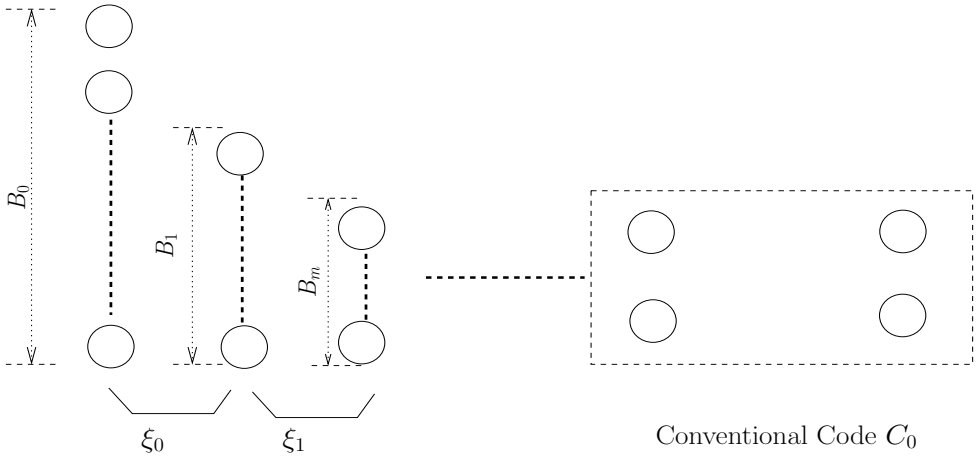


Figure 2.2: The cascaded graph sequence used to generate check packets of Tornado codes.

We have to construct the Tornado code $[C(\xi_0), \cdots, C(\xi_m)]$ from the constituent codes represented by the cascade graph $(\xi_0, \cdots, \xi_m)$, where the component part ξ_i has $\beta^i k$ left nodes and $\beta^{i+1}k$ right nodes created from the left nodes. The value m is chosen to ensure that $(\beta^{m+1}k)$ approximately equals $\sqrt{k}$. The Tornado code $C(\xi_0, \xi_1, \cdots, \xi_m, C_0)$ has a total number of parity packets calculated as follows:

$$\sum_{i=1}^{m+1} \beta^i \cdot k + \beta^{m+2} \cdot k/(1-\beta) = k \cdot \beta/(1-\beta). \tag{2.1}$$

From (2.1) we can infer the general code rate of the Tornado code that equals $[\beta/(1-\beta)]$. The Tornado code can recover a fraction of $[(1-\varepsilon)\cdot\beta\cdot k]$ of the randomly erased received packets with a high probability, provided that all of them are lost from $[C(\xi_0), \cdots, C(\xi_m)]$ and not from $[C(\xi_0), \cdots, C(\xi_m), C_0]$, where ε denotes the number of erased packets [19].

Tornado decoding process

The erased packets are then recovered from the received packets that duly arrived, which are constituted by both parity packets and information packets using the decoding process referred to as "erasure-filling decoding" [19]. This decoding process employs XOR operations to recover the erased packets, as seen in Fig. 2.3. We can analyse the decoding process by considering a subgraph seen in Fig. 2.4b of the original graph seen in

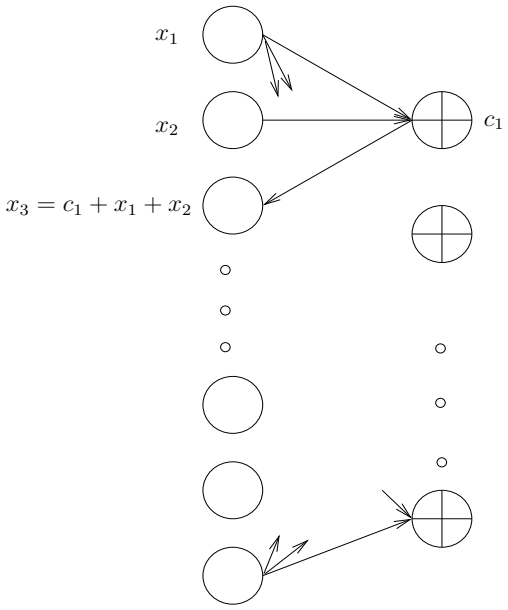


Figure 2.3: An example of the Tornado decoding process.

Fig. 2.4a. As mentioned above, this subgraph consists of left nodes, right nodes and the edges connecting these nodes, as seen in Fig. 2.4b. The decoding process is implemented as follows:

- Finding a check node corresponding to a parity packet on the right of Fig. 2.4b, where only a single adjacent message node seen on the left of Fig. 2.4b and corresponding to a transmitted degree-one packet is missing. This transmitted packet can be recovered by replacing its value by that of the parity check packet corresponding to the check node on the right of Fig. 2.4b.

- Once a degree-one packet was found, its effect can be removed from all the other packets which relied on it by taking their XOR-function. This allows us to remove the corresponding right node in Fig. 2.4b, its left neighbor node and all edges adjacent to its left neighbor node from this subgraph, as seen in Fig.2.4c. The values of the immediately adjacent parity packets corresponding to the right nodes having a connection with this left node are given by the modulo-2 of their values and this left node's value.

- Repeating the above two steps, until all packets are recovered and hence there are no more nodes of degree one available on the right and then stopping the decoding

process, as seen in Figs. 2.4d and 2.4e.

- The decoding process is deemed successful if it does not halt until all packets are recovered and hence all edges are removed.

To elaborate a little further, the overall code $C(\xi_0, \xi_1, \cdots, \xi_m, C_0)$ is constituted by a cascade of the bipartite graphs $(\xi_0, \xi_1, \cdots, \xi_m, C_0)$, as seen in Fig. 2.2. We analyse the decoding process of the Tornado code based on the process of the subgraph ξ. This graph consists of all nodes on the left that were erased, all the nodes on the right and all the edges connecting these nodes. We assume that the nodes of degree one can be chosen uniformly at random at each step. The decoding process is implemented as seen in Fig. 2.4, where Fig. 2.4a portrays the original subgraph of the encoded packets generated by the Tornado encoder. Fig. 2.4b represents the subgraph after some packets were erased, namely those corresponding to the nodes on the right of this subgraph. At the first cycle of the packet recovery process, a degree-one node is chosen randomly to start recovering the right nodes. Figs. 2.4b, 2.4c, 2.4d, 2.4e portray the decoding cycles of the packet recovering process. The decoding process is completed after four decoding cycles, when all erased right nodes are recovered. The degrees of the left nodes and right nodes are defined by the generating polynomials $\lambda(x)$ and $\rho(x)$, respectively. These polynomials are given by:

$$\lambda(x) = \sum_i \lambda_i x^{i-1} \tag{2.2}$$

$$\rho(x) = \sum_i \rho_i x^{i-1}, \tag{2.3}$$

where λ_i is the initial fraction of edges having a degree i for the left nodes and ρ_i is the fraction of edges having a degree i for the right nodes. The average degree $\overline{d_l}$ of the left nodes equals $\frac{1}{\sum_i \frac{\lambda_i}{i}}$. Let E be the number of edges in the graph, then the number of left nodes having degree i is $(E\lambda_i/i)$ and hence the number of left nodes N_l is calculated as follows:

$$N_l = E \cdot \sum_i \frac{\lambda_i}{i}. \tag{2.4}$$

The Tornado codes used in practice tend to have much fewer cascade graphs than that suggested by the corresponding theoretical analysis. The assumption of experiencing independent erasures for each symbol is crucial in the analysis of Tornado Codes. If this assumption is not satisfied, then a performance loss would be experienced. In practice the Internet is typically modelled by the Binary Erasure Channel (BEC), hence this assumption does not hold. Therefore, it is natural that the practical implementations only require

a small number of cascades. Furthermore, the length of Tornado codes is typically quite high and this results in a high encoding and decoding complexity. Finally, when the rate and the number of input packets n are fixed, the number of encoded packets generated is also fixed. In conclusion, Tornado codes are capable of improving the encoding and decoding more effectively than Reed-Solomon codes, when communicating over erasure channels.

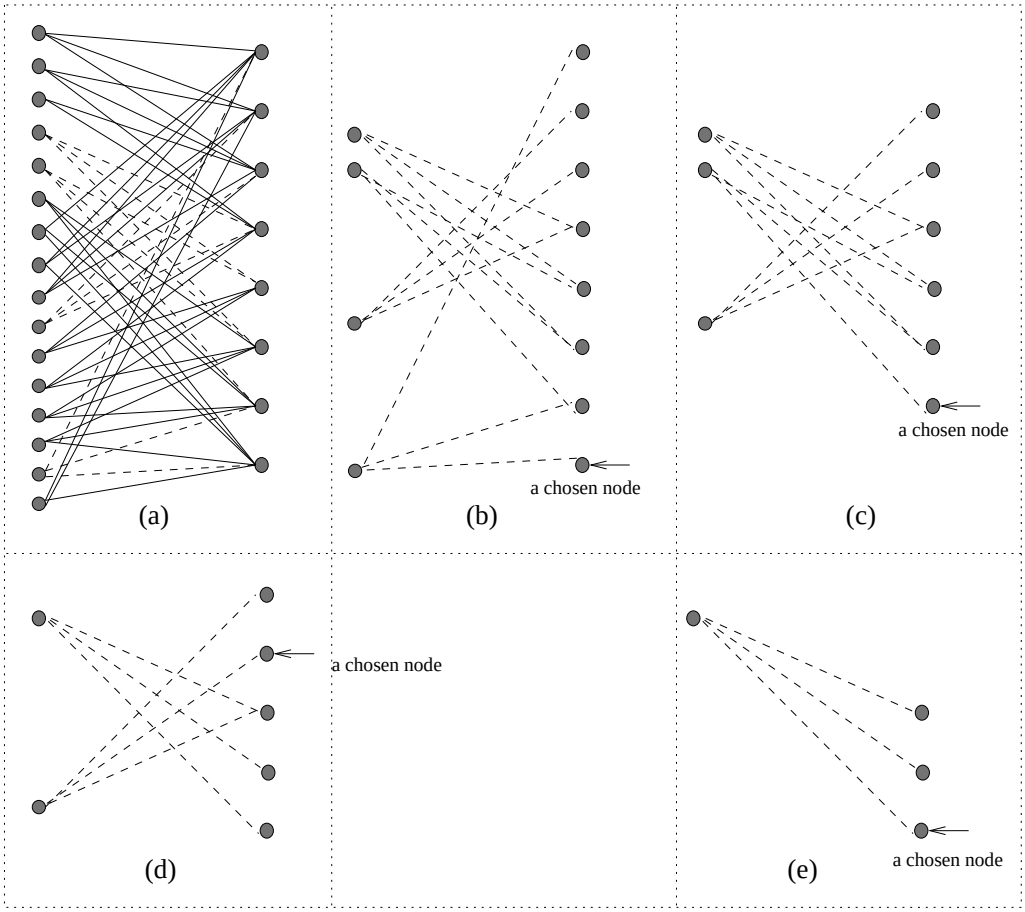


Figure 2.4: An example of the Tornado decoding process: (a) The original graph from the Tornado encoder; (b) The graph after some nodes on the right already were erased;(c)-(e) The recovery erased node process.

2.2 Fountain Codes

The encoder of fountain codes produces an endless supply of encoded packets [36]. The decoder can decode the original source packets from any set of transmitted packets N having a size slightly larger than the number of source packets K. Fountain codes are referred to as being rateless, if the number of source packets is potentially limitless. The number of encoded packet generated is best determined 'on the fly', i.e. in real-time, depending on the channel quality. Fountain codes are near-capacity codes for every erasure channel and we can also design them to have a low encoding and decoding complexity. Regardless of the erasure probabilities of the BEC, the transmitter can send as many encoded packets as needed by the receiver to recover the source data. There are several types of fountain codes, such as random linear codes [37], Luby Transform codes [10], Raptor codes [11]. To start with, we analyse the first fountain code named the random linear code.

2.2.1 Random Linear Codes

To unify our terminology, from now we refer to a packet as our standard unit of encoding, decoding and transmission. A 'packet' contains many bits and a file will be divided into many packets. We will analyse the encoding process of a data file having K source packets $x_1, x_2, \cdots, x_K$. At each encoding cycle i, an encoded packet P_i will be generated as the following equation:

$$P_i = \sum_{j=1}^{K} x_j \cdot \mathbf{G}_{ji} \qquad i = 1, \cdots, N' \,, \tag{2.5}$$

where the sum is the bitwise sum-modulo-2 of the source packets corresponding to the element $\mathbf{G}_{ji} = 1$ of the generator matrix $\mathbf{G}$, as can be seen in Fig. 2.5. At the transmitter side, N' encoded packets ($P_1,\ P_2, \cdots,\ P_{N'}$) are created from K source input packets by using (2.5). Then these encoded packets are transmitted over the erasure channel having an erasure probability δ and $(\delta \cdot N')$ transmitted packets are erased. Thus, at the receiver side, we receive $[N = N' \cdot (1 - \delta)]$ encoded packets ($P'_1,\ P'_2, \cdots,\ P'_N$) and a generator matrix $\mathbf{G'}_{[K \times N]}$ created from these N received packets. If N is smaller than K the decoder can not recover the source packets. In case $N{=}K$ and if the matrix $\mathbf{G'}$ having a size of $(K \times K)$ is an invertible matrix, the decoder can recover the source packet by using the

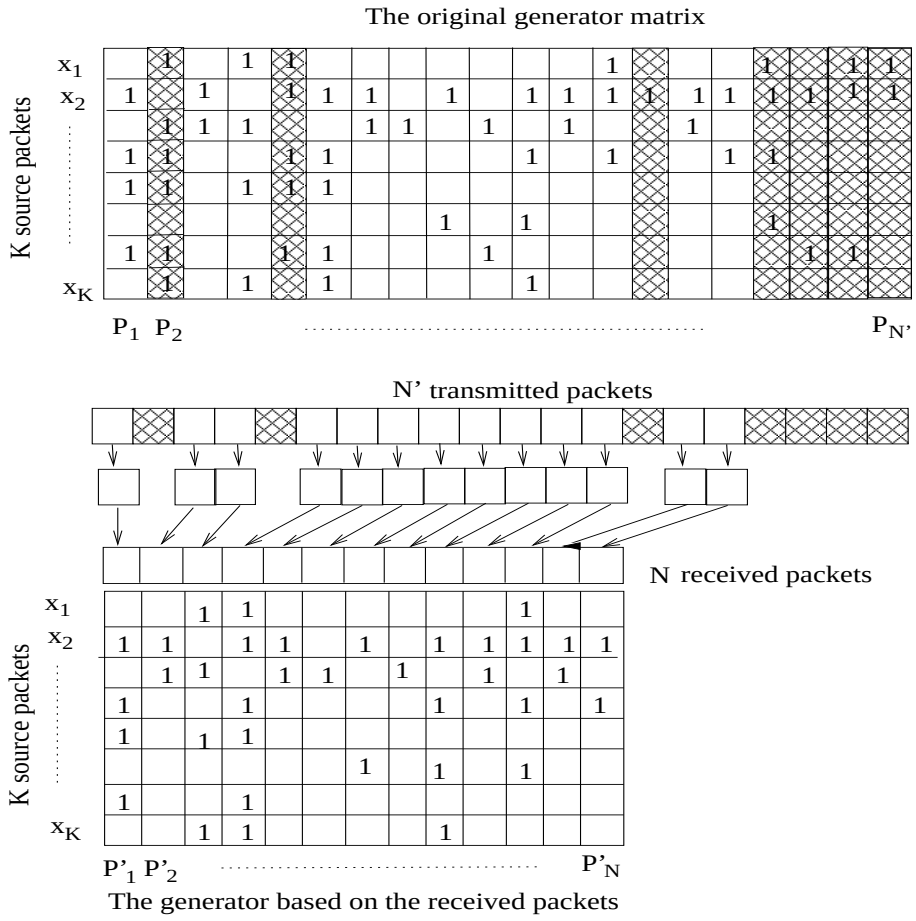


Figure 2.5: A generator matrix of random linear codes.

Gaussian elimination:

$$x_j = \sum_{i=1}^{N} P'_i \cdot \mathbf{G'}^{-1}_{ij} \qquad i = 1, \cdots, N \ . \tag{2.6}$$

In this case, the probability, at which $\mathbf{G'}_{[K\times K]}$ is an invertible matrix, depended on the size of K and can be proved as follows:

- The probability of the first column of $\mathbf{G'}_{[K\times K]}$ is not the all-zero column equals $(1 - 2^{-K})$
- The probability of the second column that it is neither the all-zero column nor the first column is $(1 - 2^{-(K-1)})$
- Repeating, we have the probability that $\mathbf{G'}_{[K\times K]}$ is an invertible matrix equals the product of the element probabilities $(1 - 2^{-K}) \cdot (1 - 2^{-(K-1)} \times \cdots \times (1 - \frac{1}{4}) \cdot (1 - \frac{1}{2})$.

With K being bigger than 10 we have the probability of just 0.289 and it is too small to ensure that $\mathbf{G'}_{[K\times K]}$ is an invertible matrix, it means that the probability, at which the decoder can recover the whole source packets, is very small. Hence, for the sake of recovering all source packets with a very high probability $p = 1 - \delta \approx 1$, in case $N = K$ we need the number of source packets K to be huge.

When N is slightly bigger than K and let $N = K + \epsilon$, where ϵ is the small number of excess encoded packets. Let $(1 - \delta)$ be the probability that $\mathbf{G'}_{[K\times N]}$ is an invertible matrix. So the probability of failure in decoding the K source packets from the N received packets is δ and this probability is bounded by $\delta(\epsilon) \leq 2^{-\epsilon}$. The probability that one column of $\mathbf{G'}_{[K\times N]}$ is an all-zero column equals:

$$(1 - \frac{1}{K})^N \approx e^{-\frac{N}{K}}. \tag{2.7}$$

For general N, the expected probability that K columns of $G'_{[K\times K]}$ are all-zero equals $K \cdot e^{-\frac{N}{K}}$. Hence, for the sake of avoiding having at least one zero-column in the $\mathbf{G'}_{[K\times N]}$ with the probability smaller than δ we have to satisfy the following inequality

$$\delta \leq K \cdot e^{-\frac{N}{K}}, \tag{2.8}$$

which yields

$$N > K \cdot \ln\frac{K}{\delta}. \tag{2.9}$$

Thus we can conclude that when the number of source packets K increases, the performance of random linear codes can get arbitrarily close to the Shannon limit. The cost for encoding random linear codes is $\frac{K}{2}$ and the cost for the decoding process is $(K^3 + \frac{K^2}{2})$. Hence, the totall cost for encoding and decoding processes of the random linear code equals $(\frac{K}{2} + \frac{K^2}{2} + K^3)$.

Understanding the encoding and decoding processes of the random linear code, we can easily analyse another type of Fountain codes-the Luby transform code, which was invented by Michael Luby in 2002 [10]. Naturally, the encoding of these codes are based on the same method used by the random linear codes. The generator created by a degree distribution and the like-Tornado erasure decoding process applied for the Luby transform will be described in the next section.

2.2.2 Luby Transform Codes

LT codes were proposed by Michael Luby in 2002 [10] as a better approximation to the digital fountain approach. Unlike Tornado Codes, these codes are rateless. Their design

does not depend on the estimate of erasure probability of the channel. Suppose that the original file is divided into n message packets. The receivers can recover these message packets with probability $(1-\delta)$ when any $\{n + O[\sqrt{n} \cdot \log^2(n/\delta)]\}$ packets have been received. The time for encoding each symbol is proportional to $O[\log(n/\delta)]$. The time for decoding each symbol is proportional to $O[n \cdot \log(n/\delta)]$. Thus LT codes have higher complexity than Tornado codes. The LT encoding process is implemented as follows:

- Generate a random degree d from the degree distribution $\mu(d)$ which will be analysed in more detail in Section 2.2.5.1.
- Randomly select a packet incident on each of d edges. The value of the encoding LT packet is the modulo 2 (XOR) of the neighboring input packets.

The LT decoding process is virtually the same as that of the Tornado codes [19] [38]. When the decoding process initiates all message packets are uncovered. At the first step, all degree one encoding packets get released to cover their unique neighbor. This set of covered message packets that have not been processed yet form a ripple. At each subsequent set, one message packet from the ripple is selected randomly and processed. It is removed as a neighbor of all encoding packets. Any encoding symbol that has degree one is now released and its neighbor is covered. If the neighbor is not in the ripple it gets added to the ripple. The process ends when the ripple is empty. It fails if at least one message packet is uncovered. An example of the LT decoding process is given in Fig. 2.6. Fig. 2.6 presents an example of the LT decoding process, where we observe three source

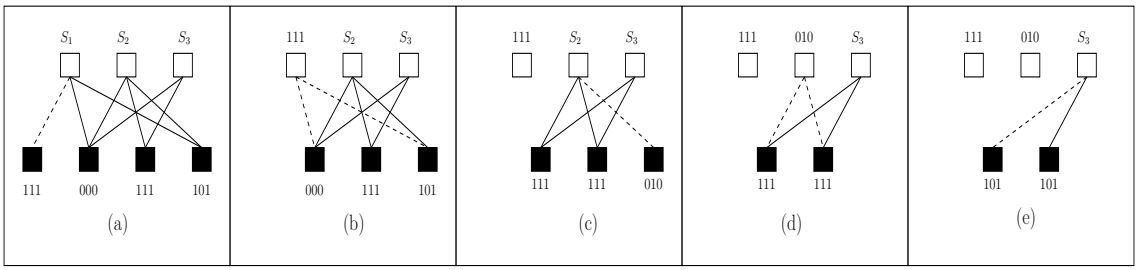


Figure 2.6: Decoding of a LT code having K=3 source packets and N=4 transmitted packets each containing 3 bits; for transmission over the BEC in the absence of errors adopted from [39].

packets S_i, each packet containing (n=3) information bits. These packets are represented by hollow squares, which were previously encoded into four transmitted packets represented by the filled black squares. The value of each bit within the transmitted packet is

the result of the XOR operation of the corresponding source packets connected to it. The decoding process is implemented as follows.
In the first cycle of the decoding process as seen in Fig. 2.6(a), the LT decoder determines, which of the received packets has a degree of one, indicating that this is a self-contained packet, which was not combined with any other source packet using the XOR operation. Hence the decoding operation is simply constituted by outputting the corresponding source packet as indicated by the dashed line in Fig. 2.6(a). At the same time, the decoder finds other transmitted packets, which are connected to this source packet, as shown in Fig. 2.6(b). In order to further exploit the encoding rules of the specific LT code used, as seen in Fig. 2.6(b), the decoder erases the already exploited connections. As the next decoding step, the received packets that have a link to the decoded degree-one packet are updated by the XOR adding their value to the value of each related ie, connected source packet. More explicitly, in this example the first transmitted degree-one packet found by the decoder is the packet having the value of "111" and the corresponding source packet is S_1. Hence, in the next decoding cycle seen in Fig. 2.6(b) S_1 is decoded a value "111" and the connections drawn from this packet to the second and the fourth transmitted packets using dashed lines are erased. Correspondingly, the values of the second and fourth transmitted packets change from "000" and "101" in Fig. 2.6(b) to "111" and "010", as observed in Fig. 2.6(c). At the end of the LT decoding process all of the three source packets S_1, S_2 and S_3 are recovered, as seen in Fig. 2.6(d) and 2.6(e).
For the sake of improving the Packet Error Ratio (PER) of the Luby transform code, Luby et al proposed other version of the Fountain code, which is termed as the Raptor code. The structure and the encoding, decoding processes of the Raptor code are analysed in the next section.

2.2.3 Raptor Codes

Raptor codes constitute an extension of LT Codes [10]. The parameters of a Raptor code of length k over a field F are given by a pre-code C of dimension k and block-length n over F, and a probability distribution Ω on F^n. Given k source packets $\{x_1, \cdots, x_k\}$, the pre-code firstly encodes these symbols into a codeword $\{y_1, \cdots, y_n\}$ of length n. Each output packet is obtained by sampling from the distribution Ω to obtain a vector $\{P_1, \cdots, P_n\}$. The value of the output packet is then obtained as $(\sum_{i=1}^{n} P_i y_i)$. Ω can be described by the numbers $(\Omega_1, \cdots, \Omega_n)$ such as the probability of a vector $\{p \in F_n\}$. Ω has the

generating polynomial ($\Omega(x) = \sum_{i=1}^{n} \Omega_i \cdot x^i$). The parameters of the Raptor code then become $[k, C, \Omega(x)]$. Note that a Raptor code does not have a fixed block-length. The Raptor code can be used to produce a potentially limitless stream of output packets. The design problem in this case consists of choosing the parameters of the Raptor code in such a way that efficient decoding is possible after reception of $[k \cdot (1 + \varepsilon)]$ output packets, for ε arbitrarily close to zero. The decoding algorithm can be called as an algorithm overhead ε. An encoding algorithm is called linear time if the pre-code can be encoded in linear time, and the average number of operations to produce an output packet is a constant. A decoding algorithm is called linear time if, after collecting a certain number of output packets, it can decode the k source packets in time $O(k)$. The Belief Propagation (BP) graph of Raptor codes is drawn in Fig. 2.7. This graph consists of two parts, the first part of the graph is the BP of LDPC codes and the other is the BP graph have the degree distribution $\Omega(x)$.

The first BP graph is the BP graph of a special class of LDPC codes which was designed

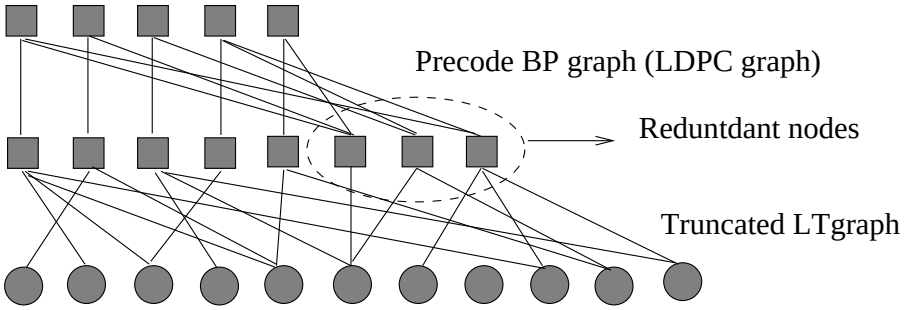


Figure 2.7: The BP graph of the Raptor code

as a pre-code of Raptor codes. Let ζ be a bipartite graph of LDPC codes with n left and r right nodes, as seen in Fig. 2.8. The left nodes are referred to as message nodes or variable nodes and the right nodes are referred to as check nodes of the bipartite graph. The linear code associated with the graph is of block-length n. The coordinate positions of a codeword are identified with the n message nodes. The codewords are those vectors of length n over the binary field such that for every check node the sum of its neighbors among the message nodes is zero. BP decoding of LDPC codes over an erasure channel is very similar to the BP decoding of LT Codes [10], [36]. It has been shown in [40] that this decoding algorithm is successful if and only if the graph induced by the erased message positions does not contain a stopping set. A stopping set is a set of message nodes such that their caused graph has the property that all the check nodes have degree greater

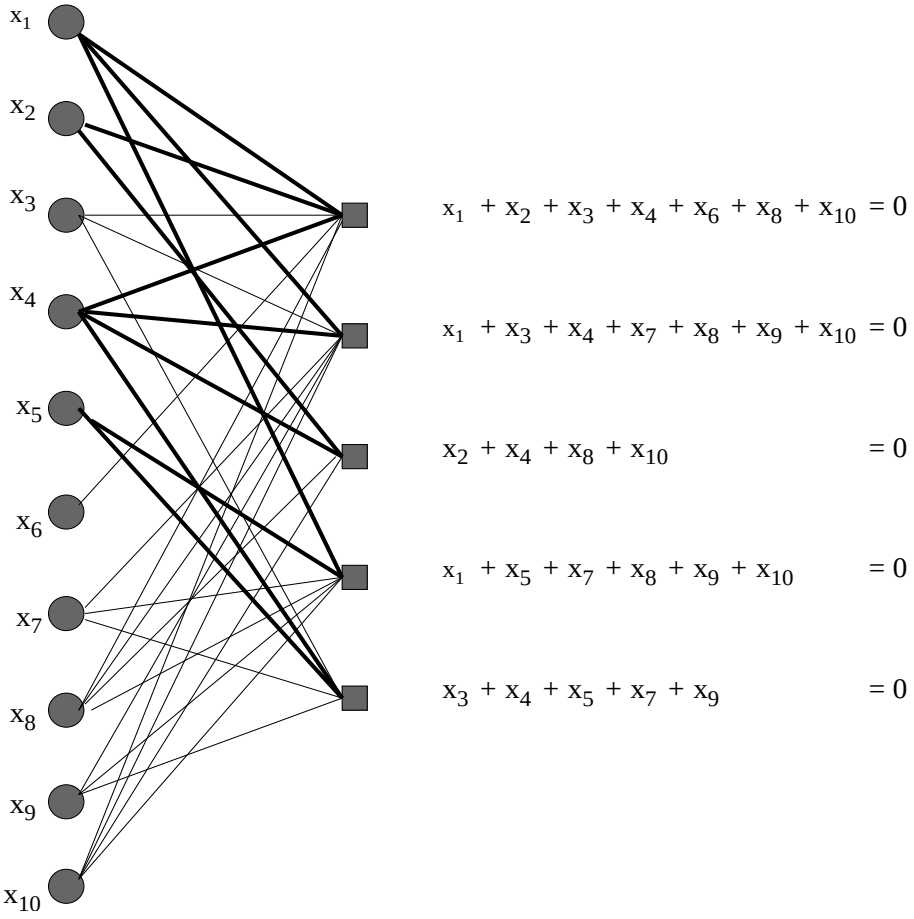


Figure 2.8: An bipartite graph of LDPC codes

than one. For example, in Fig. 2.8 the message nodes 1, 2, 4, 5 generate a stopping set of size 4. The union of two stopping sets can break a stopping set when the decoding is implemented. The LDPC code used as a pre-code in the Raptor code is constructed from a node degree distribution $[\Lambda(x) = \Sigma_d \Lambda_d x^d]$. The neighboring check node is generated from each of the n message nodes as follows:

- A degree d is chosen from the degree distribution $\Lambda(x)$.
- Then d random check nodes are chosen which constitute the neighbors of the message node.

The symbol $\mathbf{P}(\Lambda(x), n, r)$ denotes for the ensemble of bipartite graphs which are defined as above. Let ζ be a random bipartite graph in the ensemble $\mathbf{P}(\Lambda(x), n, r)$. The upper bound of the probability P that ζ has a maximal stopping of size s is calculated in [41]

and approximately equals to:

$$P = \begin{pmatrix} n \\ s \end{pmatrix} \sum_{z=0}^{r} P_z(s,t) \left(1 - \sum_{d} \Lambda_d \frac{\begin{pmatrix} r-z \\ d \end{pmatrix}}{\begin{pmatrix} r \\ d \end{pmatrix}} \right)^{n-s} , \tag{2.10}$$

where $P_n(z,t)$ is the probability that the graph ζ has z check nodes of degree zero and t check nodes of degree one. $P_n(z,t)$ is defined as follows:

$$\begin{aligned} P_0(r,0) &= 1, \\ P_0(z,t) &= 0 \text{ for } (z,t) \neq (r,0), \\ P_{n+1}(z,t) &= \\ \sum_{l,k} P_n(l,k) &\cdot \sum_{d} \Lambda_d \frac{\begin{pmatrix} l \\ l-z \end{pmatrix} \begin{pmatrix} k \\ k+l-z-t \end{pmatrix} \begin{pmatrix} r-l-k \\ d-k-2l+2z+t \end{pmatrix}}{\begin{pmatrix} r \\ d \end{pmatrix}} \\ \text{for } n \geq 0 & \end{aligned} \tag{2.11}$$

We can describe the encoding and decoding of the Raptor code by using the matrix interpretation. The encoding process for a Raptor code is the process of performing multiplications of matrices with vectors and solving sets of equations. All coefficients of involved matrices are binary having values of zero or one. The vectors are vectors of packets, they contain binary vectors and considered as row vectors. Assuming that from the above random bipartite graph ζ we can calculate the generator matrix $\mathbf{G}_{[k\times n]}$ of the pre-code C. Let $\overline{x}$ denote a row vector consisting of the input packets $\{x_1, \cdots, x_K\}$ and $\overline{y}$ denote the vector of intermediate packets. The pre-encoding step of the Raptor code corresponds to the multiplication $(\overline{y} = \overline{x} \cdot \mathbf{G}_{[k\times n]})$. Each output packet of the Raptor code is created from the intermediate packets by sampling independently from the distribution $\Omega(x)$. The output packet is calculated as a scalar product of $(\overline{v} \cdot \overline{y}^T)$, where $\overline{v}$ is called as the vector corresponding to the output encoded packets. For any given set of N output encoded packets there is a corresponding matrix $\mathbf{S}_{[N\times n]}$ in which the rows are the vectors corresponding with the output encoded packets. Hence, we have a set of equations as follows:

$$\mathbf{S} \cdot \overline{x} \cdot \mathbf{G} = \overline{z}^T, \tag{2.12}$$

where $\overline{z}$ is the column vector of final output encoded packets $\{z_1, \cdots, z_N\}$ of the Raptor code. The decoding process of the Raptor code is the solving process the set of equations 2.12.
When designing the Raptor code we need to consider some aspects as follows:

- Raptor codes require storage for the intermediate packets generated by the pre-code. Hence, when designing Raptor codes we need to consider about the space of the memory to store the intermediate packets.
- The overhead of Raptor codes is a function of the decoding algorithm used, and is defined as the number of output packets that the decoder needs to collect in order to recover the input packets with high probability. The overhead of Raptor codes equals $[(1+\epsilon) \cdot k]$, where k is the number of input packets.
- The cost of Raptor codes is the cost of encoding process and two decoding processes that are the pre-code and Luby transform decoding processes.

2.2.4 Systematic Raptor Codes

One of the disadvantages of Luby transform codes and Raptor codes is that these codes are not systematic codes. This means that the input packets are not necessarily reproduced by the encoder. The systematic Raptor code was designed for the sake of attaining better Packet Error Ratio (PER) performances. A Raptor code with parameters $[K, C, \Omega(x)]$ has a reliable decoding algorithm of overhead $(1+\epsilon)$. Let n denotes the block length of the pre-code C. The encoder having input packets $\{x_1, \cdots, x_K\}$ and the corresponding indices $(i_1, \cdots, i_K)$ creates the encoded packets $\{z_1, \cdots, z_{K(1+\epsilon)}\}$. The encoded packets $\{z_{i_1}, \cdots, z_{i_K}\}$, which have the indices $(i_1 \cdots, i_K)$, are the systematic packets. The output encoded packets having the indices $(i_{K+1}, \cdots, i_{K(1+\epsilon)})$ are the non-systematic packets. Firstly, the systematic packets $(i_1, \cdots, i_K)$ are calculated and this process yields an invertible binary matrix $\mathbf{R}$ having a size of $(K \times K)$. These packets are created by sampling $[K \cdot (1+\epsilon)]$ times from the distribution $\Omega(x)$ independently to obtain vectors $\{v_1, \cdots, v_{K(1+\epsilon)}\}$ and applying the decoding algorithm to these vectors. The matrix $\mathbf{R}$ is the product of the matrix $\mathbf{A}$ of $\mathbf{S}$ consisting of rows $\{v_{i_1}, \cdots, v_{i_K}\}$ and a generator matrix $\mathbf{G}$ of the pre-code C, where $\mathbf{S}$ having rows which are the vectors of the encoded packets. These vectors also used to create the first $[K \cdot (1+\epsilon)]$ output packets of the systematic encoder. The intermediate packets y_i are created from the input packets $(x_1, \cdots, x_K)$ using

the generator matrix $\mathbf{R}$. The output packets contain the input packets $(x_1, \cdots, x_K)$ at the systematic positions. Hence, the decoding process has two steps, the first step is decoding the intermediate packets from the received packets and the second step is decoding the source packets from the intermediate packets. The matrix $\mathbf{R}$ and the systematic indeces are calculated by the following algorithm:

- A Raptor code having parameters $[K, C, \Omega(x)]$ and a positive real number ϵ.
- Sampling $[K \cdot (1+\epsilon)]$ times independently from the distribution $\Omega(x)$ to obtain vector $\overline{v}$.
- Calculating the matrix $\mathbf{S}$ having rows which are vectors $\overline{v}$ and the product $(\mathbf{S} \cdot \mathbf{G})$.
- Using Gaussian elimination to calculate the indeces $(i_1, \cdots, i_K)$ such that the component matrix $\mathbf{R}$ of $(\mathbf{S} \cdot \mathbf{G})$ is an invertible matrix and calculating $\mathbf{R}^{-1}$. If the rank of $(\mathbf{S} \cdot \mathbf{G})$ is smaller than K then the process stops.
- Outputting the row vector $\{\overline{v} = (v_1, \cdots, v_{K(1+\epsilon)})\}$, the indices $(i_1, \cdots, i_K)$ and the invertible $\mathbf{R}$.

The probability of failure to decoding the Raptor code equals the probability of failure of the above process. The product of $(\mathbf{S} \cdot \mathbf{G})$ can be calculated with $O(N^2 \cdot K)$ operations. the invertible matrix $\mathbf{R}$ and the indeces $(i_1, \cdots, i_K)$ can be calculated by Gaussian elimination with $O(K^3)$. The cost $L(\mathbf{S}^{-1})$ for calculating $\mathbf{S}^{-1}$ equals $O(K^2)$ for any $(K \times K)$ matrix. The multiplication $(\overline{x} \cdot \mathbf{R})$ can be implemented by firstly multiplying $(\overline{x} \cdot \mathbf{G})$ to obtain the vector $\overline{y}$ and then multiplying vector $\overline{v}$ with $\overline{y}^T$. Hence, the cost $L(\mathbf{R})$ is upper bounded by $[(1+\epsilon)\Omega'(1) + \gamma + o(1)]$, where γ is the cost of encoding of the pre-code C and $o(1)$ is a function, which will approach 0 if K increases to infinity. We can summarize the total cost of the encoding and decoding processes of the systematic Raptor code as follows:

- The probability of failure to decode of the Raptor code equals the probability of failure to implement the process to calculate the invertible matrix and indeces at the encoding process.
- The cost to calculate matrix $\mathbf{R}$ equals $O(K^3 + N^2K)$ operations.
- The cost $L(\mathbf{R}^{-1})$ to calculate the inverted matrix $\mathbf{R}^{-1}$ equals $O(K^2)$

- the cost $L(\mathbf{R})$ is bounded by $[(1+\epsilon)\Omega'(1)+\gamma+o(1)]$ at high probability, where γ is the cost of encoding process of the pre-code C and $o(1)$ is a function approaching 0 when K approaches infinity.

The encoding and decoding processes of the systematic Raptor code are summarized as bellows:

- The encoding process of the systematic Raptor code can be described by the following algorithm:
  * Input packets $\overline{X} = (x_1, \cdots, x_K)$.
  * Calculate the vector $\{\overline{y} = (y_1, \cdots, y_K)\}$ from the equation $(\overline{y} = \overline{X} \cdot \mathbf{R}^{-1})$.
  * Encoding y by using the generator matrix $\mathbf{G}$ of the pre-code C to obtain a vector $\{\overline{u} = (u_1, \cdots, u_N)\}$ by the equation $(\overline{u} = \overline{y} \cdot \mathbf{G})$.
  * Calculating $\{z_i = v_i \cdot \overline{u}^T\}$, where $(1 \leq i \leq K+\epsilon)$.
  * Generating the output packets $\{z_{K(1+\epsilon)+1}, z_{K(1+\epsilon)+2}, \cdots\}$ by applying the LT code encoding process with parameters $[K, \Omega(x)]$ to the vector $\overline{u}$.
  * The first K position of the output encoded packets are systematic packets
- The decoding process of the systematic Raptor code is implemented as follows:
  * The received encoded packets are $\{z_1, \cdots, z_m\}$, where $m = K(1+\epsilon)$.
  * Decoding the received encoded packets using the decoding algorithm applied for the original Raptor code to obtain the intermediate packets $\{y_1, \cdots, y_K\}$. Stop if the decoding of the original Raptor code is not successful.
  * Calculating $\{\overline{x} = \overline{y} \cdot \mathbf{R}\}$, where $\{\overline{x} = (x_1, \cdots, x_K)\}$ and $\{\overline{y} = (y_1, \cdots, y_K)\}$.

Fig. 2.9 is an example of the *Tanner graph* of a systematic Raptor code having two stages, where the first stage is the LDPC graph and the second is the LT graph. The matrix $\mathbf{R}$, $\mathbf{G}$ and $\mathbf{S}$ deduced from Fig. 2.9 are portrayed in Fig. 2.10, Fig. 2.11 and Fig. 2.12, respectively. The degree distribution of the LT code used in this systematic Raptor code is deduced from Fig. 2.9 as follows:

$$\Omega(x) = 0.125x + 0.125x^2 + 0.25x^3 + 0.5x^4. \tag{2.13}$$

With the aid of Fig. 2.9 we can describe the encoding process of this systematic Raptor code as follows:

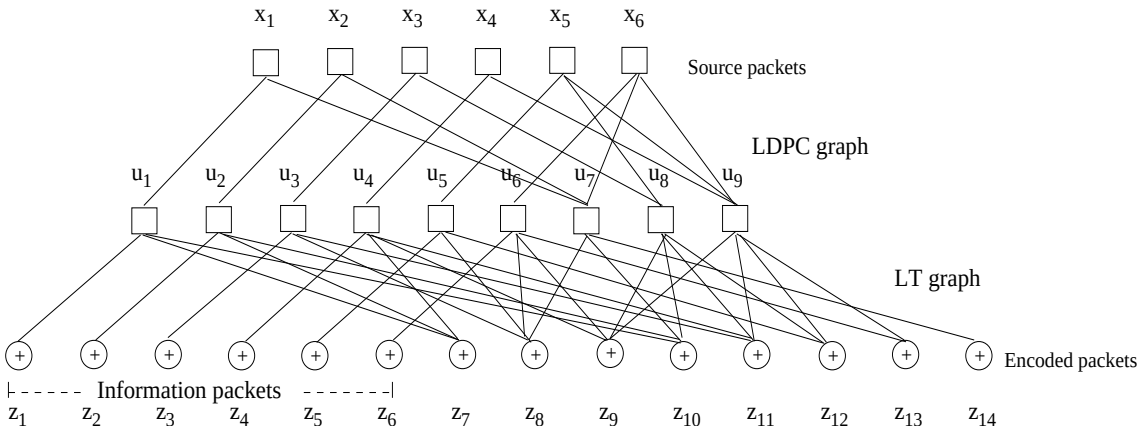


Figure 2.9: An example of the systematic Raptor code having two stages.

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

Figure 2.10: The invertible matrix $\mathbf{R}$.

- The encoding process of the systematic Raptor code portrayed in Fig. 2.9 is implemented as follows:
  - Input packets $(x_1, \cdots, x_6)$, in this case the number of source packets K=6.
  - The matrix $\mathbf{R}_{6\times 6}$ plotted in Fig. 2.10 is the unity matrix. Hence, in this case $\mathbf{R}^{-1}$=$\mathbf{R}$.
  - Calculating the vector intermediate $\{\overline{y} = (y_1, \cdots, y_6)\}$ by the given equation $(\overline{y} = \overline{x} \cdot \mathbf{R}^{-1})$. In this case, we have $\{\overline{y} = (y_1, \cdots, y_6)\} \equiv \{\overline{x} = (x_1, \cdots, x_6)\}$.
  - Encoding the vector $\overline{y}$ to receive the vector $\{\overline{u} = (u_1, \cdots, u_9)\}$ by the equation $(\overline{u} = \overline{y} \cdot \mathbf{G})$, where $N = 9$ and $G_{6\times 9}$ is portrayed in Fig. 2.11.
  - Calculating $\{z_i = v_i \cdot \overline{u}^T\}$, where $(1 \leq i \leq 9)$, ϵ= 3, v_i denotes the elements of rows of the matrix plotted in Fig. 2.12.
  - * Generating the output packets $\{z_{10}, z_{11}, z_{12}, z_{13}, z_{14}\}$ by applying the LT code encoding process with parameters $[6, \Omega(x)]$ to the vector $\overline{u}$, where $\Omega(x)$ in (2.13) is the degree distribution of the LT code.

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

Figure 2.11: The generator matrix **G** of the precode LDPC.

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

Figure 2.12: The matrix having rows which correspond to the vectors $\{v_i\}$, where $i = (1, \cdots, 9)$.

The cost of decoding process of the systematic Raptor code equals $[\alpha(1+\epsilon)+\beta+\gamma+o(1)]$, where α is the cost of encoding process, β is the cost of original Raptor decoding process, γ is the encoding pre-code cost and $o(1)$ is a function approaching to 0 when K increases to infinity [11], [42].

2.2.5 Degree Distributions

There are four different types of degree distributions, which have to be analysed, when we design Luby Transform and Raptor codes. These are the Robust Soliton Degree Distribution (RSDD) [10], the Poisson Degree Distribution (PDD) [11] and the degree distribution of Low Density Parity Check codes used as the pre-code of the Raptor code and the All-At-One Distribution (AAOD) [36]. In this Section we analyse the RSDD, PDD and AAOD distributions.

2.2.5.1 Robust Soliton Degree Distribution

As we known an appropriate choice of the degree of distribution is crucial in the design of LT codes for the sake of maintaining a low PER [10], [36]. Every source packet must have at least one *edge* leading to the received packet, where an *edge* is defined as the connection in Fig. 2.6, indicating the reception of a source packet which was LT-encoded using the time-variant generator matrix defined for the corresponding packet interval. Ideally, upon the successful reception of a packet, a new self-contained degree-one received packet should appear for the sake of avoiding having redundant packets. This objective is achieved by the *ideal soliton distribution* of [39]

$$\rho(d) = \begin{cases} 1/K & \text{for } d = 1, \\ \frac{1}{d(d-1)} & \text{for } d = 2, 3, ..., K, \end{cases} \tag{2.14}$$

which is plotted in Fig. 2.13. However, the above-mentioned ideal behaviour is only achievable with a certain probability [39]. When no degree-one packet is recovered at any stage of the consecutive LT-decoding cycles exemplified in Fig. 2.6, the decoding process will be suspended and therefore the remaining packets cannot be recovered, unless a sufficiently high number of redundant packets is received. Hence the so-called *robust soliton distribution* was designed by Luby [10] for circumventing this problem by increasing the expected number of degree-one encoded packets to $(S \equiv c \cdot log_e(K/\delta)\sqrt{K})$, where the parameter δ denotes the probability of decoding failure imposed by the lack of a degree-one packet, while c represents a constant. More explicitly, taking the above-mentioned parameters into consideration, Luby proposed the improved distribution of [10]:

$$\tau(d) = \begin{cases} \frac{S}{K}\frac{1}{d} & \text{for d=1, 2, ..., } \frac{K}{S} - 1, \\ \frac{S}{K}\log(\frac{S}{\delta}) & \text{for } d = \frac{K}{S}, \\ 0 & \text{for } d > \frac{K}{S}, \end{cases} \tag{2.15}$$

which is also plotted in Fig. 2.13 as a function of the degree d. Finally, for the sake of improving both the distributions of (2.14) and (2.15), Luby introduced the so-called *robust soliton distribution* μ, which is constituted by the superposition of the *ideal soliton distribution* $\rho(d)$ and that of (2.15) as [10]

$$\mu(d) = \frac{\rho(d) + \tau(d)}{Z}, \tag{2.16}$$

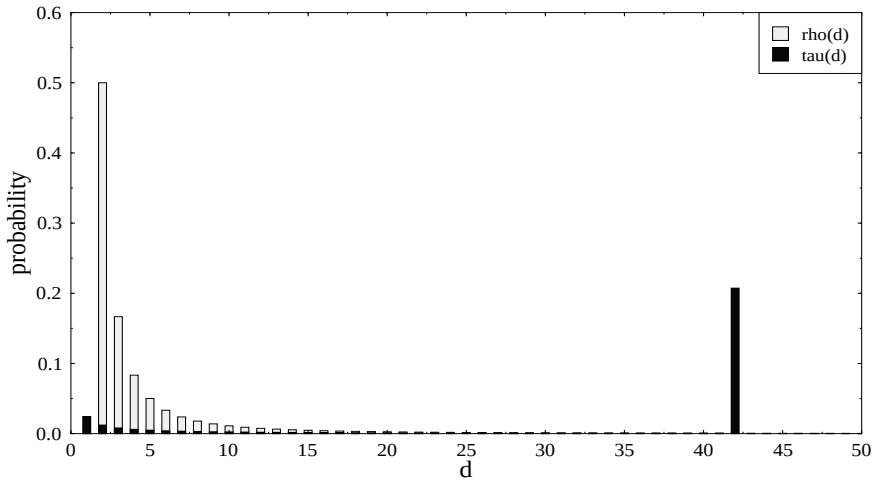


Figure 2.13: The distributions $\rho(d)$ of (2.14), $\tau(d)$ of (2.15) for the case of K=10000, c=0.2, δ=0.05 [10]

where the normalisation factor of the denominator is given by $[Z = \sum_d \rho(d) + \tau(d)]$, ensuring that all the probabilities sum to unity. The function $\mu(d)$ is portrayed in Fig. 2.14. In practice, an LT code is capable of recovering the transmitted file having K packets, if it receives about 5% more packets than the number of original source packets K [39], provided that the channel conditions are adequate. The values of the parameters c and δ in $(S \equiv c \cdot log_e(K/\delta)\sqrt{K})$ determine the number of packets required for recovering all the K source packets. For the sake of getting the higher probability to successfully decode the source packets from encoded packets we can choose other values of δ and c such as $\delta = 0.5$ and $c = 0.1$.

The degree of output encoded packets will be higher and the overhead of the LT code also is required bigger, as seen in Figs 2.15 and 2.16.

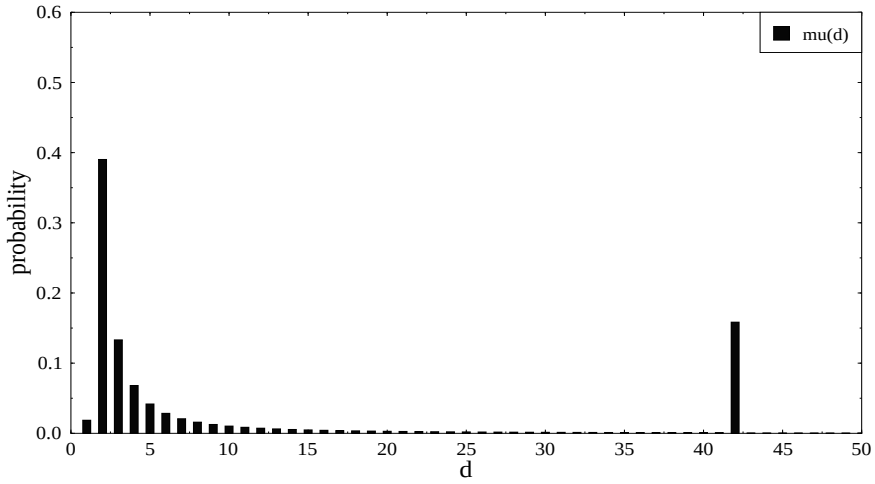


Figure 2.14: The robust distributions $\mu(d)$ of (2.16) for the case of K=10000, c=0.2, δ=0.05

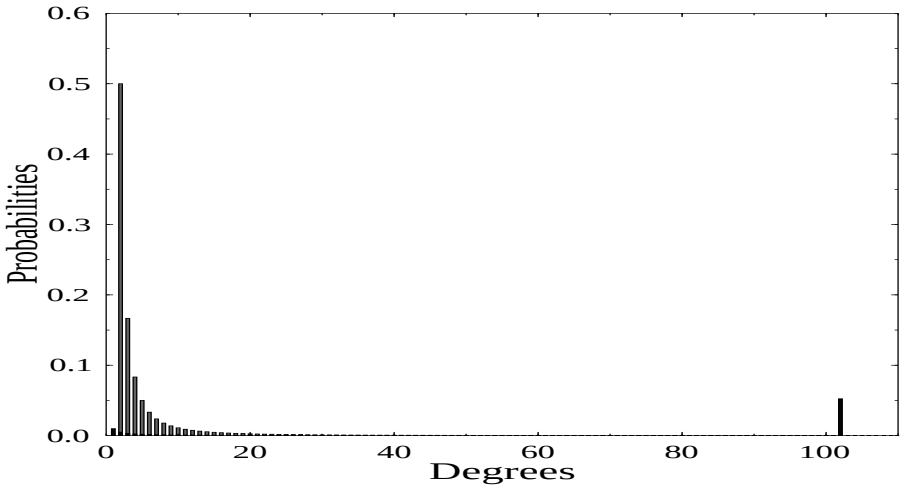


Figure 2.15: The distributions $\rho(d)$ of (2.14), $\tau(d)$ of (2.15) for the case of K=10000, c=0.1, δ=0.5 [10]

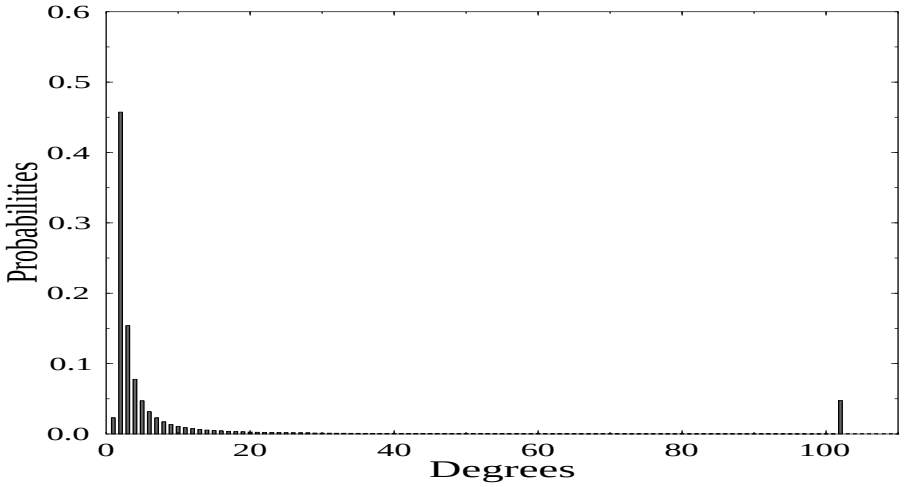


Figure 2.16: The robust distributions $\mu(d)$ of (2.16) for the case of K=10000, c=0.1, δ=0.5

2.2.5.2 Poisson Soliton Degree Distribution

The PDD used for the LT component code of raptor codes in [11], [42] and [42] is defined as follows:

$$\Omega_d(\lambda) = \begin{cases} \frac{\mu \cdot \lambda}{1+\mu} \cdot \frac{1}{\Omega(\lambda)} & \text{for } d = 1, \\ \frac{\lambda^d}{d(d-1)} \cdot \frac{1}{\Omega(\lambda)} & \text{for } d = 2, 3, ..., D, \end{cases} \tag{2.17}$$

where d is the variable degree, $(D+1)$ is the maximum degree and $[D = \frac{4(1+\varepsilon)}{\varepsilon}]$, ε is the real number bigger than zero, λ is the real number and $\{\lambda \in [\delta, 1]\}$, $(\delta = \frac{1}{D})$ and $[\mu = (\frac{\varepsilon}{2}) + (\frac{\varepsilon}{2})^2]$. The function $\Omega(\lambda)$ is defined as follows

$$\Omega(\lambda) = \frac{1}{1+\mu}(\mu \cdot \lambda + \frac{\lambda^2}{1 \cdot 2} + \frac{\lambda^3}{2 \cdot 3} + \cdots + \frac{\lambda^D}{(D-1) \cdot D} + \frac{\lambda^{D+1}}{D}), \tag{2.18}$$

Fig. 2.17 and Fig. 2.18 show examples of the Poisson Distribution function used in [11] For the sake of analysing the performance of a Raptor code having the LT component code using the Poisson distribution we analyse the LT in conjunction with the parameters $[K, \Omega(\lambda)]$, where K is the number of input packets and $\Omega(\lambda)$ is the original generating function of the above Poisson distribution. From (2.17) we have the average of the degree of encoded packets is a and is calculated by following equation:

$$a = \sum_{d}^{D+1} d \cdot \Omega_d \tag{2.19}$$

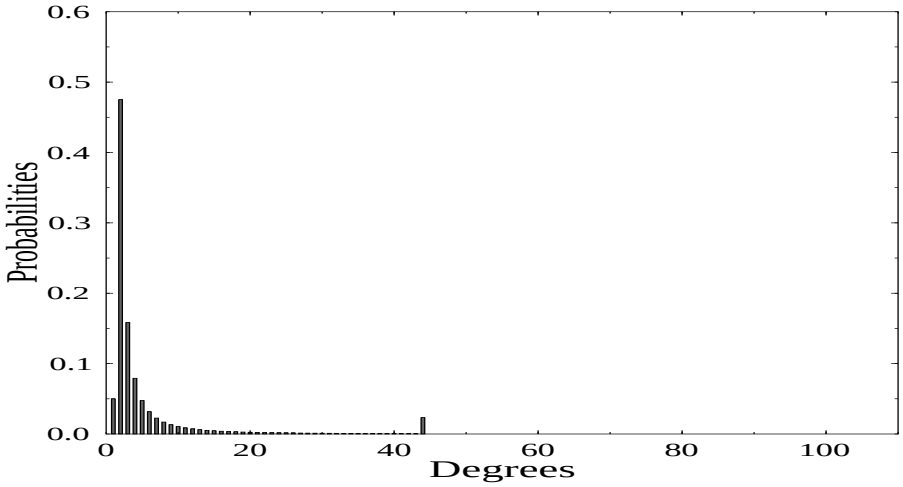


Figure 2.17: The Poisson distributions with parameters $\lambda = 1$, $\varepsilon = 0.1$

The generating function of the input packet degree distribution can be determined from the probability $P_{neighbor}$ of the event that an input packet is the neighbor of exactly l encoded packets. This probability is calculated as follows:

$$P_{neighbor} = \binom{N}{l} \left(\frac{a}{K}\right)^{l} \left(1 - \frac{a}{K}\right)^{N-l}, \tag{2.20}$$

where $\{N = [K \cdot (1 + \frac{\varepsilon}{2}) + 1]\}$ is the number of encoded packets. From (2.20), we have the generating function F of the input packet degree distribution as defined in [11]

$$F = \sum_{l} \binom{N}{l} \cdot (\frac{a}{K})^{l} \cdot \lambda^{l} \cdot (1 - \frac{a}{K})^{N-l} \tag{2.21}$$

The LT code, which has the code rate $[R = (1 + \frac{\varepsilon}{2})/(1 + \varepsilon)]$, can recover all the source packets with the erasure probability δ of the BEC channel which is calculated as follows:

$$\delta = \frac{(\frac{\varepsilon}{4})}{(1 + \varepsilon)} = \frac{(1 - R)}{2}, \tag{2.22}$$

and the cost for its decoding process equals $O[K \cdot \log(\frac{1}{\varepsilon})]$ arithmetic operations. With $\delta = 0.01$, some truncated optimal distributions of Ω_d are listed in Table 2.1 as designed in [11].

From Table 2.1 the generating functions of the Poisson degree distribution having the

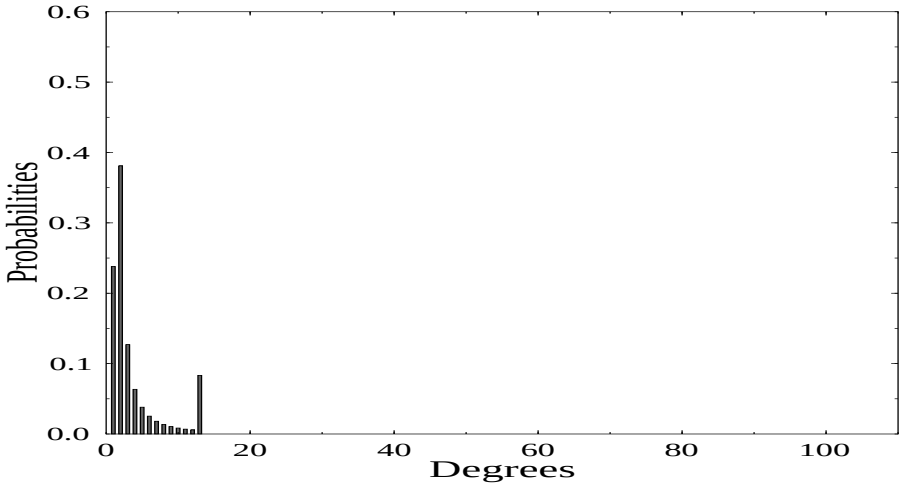


Figure 2.18: The Poisson distributions with parameters $\lambda = 1$, $\varepsilon = 0.5$

number of different input packets K are expressed correlatively as follows:

$$\begin{aligned}\Omega_d(\lambda) &= 0.007965 \cdot \lambda + 0.493570 \cdot \lambda^2 + 0.166220 \cdot \lambda^3 + 0.072646 \cdot \lambda^4 + 0.082558 \cdot \lambda^5 + \\ &\quad 0.056058 \cdot \lambda^8 + 0.037229 \cdot \lambda^9 + 0.055590 \cdot \lambda^{19} + \\ &\quad 0.025023 \cdot \lambda^{65} + 0.003135 \cdot \lambda^{66} \quad \text{for } K = 65536 \qquad (2.23)\\ \Omega_d(\lambda) &= 0.007544 \cdot \lambda + 0.493610 \cdot \lambda^2 + 0.166458 \cdot \lambda^3 + 0.071243 \cdot \lambda^4 + 0.084913 \cdot \lambda^5 + \\ &\quad 0.049633 \cdot \lambda^8 + 0.043365 \cdot \lambda^9 + 0.045231 \cdot \lambda^{19} + 0.010157 \cdot \lambda^{20} + \\ &\quad 0.010479 \cdot \lambda^{66} + 0.017365 \cdot \lambda^{67} \quad \text{for } K = 80000 \qquad (2.24)\\ \Omega_d(\lambda) &= 0.006495 \cdot \lambda + 0.495044 \cdot \lambda^2 + 0.168010 \cdot \lambda^3 + 0.067900 \cdot \lambda^4 + 0.089209 \cdot \lambda^5 + \\ &\quad 0.041731 \cdot \lambda^8 + 0.050162 \cdot \lambda^9 + 0.038837 \cdot \lambda^{19} + 0.015537 \cdot \lambda^{20} + \\ &\quad 0.016298 \cdot \lambda^{66} + 0.010777 \cdot \lambda^{67} \quad \text{for } K = 100000 \qquad (2.25)\\ \Omega_d(\lambda) &= 0.004807 \cdot \lambda + 0.496472 \cdot \lambda^2 + 0.166912 \cdot \lambda^3 + 0.073374 \cdot \lambda^4 + 0.082206 \cdot \lambda^5 + \\ &\quad 0.057471 \cdot \lambda^8 + 0.035951 \cdot \lambda^9 + 0.001167 \cdot \lambda^{18} + 0.054305 \cdot \lambda^{19} + \\ &\quad 0.018235 \cdot \lambda^{65} + 0.009100 \cdot \lambda^{66} \quad \text{for } K = 120000.\end{aligned}$$

Fig. 2.19, Fig. 2.20, Fig. 2.21, Fig. 2.22 show the Poisson degree distributions which are designed for different numbers of input packets K in Table 2.1.

K	65536	80000	100000	120000
Ω_1	0.007969	0.007544	0.006495	0.004807
Ω_2	0.493570	0.493610	0.495044	0.496472
Ω_3	0.166220	0.166458	0.168010	0.166912
Ω_4	0.072646	0.071243	0.067900	0.073374
Ω_5	0.082558	0.084913	0.089209	0.082206
Ω_8	0.056058	0.049633	0.041731	0.057471
Ω_9	0.037229	0.043365	0.050162	0.035951
Ω_{18}				0.001167
Ω_{19}	0.055590	0.045231	0.038837	0.054305
Ω_{20}		0.010157	0.015537	
Ω_{65}	0.025023			0.018235
Ω_{66}	0.003135	0.010479	0.016298	0.009100
Ω_{67}		0.017365	0.010777	
ε	0.038	0.035	0.028	0.02
a	5.87	5.91	5.85	5.83

Table 2.1: Optimal Truncated Poisson Degree Distribution (TPDD) examples

From any set of $(1+\varepsilon/2)\cdot K+1)$ output encoded of the LT code with parameters $[K,\Omega_d(\lambda)]$ are sufficient to recover at least $[(1-\delta)K]$ input packets by the Belief Propagation (BP) decoding with the error probability of at most e^{-cK}, where K is the number of input packets, c is a positive real number and depends on ε and $[\delta=\frac{(\frac{\varepsilon}{4})}{(1+\varepsilon)}]$. We can prove it by considering a set of $[(1+\varepsilon/2)\cdot K+1]$ output packets from the LT encoder which associates to a graph defined by the LT generator matrix. This graph is a random graph with degree distributions $\nu(\lambda)$ and $\omega(\lambda)$ corresponding to the input and output packets,respectively. The above statement will is valid if and only if:

$$\nu(1-\omega(1-\lambda))<\lambda, \tag{2.26}$$

with $\lambda\in[\delta,1]$ as shown in [1]. If an LT code has parameters $[K,\Omega(\lambda)]$ we have the degree distribution of the output encoded packets $[\omega(\lambda)=\frac{\Omega'(\lambda)}{\Omega'(1)}]$ and the degree distribution of the input packets $\nu(\lambda)$ is based on the generating function $\left[\sum_l \binom{N}{l}\cdot(\frac{a}{K})^l\cdot\lambda^l\cdot(1-\frac{a}{K})^{N-l}\right]$ as calculated in (2.20), where a is the average degree of an output packet and N equals to:

$$[N=K(1+\varepsilon/2)+1]\,. \tag{2.27}$$

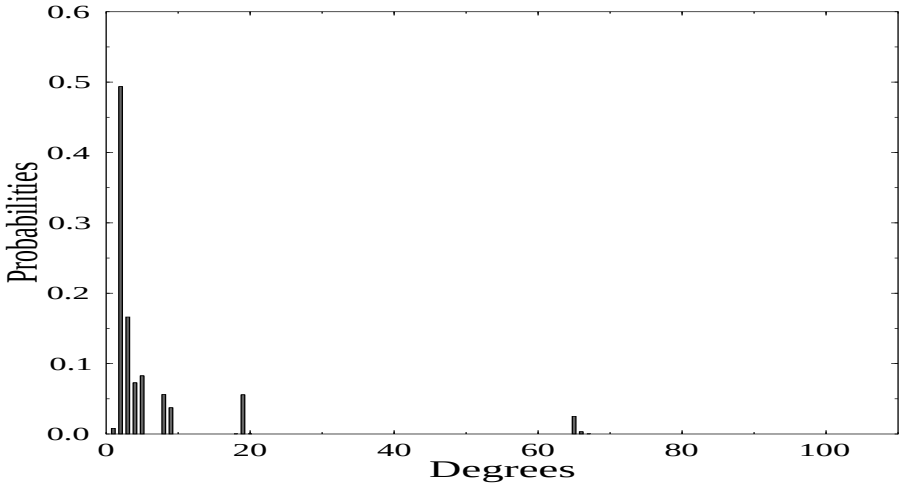


Figure 2.19: The truncated Poisson distributions with parameters $K = 65536$, $\varepsilon = 0.038$

We can re-write the generating function of the input packet degree distribution $\nu(\lambda)$ as follows:

$$\sum_{l} \binom{N}{l} \cdot (\frac{a}{K})^{l} \cdot \lambda^{l} \cdot (1-\frac{a}{K})^{N-l} = \left(1 - \frac{a(1-\lambda)}{K}\right)^{N} = \left(1 - \frac{a(1-\lambda)}{K}\right)^{(1+\varepsilon/2)\cdot K+1}. \quad (2.28)$$

As seen in (2.28), the input packets $\nu(\lambda)$ are normalized with respect to λ, so that we have $[\nu(1) = 1]$. Hence, $\nu(\lambda)$ equals to:

$$\nu(\lambda) = \left(1 - \frac{a(1-\lambda)}{K}\right)^{(1+\varepsilon/2)\cdot K+1}, \quad (2.29)$$

where we have the inequality $[(1 - \frac{n}{m})^{m} < e^{-n}]$[1] with $(n \leq m)$. Hence, we can write:

$$\nu(1 - \omega(1-\lambda)) < e^{-(1+\varepsilon/2)\omega(1-\lambda)} = e^{-(1+\varepsilon/2)\Omega'(1-\lambda)}. \quad (2.30)$$

In order to prove (2.26), we will show that $[e^{-(1+\varepsilon/2)\Omega'(\lambda)} < 1 - \lambda]$ with $\{\lambda \in [0, 1-\delta]\}$. We have the first order derivative $\Omega'(\lambda)$ in the form of:

$$\begin{aligned} \Omega'(\lambda) &= \frac{1}{\mu+1}\left(\mu + \lambda + \frac{\lambda^2}{2} + \cdots + \frac{\lambda^{D-1}}{D-1} + \frac{(D+1)\cdot\lambda^{D}}{D}\right) \\ &= \frac{1}{\mu+1}\left(\mu - ln(1-\lambda) + \lambda^{D} - \sum_{d=D+1}^{\infty} \frac{\lambda^{d}}{d}\right). \end{aligned} \quad (2.31)$$

[1]If (2.26) is satisfied, the graph constituted by the degree distributions $\nu(d)$ and $w(d)$ is a random graph. The random characteristic of the graph defined by the LT generator matrix determines the decoding ability of the LT code [11].

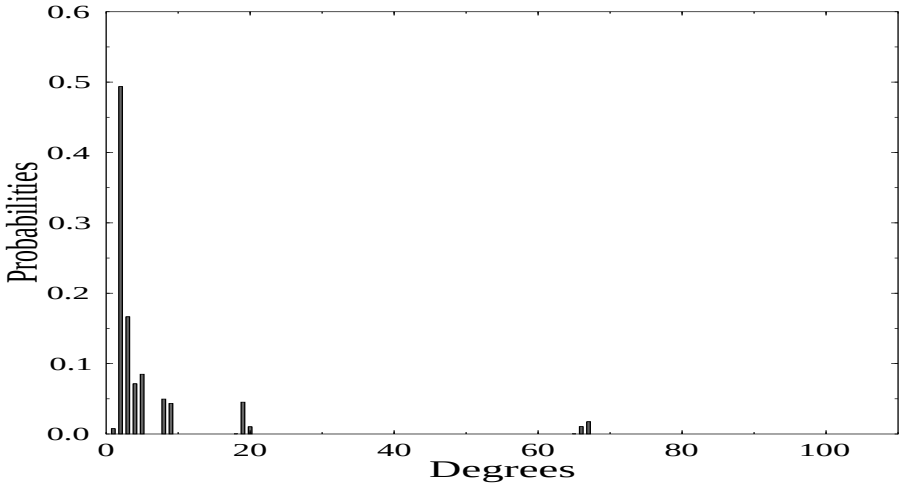


Figure 2.20: The truncated Poisson distributions with parameters $K = 80000$, $\varepsilon = 0.035$

We also recognise that:

$$\lambda^D > \sum_{d=D+1}^{\infty} \frac{\lambda^d}{d} \quad \text{for } \lambda \in [0,\ 1-\delta]. \tag{2.32}$$

To prove (2.32) we have to prove the following inequality:

$$\sum_{d=D+1}^{\infty} \frac{\lambda^{d-D}}{d} < 1. \tag{2.33}$$

The left side of the above inequality is monotonically increasing, so to prove it we only have to prove that it is true at a certain value of λ. Hence we opted for providing it at $(\lambda = 1-\delta)$ with $[\delta = \frac{\varepsilon}{4\cdot(1+\varepsilon)}]$. Then from (2.32) we have:

$$\begin{aligned} \sum_{d=D+1}^{\infty} \frac{(1-\delta)^{d-D}}{d} &< \frac{1}{D+1}\sum_{d=1}^{\infty}(1-\delta)^d \\ &= \frac{1-\delta}{(D+1)\cdot\delta} \\ &\leq \frac{4+3\varepsilon}{\varepsilon}\cdot\frac{\varepsilon}{4+5\varepsilon} \\ &< 1. \end{aligned} \tag{2.34}$$

From (2.31), (2.33) and (2.34) we have:

$$\Omega'(\lambda) > \frac{\mu - ln(1-\lambda)}{\mu+1}. \tag{2.35}$$

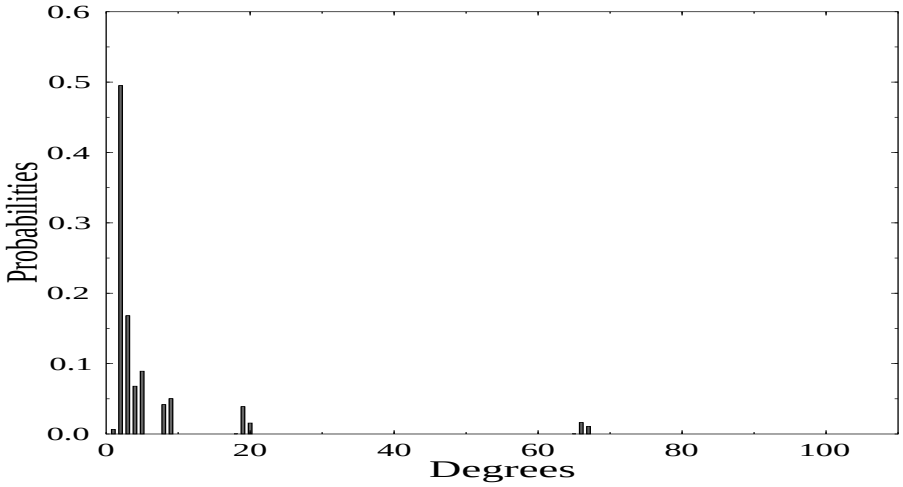


Figure 2.21: The truncated Poisson distributions with parameters $K = 100000$, $\varepsilon = 0.028$

Hence we arrive at:

$$e^{-(1+\varepsilon/2)\Omega'(\lambda)} < e^{-(1+\varepsilon/2)\mu/(1+\mu)}(1-\lambda)^{(1+\varepsilon/2)/(1+\varepsilon)}. \tag{2.36}$$

With $\{\lambda \in [0, 1-\delta]\}$ and $(\mu > \varepsilon/2)$ Inequality (2.36) is satisfied. Hence, the right of Inequality (2.35) is smaller than 1. This means Inequality (2.26) has been proven.

2.2.5.3 All-At-One Degree Distribution

The All-At-One Degree Distribution (AAODD) has the function of the distribution is $\rho(1)= 1$. This corresponds to generating each encoding packet by selecting a random source packet and copying its value to the LT-encoded packet. The analysis of the classical balls and bins [10], [36] process implies that $K \cdot \ln(K/\delta)$ encoded packets are needed to cover all K source packets with probability p at least $p = (1-\delta)$ with respect to the AAODD. This result can be easily modified to show that for this distribution the number of XOR operations must be at least $K \cdot \ln(K/\delta)$ to recover all source packets. Hence, although the total number of XOR operations with respect to the AAODD is minimal, but the number of encoded packets required to recover the K source packets is an unacceptable $\ln(K/\delta)$ times bigger than the number of the source packets [36].

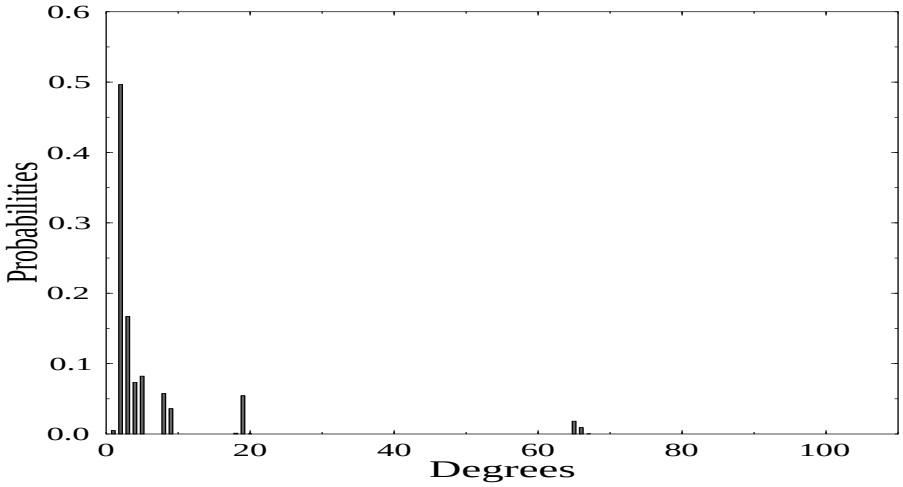


Figure 2.22: The truncated Poisson distributions with parameters $K = 120000$, $\varepsilon = 0.02$

2.2.5.4 Conclusions

The degree distributions decides the Package Error Ratio (PER) as well as the complexity of both LT codes and Raptor codes. When LT codes or Raptor codes use the Poisson distributions, their complexity will reduce but their PER performances also decrease. In contrast, when using the Robust Soliton Degree Distribution (RSDD), the PER performances of LT codes and Raptor codes are improved, but their complexity will increase due to the high density of this degree distribution.

2.3 Chapter Conclusions

For the sake of analysing the history as well as the LT encoding and decoding algorithms, we did analyse several related erasure codes such as Tornado codes, linear random codes and Raptor codes. Actually, wherein the encoding and decoding mechanism of LT codes is based on the Tornado codes and linear random codes. Both LT codes and the above erasure codes create encoded packets by adding XOR randomly the source packets based on a generator matrix. This generator matrix is generated from the different mechanisms according to different erasure codes. For Tornado codes, the generator is created by using

the cascade graph as analysed in Section 2.1.1. Linear random codes create the random generator for their encoding process, while LT and Raptor codes generate the generator matrix by using the different degree distribution functions as mentioned in Section 2.2.5. The decoding method of LT codes termed as the erasure decoding, which is proposed for Tornado codes, starts by choosing the received degree-one packets for replacing the source packets's values and erasing the related edges in the *Tanner graph*. Analysing Raptor codes help us extend our research for designing the combination structure of LT codes with other channel codes such as LDPC, G-LDPC to improve their BER performances. The designs of LT codes and their PER and BER performances in different communication schemes will be presented in Chapter 3.

Chapter 3

Hard-Bit Decoding Algorithms of Luby Transform Codes

3.1 Iterative Decoding of LT Codes and BICM

3.1.1 Improved Robust Soliton Degree Distribution

An appropriate choice of the degree of distribution is crucial in the design of LT codes for the sake of maintaining a low Packet Error Ratio (PER). For example, the LT decoding process was shown in Fig. 2.6. Recall that every source packet must have at least one *edge* leading to the received packets, where an *edge* is defined as a connection seen in Fig. 2.6, indicating the reception of a source packet which was LT-encoded using the time-variant generator matrix defined for the duration of the corresponding packet interval. Ideally, upon the successful reception of a packet, a new self-contained degree-one received packet should appear for the sake of avoiding the transmission of an excessive number of redundant packets[1]. This objective is achieved by the *ideal soliton distribution* of [39] described by:

$$\rho(d) = \begin{cases} 1/K & \text{for } d = 1, \\ \frac{1}{d(d-1)} & \text{for } d = 2, 3, ..., K, \end{cases} \tag{3.1}$$

[1]More explicitly, if the degree distribution of the code fails to sastify this requirement, an unnecessarily high number redundant packets may be required for ensuring that there is always a degree-one packet facilitating the next decoding step

which is plotted in Fig. 3.1. However, the above-mentioned ideal behaviour is only achievable with a certain probability [39]. When no degree-one packet is recovered at any stage of the consecutive LT-decoding cycles exemplified in Fig. 2.6, the decoding process will be suspended and therefore the remaining packets cannot be recovered, unless a sufficiently high number of redundant packets is received. Hence the so-called *robust soliton distribution* was designed by Luby [10] for circumventing this problem by increasing the expected number of degree-one encoded packets in a controlled manner to a value of $S \equiv [c \cdot log_e(K/\delta)\sqrt{K}]$, where the parameter δ denotes the probability of decoding failure imposed by the lack of a degree-one packet, while c represents a constant. More explicitly, taking the above-mentioned parameters into consideration, Luby proposed the improved distribution of [10]:

$$\tau(d) = \begin{cases} \frac{S}{K}\frac{1}{d} & \text{for d=1, 2, ..., } \frac{K}{S} - 1, \\ \frac{S}{K}\log(\frac{S}{\delta}) & \text{for } d = \frac{K}{S}, \\ 0 & \text{for } d > \frac{K}{S}, \end{cases} \tag{3.2}$$

which is then amalgamated with the Ideal Soliton Degree Distribution (ISDD) of (3.1) yielding the Robust Soliton Degree Distribution (RSDD) is given as follows:

$$\mu(d) = \frac{\rho(d) + \tau(d)}{Z}, \tag{3.3}$$

where the normalisation factor of the denominator is given by $[Z = \sum_d \rho(d) + \tau(d)]$, ensuring that all the probabilities sum to unity. The function $\rho(d)$ and $\tau(d)$ of the RSDD is characterized in Fig. 3.2. In this section, we propose an Improved Robust Degree Distribution (IRDD). Referring to Luby's *robust soliton distribution* characterized in Fig. 3.2, we observe that there exist some degree distributions, which have such a low probability P_{d_i} that the number of packets given by the the product of P_{d_i} and K' may be less than one, indicating the potential absence of packets having these degrees, where $K' = K + E$ and E denotes the overhead of the LT code. It is undesirable to invoke this particular distribution in the computation of the LT encoder's generator matrix $\mathbf{G}$. Therefore, in some cases Luby's *robust soliton distribution* defined by (3.3) may lead to premature decoding abortion and a concomitant loss of packets during the decoding process, unless the number of redundant packets is sufficiently high which in turn requires that the size of the file to be transmitted must be large. When the decoding process is suspended due to the absence of a degree one packet, we need more received packets

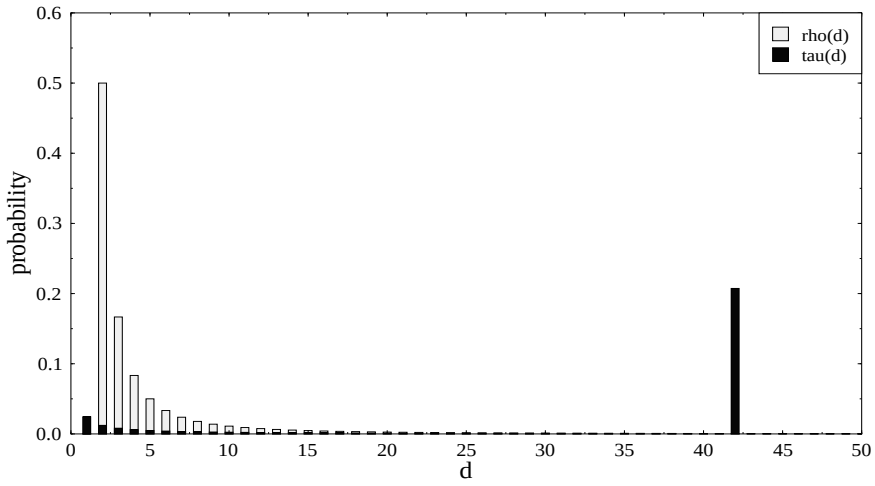


Figure 3.1: The distributions $\rho(d)$ of (3.1), $\tau(d)$ of (3.2) for the case of K=10,000, c=0.2, δ=0.05 [10].

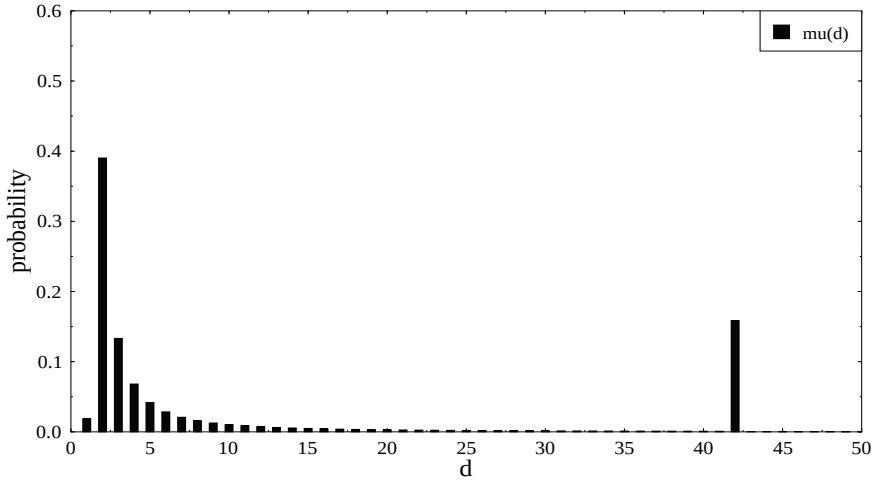


Figure 3.2: The robust distributions $\mu(d)$ of (3.3) for the case of K=10,000, c=0.2, δ=0.05.

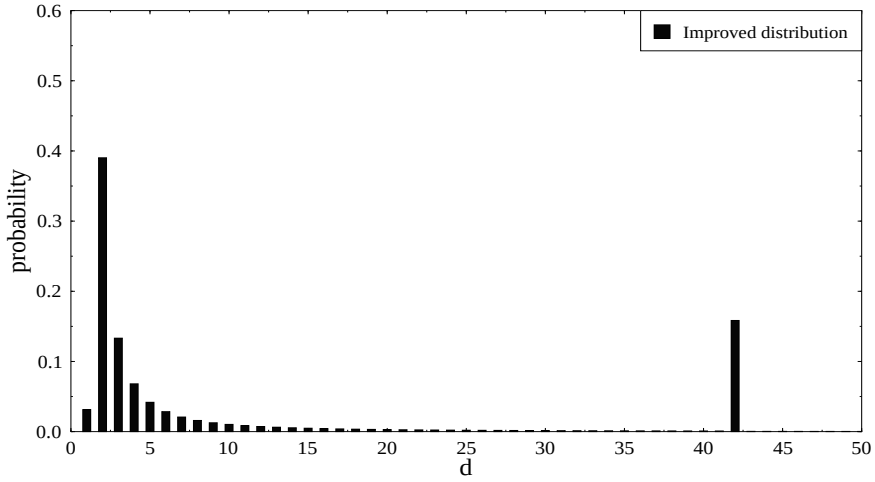


Figure 3.3: The improved robust distributions for the case of K=10,000, c=0.2, δ=0.05.

for recovering all source packets. In view of this, let us consider the case, when we have $(P_{d_i} \cdot K' < 1)$ in Luby's *robust soliton distribution*. Our novel proposition is to improve the distribution's behaviour by introducing an extra factor of:

$$\nu = \sum_{d=2}^{K} \mu(d) \cdot K'; \tag{3.4}$$

for the sake of creating a more beneficial degree distribution, where d_i represents the degree-i term of the distribution, satisfying the following conditions

$$\begin{cases} (\frac{\frac{1}{d(d-1)} + \frac{S}{K} \cdot \frac{1}{d}}{Z}) \cdot K' < 1 & \text{for } 2 \leq d \leq \frac{K}{S} - 1, \\ \frac{1}{d(d-1)} \cdot \frac{K'}{Z} < 1 & \text{for } (\frac{K}{S} + 1) \leq d \leq K. \end{cases} \tag{3.5}$$

Finally, the Improved Robust Soliton Degree Distribution (IRSDD) is defined as follows.

$$D(d) = \begin{cases} 0 & d = 0, \\ \frac{1+S+\nu}{Z \cdot K} & \text{if } d = 1, \\ \frac{1}{Z}\left[\frac{1}{d(d-1)} + \frac{S}{K} \cdot \frac{1}{d}\right] & \text{if } 2 \leq d < \frac{K}{S}, \\ \frac{S}{Z \cdot K}\left[\log\frac{S}{\delta} + \frac{1}{(\frac{K}{S}-1)}\right] & \text{if } d = \frac{K}{S}, \\ \frac{1}{Z}\left[\frac{1}{d \cdot (d-1)}\right] & \text{for } K > d > \frac{K}{S} \end{cases} \tag{3.6}$$

We determine the parameter ν with the aid of the set of inequalities given by (3.5), so that we maximize the relative frequency of having packets of degree-one, because this

allows us to recover the source packets from a single received packet. In other words, the degree-one packets' reception probability was ignored by the original *robust soliton distribution* [10] [39] [43], which was now taken into account by our improved robust distribution. The measure of increasing the relative frequency of degree-one packets has however not only benefits, but also disadvantages. The main benefit is that a degree-one packet is self-contained and hence may be decoded without reference to any other received packets. By contrast, the disadvantage is that it contains less information about other received packets in comparison to higher-degree packets. Hence it may neither be used to recover other packets nor be recovered from other packets. Fig. 3.4 and Fig. 3.5 characterize the improved robust distribution. When using Luby's *robust soliton distribution* of [39] and the LT coding scheme characterized in Fig. 50.4 of [39], approximately 12,000 packets were required for recovering 10,000 packets with the probability of at least $(1 - \delta)$ [39]. By contrast, the number of packets required was reduced to about 10,800, when using the improved robust distribution. Having characterized the proposed degree distribution, let us now focus our attention on the advocated system architecture.

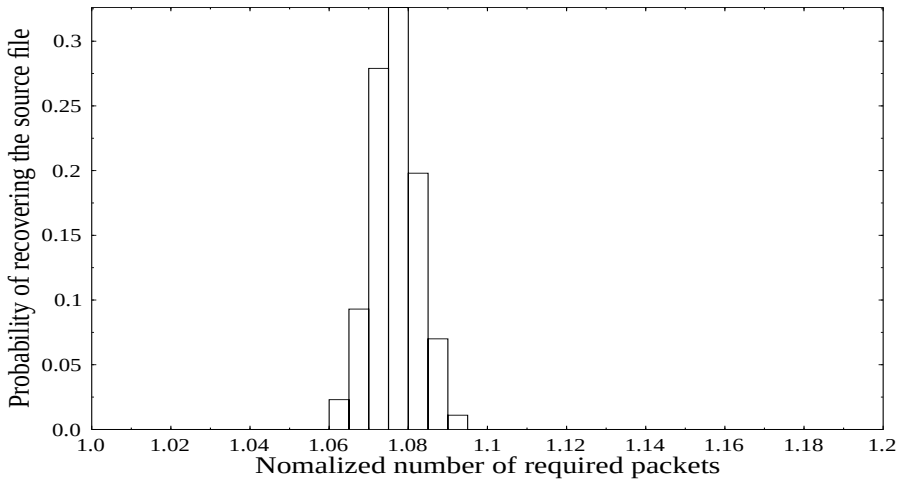


Figure 3.4: IRSDD based LT code specified in Fig 50.4 of [39] for c=0.1 and δ=0.5. The actual number of packets N required to recover the original source file of size K=10,000 packets is given by the product of K and the abscissa value.

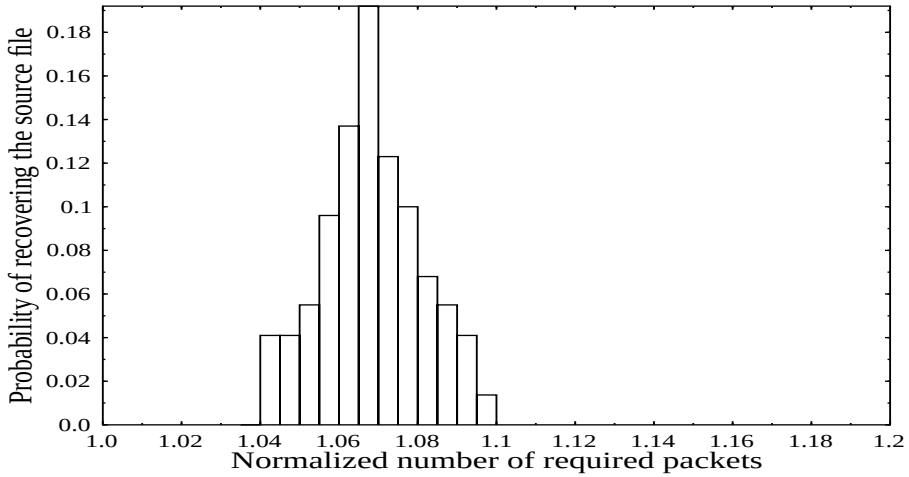


Figure 3.5: IRSDD based LT code specified in Fig 50.4 of [39] for c=0.03 and δ=0.5. The actual number of packets N required to recover the original source file of size K=10,000 packets is given by the product of K and the abscissa value.

3.1.2 Serially Concatenated LT Coding and BICM-ID for the Wireless Internet

In Bit Interleaved Coded Modulation (BICM) the coding and modulation schemes were jointly optimized for the sake of attaining the best possible performance when communicating over fading wireless communication channels. The iterative decoding scheme of BICM (BICM-ID) invoking an appropriate bit-to-symbol mapping strategy enhances its achievable performance in both AWGN and Rayleigh channels. BICM-ID may be conveniently combined with Luby Transform (LT) codes, which were designed for handling packetized wireless Internet data traffic in erasure channels without retransmitting the corrupted packets. By jointly designing a serially concatenated LT-BICM-ID code, an infinitesimally low Bit Error Ratio (BER) is achieved for E_b/N_0 in excess of 7.5dB over wireless Internet type erasure channels contaminated by AWGN. In both wireless and wired Internet based data communication, each data file is transmitted in terms of small packets. In the Medium Access Control (MAC) layer, an ACKnowledgment (ACK) mechanism is used for requesting the source to retransmit lost or corrupted packets. Under hostile channel conditions, the retransmission overhead may become excessive, especially

when using header-compression for reducing the extra overhead imposed by the Internet Protocol (IP). After the reception of an error-infested IP header, an uncompressed header has to be transmitted. Therefore in hostile wireless channel conditions the employment of the IP header compression and retransmission mechanism may substantially increase the transmission overhead and hence may in fact reduce the effective throughput.

For the sake of addressing these wireless Internet design problems, in this treatise we propose a rather different solution. The Internet packet may consist of a number of symbols, where each symbol contains a certain number of bits. These individual bits would be transmitted in terms of symbols over the wireless channel after the modulation. An error correcting code is employed for the sake of recovering the original symbols. However, to avoid the employment of a retransmission mechanism, in addition to using redundant data symbols for FEC coding, the source may transmit a limited number of redundant data packets, which helps to recover the original data file, even if the FEC code failed to remove all errors in some of the packets. Our design objective is then to minimize the number of redundant packets required for the complete recovery of the original data file, rather than aiming for the highest possible coding rate, which is the design objective of classic FEC codes. The associated design trade-off is that the number of corrupted packets can be reduced by reducing the code-rate of the FEC scheme, which clearly imposes a higher burden on the packet-recovery scheme. Our ultimate goal is hence to minimize the total redundancy imposed by both FEC and LT coding, while maximizing the overall integrity. Serially concatenated and iteratively detected coded modulation schemes are capable of attaining a near-capacity performance and hence in the proposed system Bit Interleaved Coded Modulation using Iterative Decoding (BICM-ID) is combined with a LT code [10] [36] [39] .

Again, LT codes were proposed by Luby [10] based on the philosophy of the so-called Fountain codes [36] which were first proposed for the sake of reliably transmitting data files over the Ethernet-based Internet. The files are assumed to be constituted by K information packets, each containing n bits. For the sake of reliably recovering the original data file, (K'=K+E) encoded data packets have to be transmitted, where E is the number of redundant packets that has to be as low as possible.

BICM was proposed by Zehavi [44] for transmission over wireless fading channels. It invokes both a non-systematic convolutional code as well as independent interleavers for each of its bit. The bit interleavers assist in minimizing the effects of bursty errors as

well as maximizing the attainable time-diversity order. Combining BICM with iterative decoding [45] leads to an improved performance for BICM-ID over both AWGN as well as Rayleigh channels. For more detail about the architecture of BICM-ID please refer to Appendix A. When using n original uncoded information bits per symbol, the number of coded BICM-ID output bits becomes ($m = n + 1$). Assuming that the packets contain P number of symbols and that no tail bits are used by the BICM-ID scheme, we have a total of $P.n$ number of bits in the LT source packets and the total rate of the LT-BICM-ID code becomes (r=$Kn/K'm$). Fig. 3.6 illustrates the structure of the source packets as well as the encoded packets employed in the LT-BICM-ID scheme. Fig. 3.7 briefly out-

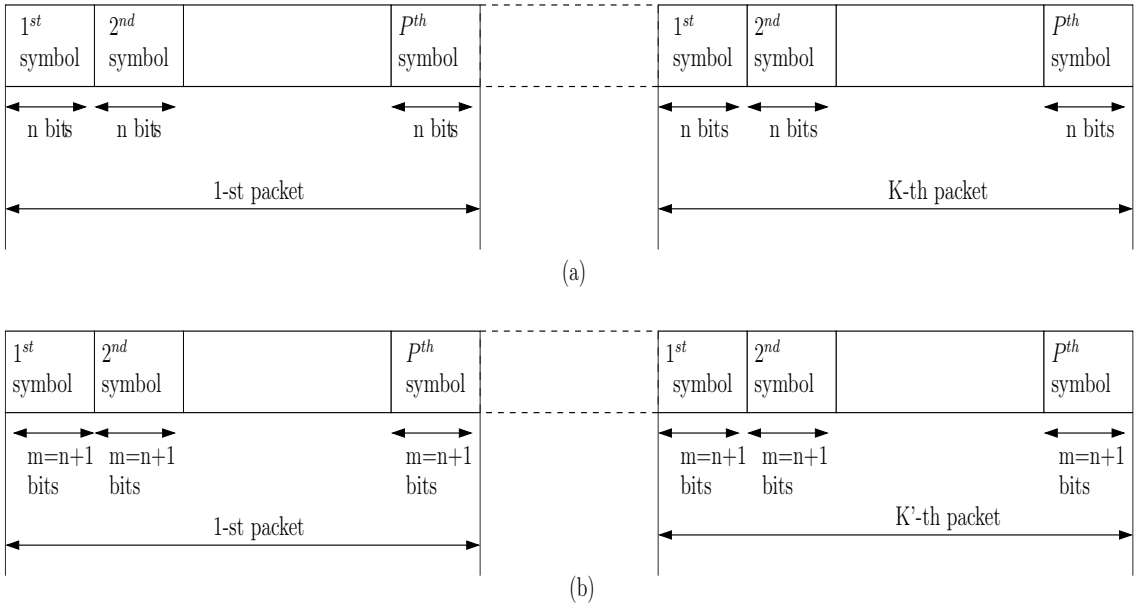


Figure 3.6: (a) Structure of the source packets before LT coding. (b) Structure of the BICM-ID encoded packets.

lines the design of our system, where the LT scheme is the outer code and the BICM-ID arrangement is the inner code. Our investigations are based on ($P.n$=165) bits per original source packet, which is similar to the 164-bit packet size of a transport multimedia service packet in the Time DiVision Duplex (TDVD) mode of the Third-Generation (3G) wireless system [46]. The success or failure of a packet is decided on the basis of Cyclic Redundancy Check (CRC) detection [46]. We transmitted a total of (K=10^4) source packets corresponding to 1,650,000 bits. The number of 'parity' packets was 3,000, rendering the total number of received packets (N=13,000). The BICM-ID encoder's output has ($m = n + 1$=3+1=4) encoded bits per symbol and therefore each packet consists of

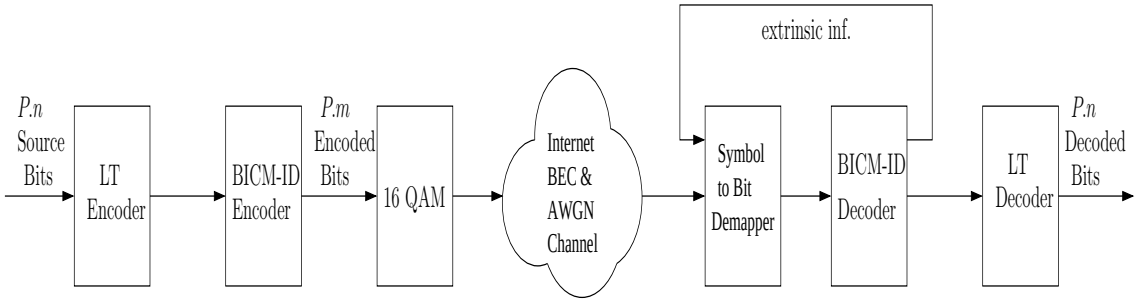


Figure 3.7: System structure of the proposed LT-BICM-ID scheme. The corresponding transmission frame structure is seen in Fig. 3.6 and the associated system parameters are summarized in Table 3.1.

(P=165/3=55) 16QAM symbols. Hence the size of the BICM-ID encoded packet transmitted over the wireless channel is constituted by (165x4/3 = 220) bits/packet. Each symbol is modulated using 16QAM and transmitted over an AWGN channel, where random data bits are assumed to be erased by the Internet modelled by the BEC. The received packet is demodulated and decoded at the receiver. The effective code rate of our system is [r=$K.n/K'm$=(10,000x3)/(13,000x4)=30,000/52,000=15/26]. The erasure channel imposes a packet erasure probability of (P_e=0.1). The family of LT codes has been originally designed for channels prone to packet loss, such as the Internet, where each packet can be readily identified by its packet ID. These channels are often referred to as 'erasure' channels. However, in order to correctly identify an erased packet, we have to make the assumption that the LT encoder and the LT decoder are perfectly synchronized and hence are capable of identifying the lost packet's ID. At the demodulator, the depacketized bits

RS codes	C(7,3,2)
LT codes	C(10,000, 13,000)
LT-BICM-ID codes	C(30,000, 52,000)
BEC erasure probability	0.1
Modulation	16QAM
Number of source packets K	10,000
Number of transmitted packets K'	13,000
Number of bits per source packets $P.n$	165

Table 3.1: System parameters of the concatenated LT-BICM-ID scenario.

are converted to soft bit values that are fed into the BICM-ID MAP decoder [30], which feeds the output extrinsic information constituting the *a priori* information to the input of the symbol-to-bit demapper seen at the input of the BICM-ID scheme in Fig. 3.7. We used Set Partitioning (SP) based mapping [45] for the sake of achieving an iteration gain as the benefit accruing from having a maximized Euclidean distance amongst the BICM constellation points. The BICM-ID-decoded hard-decision bits are then used as the input bits of the LT decoder in Fig. 3.7. Since a message-passing LT decoding algorithm is used [36], an excess number of errors found in the received packets may inflict error propagation and subsequently might degrade the reliability of the neighbouring check nodes, potentially imposing a low LT decoding performance. The output of the BICM-ID decoder ensures that the overwhelming majority of the erroneous bits has been corrected, hence reducing the probability of error propagation.

3.1.3 Simulation Results

These results were extracted from Fig. 3.8 and 3.9. Reed-Solomon (RS) codes [30] have been widely used for transmission over erasure-type channels before the introduction of Fountain codes [36]. Fig. 3.8 shows the BER performance of the RS (7,3,2) code, the LT (13,000, 10,000) code and the proposed LT-BICM-ID (30,000, 52,000) code, when communicating over the BEC contaminated by AWGN, as shown in Fig. 3.7. Again, the source file was constituted by (K=10,000) packets and a total of (K'=13,000) packets were transmitted. As it transpires from Fig. 3.8, the stand-alone RS or LT code does not perform well, when communicating over the BEC contaminated by AWGN. The BEC employed has an erasure probability of 0.1. The LT-BICM-ID code also has a high BER for E_b/N_0 values below 7.4dB, which however significantly improves above this point, as seen in Fig. 3.8. This indicates that for $E_b/N_0 > 7.4$dB, the BICM-ID decoder becomes capable of correcting most of the bits in the P=55-symbol packets and the resultant number of corrupted bits becomes sufficiently low for the LT decoder to correct them.

Fig. 3.8 also portrays the BER performance of the LT-BICM-ID scheme in conjunction with different number of inner BICM-ID iterations. The 2-iteration aided LT-BICM-ID code having eight states is only capable of achieving an infinitesimally low BER performance for E_b/N_0 values above 8.8dB, which is almost 1.5 dB higher than the corresponding threshold E_b/N_0 of the 10-iteration based inner code. This illustrates that as the inner code's error correcting capability improves, a higher fraction of error-free bits is passed to

the LT decoder, hence enhancing the overall BER performance of the system.

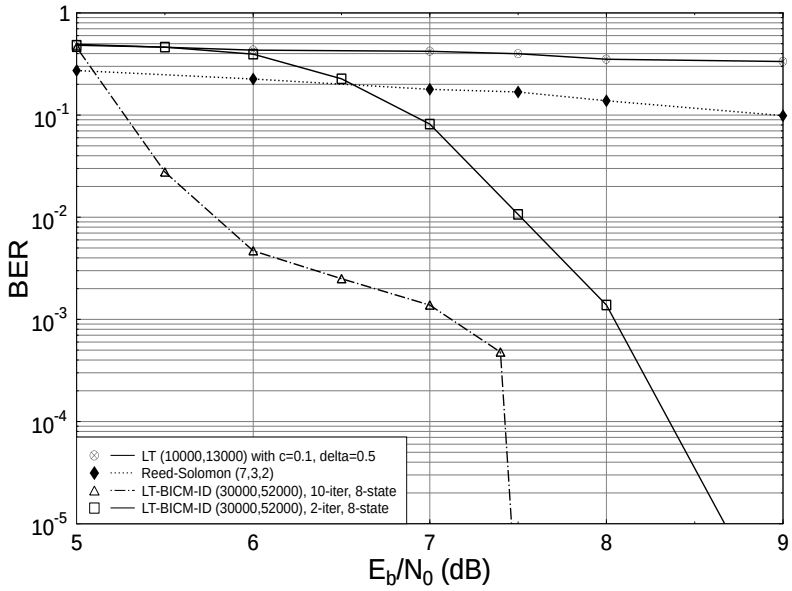


Figure 3.8: BER versus E_b/N_0 performance of the rate 15/26, 10/13 and 3/7 of LT-BICM-ID, LT and RS codes communicating over the BEC having $P_e = 0.1$ contaminated by AWGN, when using 16QAM. The BICM-ID scheme dispensing with LT coding was not included here, because at $P_e = 0.1$ it would have a high error floor.

Fig. 3.9 further characterizes a range of various LT-BICM-ID codes communicating over the BEC contaminated by AWGN and having different degree distributions [10]. For the general Poisson distribution, the degree of transmitted packets is defined by a function given in [10] as:

$$\Omega_D(x) = \frac{1}{\mu+1}\left(\mu.x + \frac{x^2}{1.2} + \frac{x^3}{2.3} + ... + \frac{x^D}{(D-1).D} + \frac{x^{D+1}}{D}\right), \tag{3.7}$$

where D=$[\frac{4(1+\epsilon)}{\epsilon}]$ and μ=$(\epsilon/2) + (\epsilon/2)^2$ and ϵ is defined as a small positive real-valued number.

We further investigated the truncated Poisson 1 (Ω_1) [11], the truncated Poisson 2 (Ω_2) [11], and the *robust* degree distributions, where the Poisson degree distributions [11] were used for the sake of reducing the number of corrupted bits inflicted by the AWGN channel at low E_b/N_0 values. Here, we employed the truncated Poisson distributions, which were

defined as follows [11] :

$$\begin{aligned}\Omega_1(d) &= 0.007969d^1 + 0.5161d^2 + 0.166220d^3 + 0.072646d^4 + 0.082558d^5 + 0.056058d^8 + \\ &\quad 0.037229d^9 + 0.055590d^{19} + 0.025023d^{65} + 0.003135d^{66};\\ \Omega_2(d) &= 0.007544d^1 + 0.5033d^2 + 0.166458d^3 + 0.071243d^4 + 0.084913d^5 + 0.049633d^8 + \\ &\quad 0.043365d^9 + 0.045231d^{19} + 0.010157d^{20} + 0.010479d^{66} + 0.017365d^{67}, \qquad (3.8)\end{aligned}$$

where d^n represents the degree order of each of the above items. The *robust* degree distribution proposed in Section 3.1.1 of this scenario was used. Fig. 3.9 demonstrates that the Poisson degree distribution Ω_1 exhibited the worst BER performance, resulting in a constant BER around $6\text{x}10^{-2}$ for E_b/N_0 values in excess of 7 dB. The BER performance of the Ω_2 degree distribution was slightly better in Fig. 3.9 due to its higher mean degree of five as opposed to the average degree of 4.63 for the Ω_1 distribution. Above the E_b/N_0 of 7dB, all the Poisson distribution based LT-BICM-ID schemes are seen in Fig. 3.9 to maintain a constant BER with no further improvement in their error correction capability. This is due to the lack of connecting *edges* between the LT-coded packets and the source packets in the corresponding *Tanner* graph. In other words, the average packet degree is insufficiently high for recovering, i.e. filling the packets chopped by the erasure channel. The LT-BICM-ID scheme associated with the *robust* distribution of Section 3.1.1, is however seen in Fig. 3.9 to be capable of correcting all the errors, when the E_b/N_0 reaches 7.5dB, since it contains a higher number of *Tanner* graph connections between the LT-coded packets and the source packets, which assists in correcting more erroneous packets. As the decoding process continues, we may encounter a lack of degree *one* connecting edges, thus forcing the decoding process to halt prematurely. This explains the phenomenon observed in Fig. 3.9, namely that the BER performance no longer improves upon increasing the E_b/N_0 for any of the Poisson distribution based LT-BICM-ID schemes above E_b/N_0=7dB.

We further investigate the performance of our proposed *robust* distribution compared to the original *robust* distribution of [36] in conjunction with the same value of c=0.1 and δ=0.5. We observe from Fig. 3.9 that the LT-BICM-ID scheme using the original *robust* distribution of [36] performs better in the range of $E_b/N_0 < 7.5$dB. This is due to the lower number of *edges* or lower mean value of the degree distribution compared to our *robust* scheme of Section 3.1.1. When the mean value of the degree of the distribution is lower, resulting in a reduced number of connecting *edges* in the factor graph, there would be a reduced error propagation between the potentially erroneously decoded source pack-

ets and the LT-coded packets yet to be decoded, but relying on the erroneously decoded source packets. This is particularly so at lower E_b/N_0 values, when the inner BICM-ID code is less capable of correcting the bits corrupted by the AWGN. This means that, if a high number of received packets are corrupted, the corrupted packets are less likely to propagate their erroneous bits due to the lower number of connecting *edges* between the source packets and LT-encoded packets. However, at higher E_b/N_0 values, the proposed *robust* distribution of Section 3.1.1 outperforms the schemes using the original *robust* distributions due to their higher average degree. When the inner BICM-ID code is capable of correcting those bit errors, the higher mean value of the degree of the *robust* degree distribution scheme assists in recovering the corrupted packets, as a benefit of having an increased number of connecting *edges* in the factor graph. This shows that the proposed *robust* distributed LT outer codes of Section 3.1.1 exhibit a better BER performance at higher E_b/N_0 values, as shown in Fig. 3.9, while requiring a reduced number of extra LT packets for the sake of recovering the original source packets, as shown earlier in Fig. 3.4 and Fig. 3.5.

In order to circumvent these error-propagation phenomena, a sophisticated serial concate-

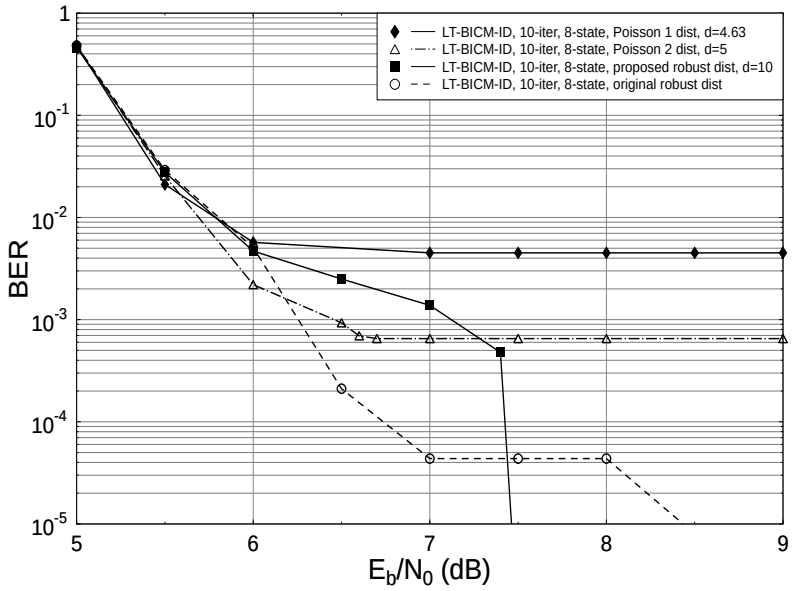


Figure 3.9: BER versus E_b/N_0 performance of various LT-BICM-ID codes communicating over the BEC contaminated by AWGN using various Poisson and robust distributions employing 16QAM.

nated LT-BICM-ID scheme was proposed, which is capable of improving the achievable BER performance of packetized data transmission, when communicating over the AWGN-

contaminated BEC. The concept of using BICM-ID to improve the attainable BER of the LT codes invoked for correcting the packet errors was the main benefit of the proposed scheme. With the advent of using our proposed improved *robust* distribution of Section 3.1.1 and a strong BICM-ID inner code, we demonstrated that our scheme is capable of achieving a better BER performance than that of the classic RS (7,3,2) and LT (10,000, 13,000) benchmarkers, when communicating over the BEC contaminated by AWGN. This scheme is particularly attractive in a wireless Internet scenario. We elaborated on the joint optimization of the inner and outer code-rates for the sake of maximizing both the effective throughput and the attainable integrity of the system. Our future research will consider the employment of both binary and non-binary LDPC codes as well as TTCM schemes in dispersive turbo-equalized channels [30]. For the sake of improving the attainable performance of LT coded systems communicating over Wireless Internet Channels (WIC) iteration detection exchanging extrinsic information between the LT codes and various other block codes will be invoked in the following section.

3.2 Generalized-LDPC and LT Codes for the Wireless Internet

Again, LT codes were originally designed for the Binary Erasure Channel (BEC) encountered owing to randomly dropped packets in the statistical multiplexing aided Internet, where transmitted packets are not affected by the fading or noise of the environment. For the sake of transmitting data over the BEC routinely encountered in statistical multiplexing aided wireless Internet-style scenarios, LT codes have to be combined with classic channel codes.

In this section we introduce a novel almagamated LT coding and Generalized-LDPC (G-LDPC) coding scheme. We will demonstrate that upon exploiting the packet erasure information provided by the G-LDPC decoder for the LT decoder, the proposed scheme achieves a low BER for E_b/N_0 values in excess of 3.2dB, when transmitting data over the AWGN-BEC channel having a packet erasure probability of P_e=0.1. We will also show that for communicating over the uncorrelated Rayleigh-BEC channel at P_e=0.1, an E_b/N_0 value of 5.3 dB was required. We also demonstrate in Section 3.2.2 that in comparison to the serially concatenated LT-G-LDPC scheme using no information exchange between the G-LDPC decoder and the LT decoder, the amalgamated LT-G-LDPC ar-

rangement achieved an E_b/N_0 gain of up to 1.6-2.5 dB. G-LDPC codes [47] [48] offer the beneficial design option of replacing the simple parity-check codes of classic LDPC codes by more sophisticated constituent encoders, such as for example binary or non-binary Bose-Chaudhuri-Hocquenghem (BCH) codes [30]. They also benefit from innovatively combining low-complexity constituent encoders and decoders with intelligent iterative detection in the interest of approaching the attainable capacity at a moderate complexity. In this treatise G-LDPC codes are amalgamated with LT codes [10], which were originally designed for achieving an infinitesimally low PER over the BEC channel routinely encountered in statistical multiplexing aided wireless Internet-style scenarios. However, LT codes were not intended for communications over gravely error-infested hostile channels encountered in the wireless Internet.
Hence, the novel contribution of this section is that:

- *We amalgamated LT codes with G-LDPC codes, where the latter combats the effects of channel errors and informs the LT decoder, as to when it is unsafe to carry out LT decoding without catastrophic error propagation across the LT-encoded packets.*
- *We will demonstrate that as a result, E_b/N_0 gains of 1.6 - 2.5 dB are achievable in the Rayleigh- faded wireless Internet environment considered.*

In fact, owing to their potentially avalanche-like decoding error propagation across LT-encoded packets, the employment of LT codes may catastrophically degrade the overall system performance, unless they are combined with powerful Forward Error Connection (FEC) codes. As a benchmarker, we used the potent combination of an iteratively detected BICM-ID and LT coding scheme [20]. However, the operation of the BICM-ID and LT schemes as portrayed in [20] was based on simply passing the channel-decoded hard-decision bits to the LT decoder, rather than on intrinsically amalgamating their operation. Any potential residual errors persisting at the output of the BICM-ID decoder were passed to the LT decoder, which may have precipitated the errors by propagating them to further LT-encoded packets. Hence, in this treatise LT codes are intrinsically amalgamated with G-LDPC codes [47] [48] [49]. More explicitly, before LT decoding ensues, each G-LDPC codeword is tested by the classic parity check of ($\mathbf{X} \cdot \mathbf{H}^T = 0$), to ensure that a legitimate although not necessarily error-free codeword was generated, where $\mathbf{X}$ represents a legitimate G-LDPC codeword, while $\mathbf{H}^T$ is the transpose of the parity check matrix. This measure prevents the LT decoder from propagating G-LDPC decoding errors from one LT-encoded

packet to another, unless the G-LDPC decoder failed to recognize that a legitimate, but erroneously decoded codeword was encountered.
In Section 3.2.1 we describe the proposed G-LDPC-LT scheme, while in Section 3.2.2 we characterise its achievable performance.

3.2.1 Amalgamated LT and G-LDPC Coded Scenario

The schematic of the proposed coding arrangement is shown in Fig. 3.10, while the mapping of the G-LDPC codewords to an LT-encoded transmitted packet is seen in Fig. 3.12. To elaborate a little further, the philosophy of G-LDPC codes is reminiscent of the classic turbo encoding principle of using low-complexity constituent encoders and decoders for creating powerful, near-capacity iterative decoding schemes having a moderate complexity. More detail about the G-LDPC structure can be seen in Appendix B. The liaison of the G-LDPC decoder with the LT decoder is based on counting the number of illegitimate G-LDPC codewords in an LT-encoded packet and then comparing ϵ to an appropriately chosen threshold T. To elaborate a little further, the decoding operations commence by setting the threshold T to zero, indicating that only error-free G-LDPC packets are allowed to be passed to the LT decoder, in order to avoid LT-decoding-induced inter-packet error propagation. If we have $\epsilon = 0$, no erroneous G-LDPC codewords were found within the current LT-encoded packet and hence this packet is ready for LT decoding. Furthermore, if this error free LT-packet is a degree-one packet, LT decoding may commence immediately. By contrast, if no degree-one packet is available at the LT decoder, the commencement of the LT-decoding process is deferred until the arrival of a degree-one packet. For the sake of avoiding the risk of losing all the remaining undecoded source packets, in the absence of a degree-one packet, we may decide to increase the threshold T, in order to request further LT-encoded packets, which now contains $\epsilon > 0$ number of erroneous G-LDPC codewords. If this is the case, the residual errors of the current LT-encoded packets are passed on to other packets in the same bit position owing to their modulo-2 relationship. To elaborate a little further, it is determined by the degree of the packet concerned, how many further errors are caused. Therefore erroneous G-LDPC codewords should only be considered for LT decoding, if they belong to degree-one packets, since the absence of an error-free degree-one packet results in losing all the remaining packets. This measure prevents the propagation of G-LDPC decoding errors to other LT packets, since degree-one packets are decoded without the aid of other packets, while avoiding the risk

of losing all the remaining packets owing to the absence of degree-one packets.
By contrast, in case of G-LDPC decoding errors in LT packets having a higher degree the erasure of a packet does not prevent decoding the rest of the packets, but would propagate the errors to as many LT-encoded packets, as the degree of the packet concerned. Hence, higher-degree packets containing G-LDPC decoding errors have to be considered as erased. However, given the ability of the LT decoder to fill the erased codewords, the amalgamated scheme is expected to outperform both its components at a modest complexity. For simplicity, we refer to the system benefiting from information exchange between the LT decoder and the G-LDPC decoder as Scheme 1, while the one dispensing with it is termed as Scheme 2.

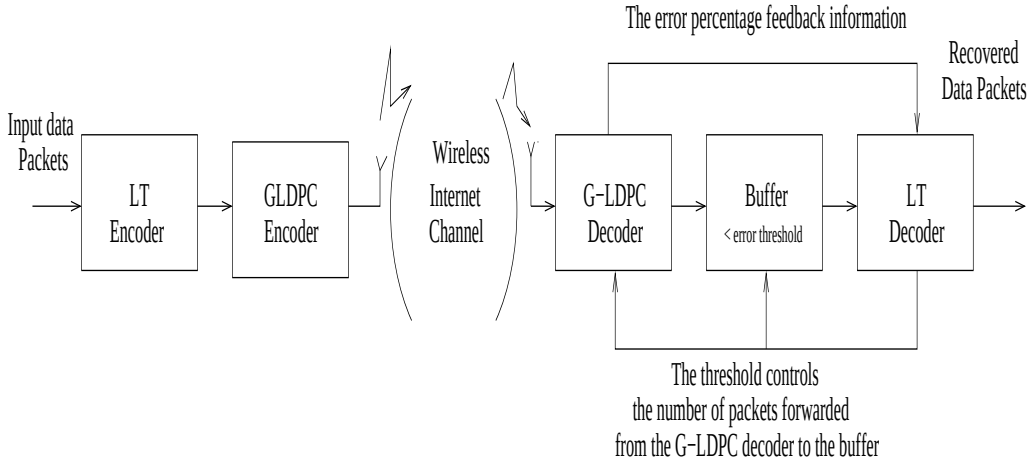


Figure 3.10: Amalgamated G-LDPC codes and LT codes

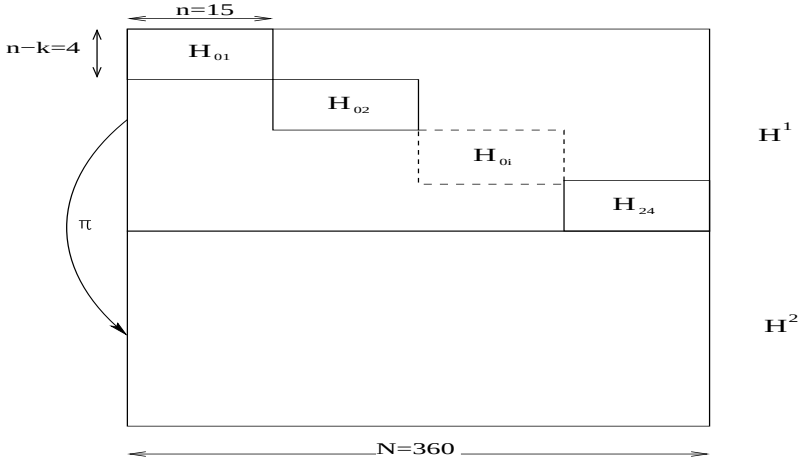


Figure 3.11: The structure of the G-LDPC parity check matrix

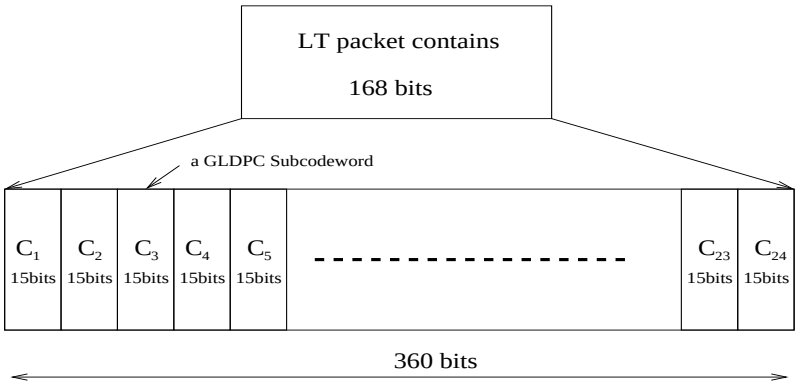


Figure 3.12: The relation between the LT-encoded packet and the G-LDPC codewords

3.2.2 System Parameters and Performance Results

Erasure probability	P_e= 0.1
LT code parameters	δ= 0.5 c= 0.1
Number of bits per LT packet	168
Number of source packets	10,000
Number of encoded LT packets	13,000
G-LDPC component codes	BCH(15,11,1)
Component code rate of G-LDPC	$r = \frac{11}{15}$
Code rate of G-LDPC	$\frac{7}{15}$
Modulation	Binary Phase Shift Keying (BPSK)

Table 3.2: System parameters of the almagamated G-LDPC-LT scheme.

The parameters of the system investigated are summarized in Table 3.2. The G-LDPC encoder used the binary BCH(n,k,t)= BCH(15,11,1) constituent codes, which operated over the Galois Field GF(16) and each BCH codeword was capable of correcting t=1 error, when using classic algebraic decoding [30]. The number of input data bits per LT-encoded packet was 168 and each of these packets was mapped to 24 G-LDPC codewords, as seen in Fig. 3.12. We can calculate the number of G-LDPC subcodewords in an LT-encoded packet as follows. We denote the code rate of the component BCH codes by r, while r is the overall code rate of the G-LDPC code. Observe in Fig. 3.11 that the parity matrix of the G-LDPC encoder has two superblocks, namely $\mathbf{H}_1$ and $\mathbf{H}_2$, where $\mathbf{H}_2$ represents a permuted version of $\mathbf{H}_1$, as detailed in [47] [48] [49]. More specifically, $\mathbf{H}_1$ contains L= 24 parity matrices corresponding to the BCH(15,11,1) codes given that we use J= 2 G-LDPC

superblocks and that the number of parity bits is $[(n-k)= 4]$, the resultant G-LDPC code has a total of $(4 \times 2 = 8)$ parity bits. Hence, the resultant G-LDPC code rate becomes $r= \frac{7}{15}$. More generally, the overall code rate r is given by $[r= 1\text{-}J\cdot(1\text{-}\frac{k}{n})]$ [48], where again, $J= 2$ is the number of the so-called G-LDPC superblocks and $r=\frac{k}{n}=\frac{11}{15}$, therefore we arrive at $r=\frac{7}{15}$. Hence, as seen in Fig. 3.12, the number of G-LDPC subcodewords used by the G-LDPC scheme encoding a single LT source packet equals to $\frac{168}{7}=24$. The LT encoder used the Improved Robust Soliton Degree Distribution (IRSDD) designed in [20] and imposed a 30% packet overhead, while using the LT encoder parameters of $\delta=0.5$ and $c = 0.1$ as discussed in [20]. The number of G-LDPC decoding iterations was set to $I=10$ in the schematic of Fig. 3.13. The BER performance of the system using the parameters of Table 3.2 is shown in Figs. 3.14 and 3.15, when communicating over both AWGN and uncorrelated flat-fading Rayleigh channels, initially inflicting no erasures, ie. assuming that we have $P_e=0$. By contrast, in Fig. 3.16 and Fig. 3.17 we used a somewhat unconventional channel model, which may be encountered in AWGN-contaminated and Rayleigh-faded radio links also subjected to a statistical multiplexing-induced LT-packet erasures with a probability of $P_e=0.1$. Naturally, the packet erasure events may have been imposed by other phenomena, such as shadow-fading-infested G-LDPC packets. Observe in Fig. 3.16 that the amalgamated G-LDPC-LT Scheme 1 exhibits an approximately 1.6 dB E_b/N_0 gain over Scheme 2, when using the Approximate-log MAP decoder [30]. Similar performance trends were observed in Fig. 3.17 for transmission over uncorrelated Rayleigh channels, exhibiting an approximately 1.5dB gain for Scheme 1 employing the proposed threshold-based information exchange between the G-LDPC and LT decoders. Fig. 3.18 and Fig. 3.19 characterise the PER performance of the amalgamated G-LDPC-LT Scheme 1 as well as that of Scheme 2, when the data is transmitted over the AWGN-BEC channel at $E_b/N_0= 4$dB employing both decoding methods, namely scheme-benefitting from passing moderately contaminated degree-one packets to the LT decoder from the G-LDPC decoder and scheme erasing them. We observe from Fig. 3.18 and Fig. 3.19 that the performance of Scheme 1 is better than that of Scheme 2. As seen in Fig. 3.18 and Fig. 3.19, Scheme 1 is capable of achieving an infinitesimally low PER for erasure probabilities below at $P_e=$ 0.16, while the latter has a PER of about 10^{-1} even at $P_e= 0.10$.

In summary, in this study an amalgamated G-LDPC-LT coding scheme was proposed, where the constituent G-LDPC and LT decoders exchanged information. In an effort to avoid curtailing the LT decoding operation in the absence of degree-one packets, Scheme 1 allowed the G-LDPC decoder to pass a maximum of T illegitimate G-LDPC codewords to

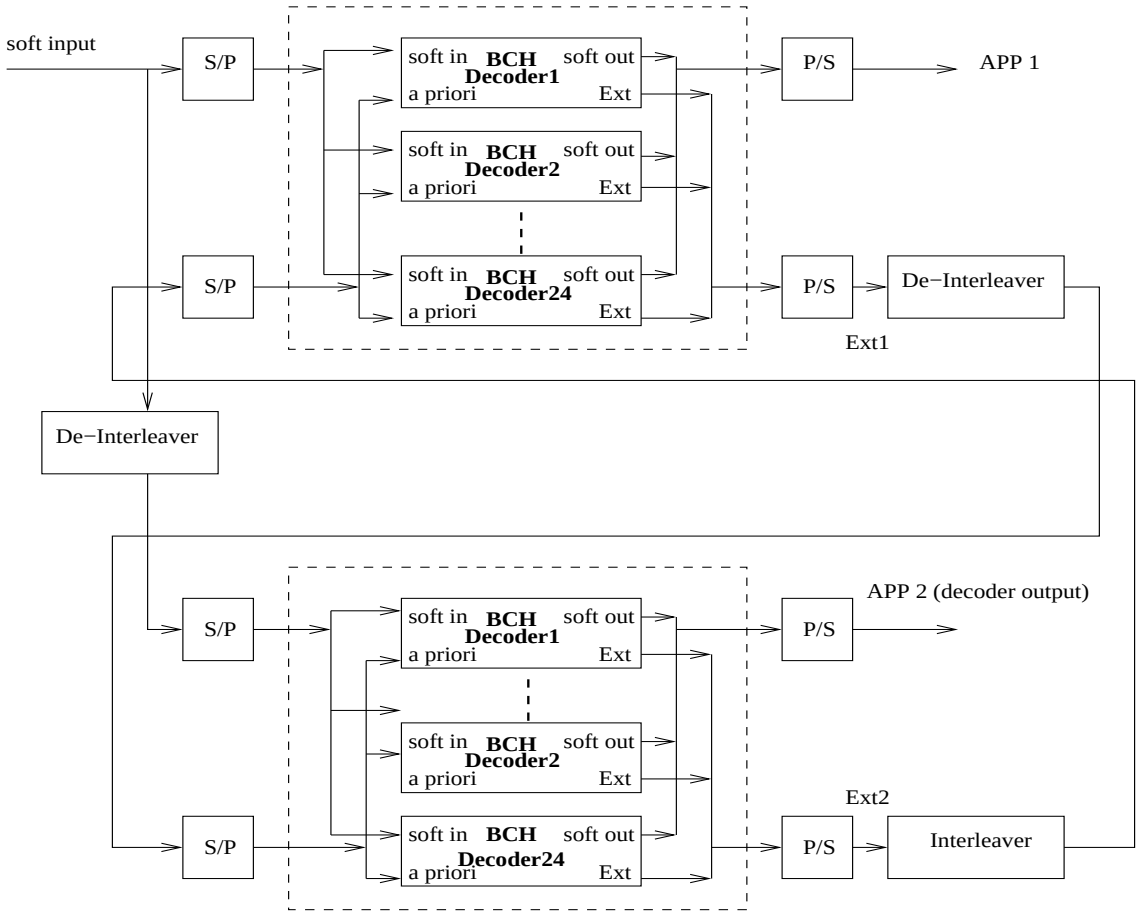


Figure 3.13: The G-LDPC decoder.

the LT decoder. By contrast, Scheme 2 allowed the G-LDPC decoder to pass all G-LDPC codewords to the LT decoder, which causes error propagation during the LT decoding operation. As a result observed in Fig. 3.16 and Fig. 3.17 that E_b/N_0 gains of 1.6 and 2.5dB were achieved over the BEC-AWGN and BEC-Rayleigh channels considered. This study motivated the investigation of a future improved scenario of combining LT codes and BICM-ID in Section 3.3. More explicitly, a new Random Integer Generator (RIG) design will be proposed for coining the specific degrees of the input LT packets.

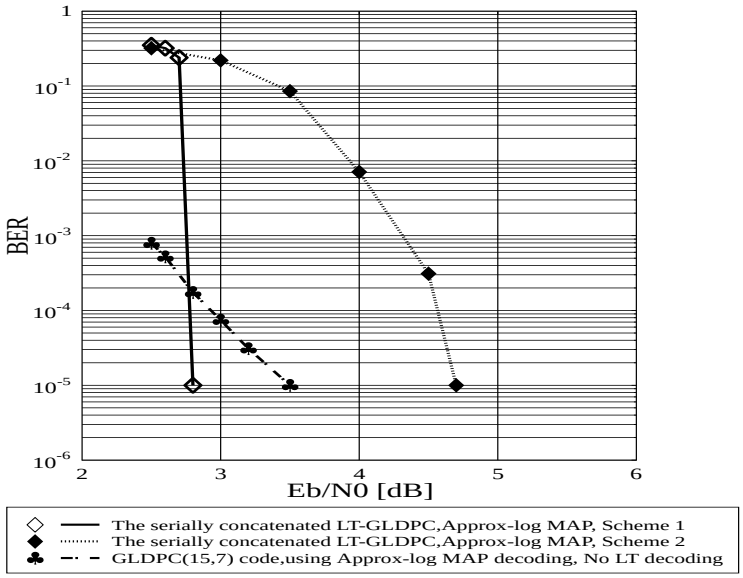


Figure 3.14: BER versus E_b/N_0 performance of the serially concatenated LT and G-LDPC code, using BPSK modulation for transmission over the AWGN channel, at P_e=0.

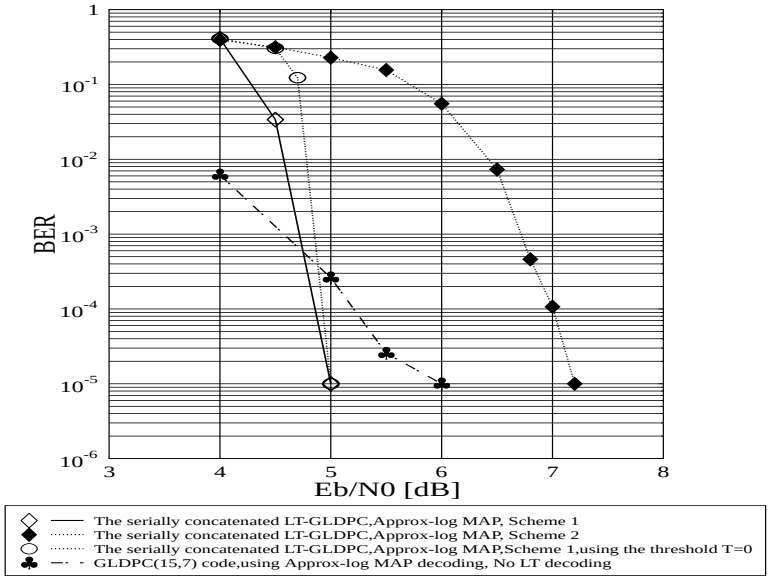


Figure 3.15: BER versus E_b/N_0 performance of the serially concatenated LT and G-LDPC code, using BPSK modulation for transmission over the uncorrelated Rayleigh channel, at P_e=0.

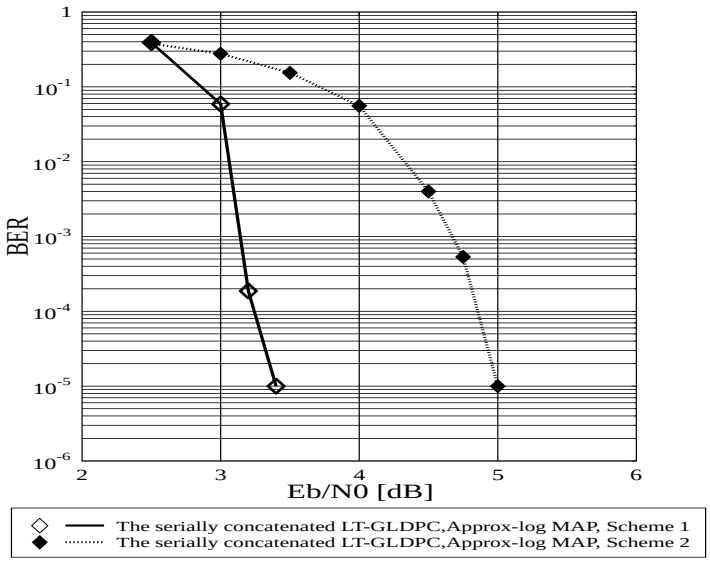


Figure 3.16: BER versus E_b/N_0 performance of the serially concatenated LT and G-LDPC code, using BPSK modulation for transmission over the BEC-AWGN channel, at P_e=0.1.

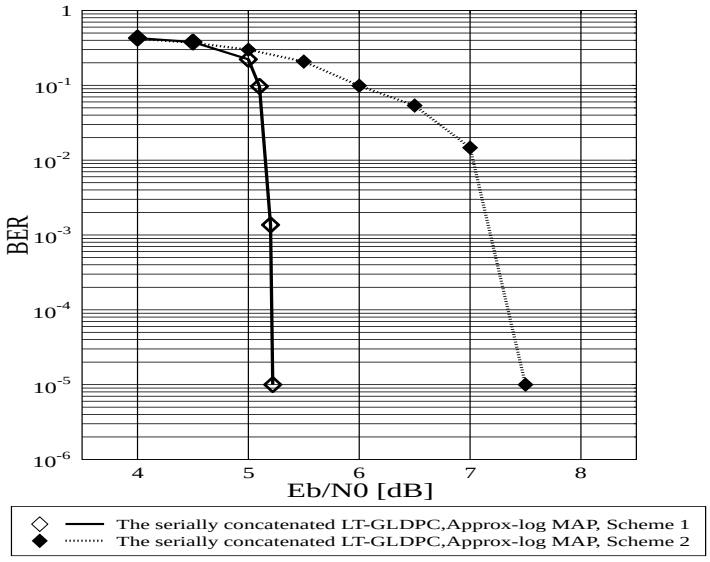


Figure 3.17: BER versus E_b/N_0 performance of the serially concatenated LT and G-LDPC code, using BPSK modulation for transmission over the BEC-Rayleigh channel, at P_e=0.1.

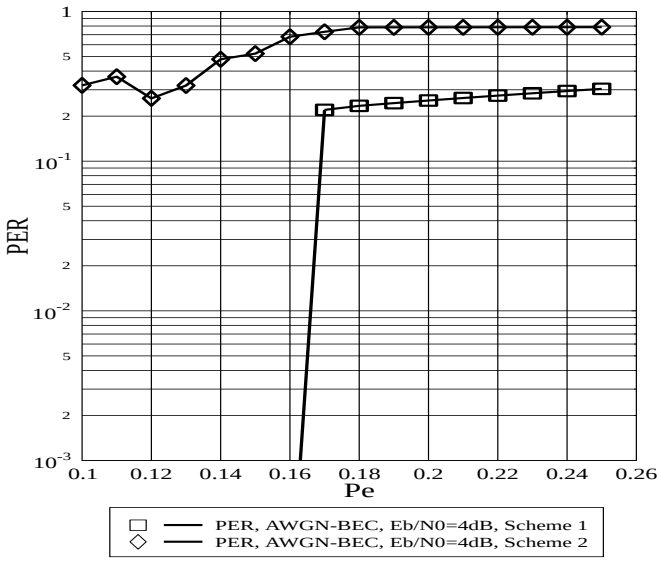


Figure 3.18: PER versus E_b/N_0 performance of the amalgamated G-LDPC-LT Scheme 1 and Scheme 2 for transmission over an AWGN-BEC channel, at E_b/N_0= 4dB.

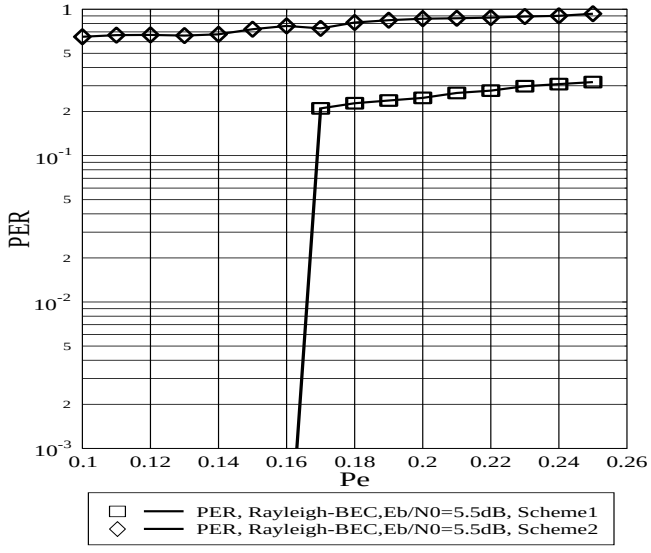


Figure 3.19: PER versus E_b/N_0 performance of the amalgamated G-LDPC-LT Scheme 1 and Scheme 2 for transmission over the uncorrelated Rayleigh-BEC channel, at E_b/N_0= 5.5dB.

3.3 Source LT Packet Degree Distribution Design and Hard Bit-by-Bit Decoding

In this section Bit-Interleaved Coded Modulation using Iterative Decoding (BICM-ID) is amalgamated with LT coding. The resultant joint design of the physical and data link layer substantially improves the attainable BER performance. A Cyclic Redundancy Check (CRC) combined with a novel Log-Likelihood Ratio (LLR) based packet reliability estimation method is proposed for the sake of detecting and disposing of erroneous packets. Subsequently, bit-by-bit LT decoding is proposed, which facilitates a further BER improvement at a reduced number of BICM-ID iterations. Finally, we revisit the pseudo random generator function used for designing the LT code's generator matrix. The classic data link layer technique of mitigating the wireless channel impairments is to employ Automatic Repeat Requests (ARQ) [50] [51]. Naturally, this method requires additional bandwidth for the retransmission of data, once a corrupted packet is received which necessitates a feedback channel. By contrast, in this study we dispense with the feedback channel by jointly designing the physical and data link layer. More explicitly, LT [38] coding is employed for eliminating the feedback required by ARQ aided transmissions. LT coding was originally designed for the BEC, which requires the transmission of redundant packets for the sake of recovering the original source packets. We propose to amalgamate LT coding with Bit-Interleaved Coded Modulation using Iterative Decoding (BICM-ID) [44] for the sake of mitigating the adverse effects of wireless communication channels. The novelty of this section is that instead of using a simple serial concatenated LT and BICM-ID scheme, we intrinsically amalgamate LT coding and BICM-ID by creating an exchange of extrinsic LLR information between them. Again, a CRC scheme is used for detecting the presence of any erroneous LT packets. A further novel improvement suggested is that an LLR based packet reliability estimation is introduced. Extrinsic Information Transfer (EXIT) charts are used by the proposed LLR-based packet reliability estimation scheme. This potentially enhances the error checking capability, especially at higher E_b/N_0 values and hence may eliminate the potential redundancy introduced by the CRC overhead. Error propagation across LT packets encountered during packet decoding may cause failure in LT decoding. Hence, as a counter measure, in this section we introduce a bit-by-bit LT decoding technique by exploiting the LLR estimates provided by the associated BICM-ID decoder. The advantages are two folds:

1) We reduce the probability of occurrence of erroneous LT packets, which would otherwise have to be discarded.

2) We mitigate the probability of potential error propagation inflicted by erroneous LT packets.

When corrupted bits are in the same index locations of different source packets, we can readily discard them with the aid of bit-by-bit LT decoding, as long as we have a efficiently high number of received packets providing the minimum number of redundant packets. Furthermore, we propose a novel pseudo-random generator [52] for determining the specific degree of the individual LT-encoded packets, which will allow us to reduce the proportion of redundant packet required for successful LT decoding. The rest of this section is organized as follows. Section 3.3.1 provides an overview of our system, outlining the joint design of our amalgamated LT and BICM-ID coding scheme. The proposed CRC and LLR-based packet reliability estimation scheme designed with the aid of EXIT charts are detailed in Section 3.3.2. Section 3.3.3 describes the bit-by-bit LT decoding as well as our pseudo-random LT generator matrix based approach, while Section 3.3.5 quantifies the achievable performance of this novel jointly designed scheme.

3.3.1 System Overview

The schematic of the joint LT BICM-ID arrangement is shown in Fig. 3.20. Consider the single-transmit and single-receive antenna aided system seen in Fig. 3.20, when the source data file to be transmitted consists of K number of packets. The LT decoder will require the transmission of $K' = (K + E)$ number of LT-encoded packets as seen in requiring E redundant packets for the success of the decoding process. After transmission over the BEC imposing an erasure probability of P_e, $(P_e.K')$ number of the original LT-encoded packets will be obliterated. At the physical layer, we deal with frame-by-fame transmission. Each transmission frame consists of a number of packets. A transmission packet is in turn broken into S number of symbols. Each BICM-ID source symbol is comprised of k source bits and the BICM-ID encoded symbols have $n = (k + 1)$ encoded bits. The encoded bits are mapped to an M-QAM constellation before transmission over an AWGN channel. At the receiver seen in Fig. 3.20, the BICM-ID decoder receives *a priori* information from the demodulator and outputs extrinsic information, which is to be fed back to the demodulator for iterative decoding. The sufficiently reliable output bits are then passed to

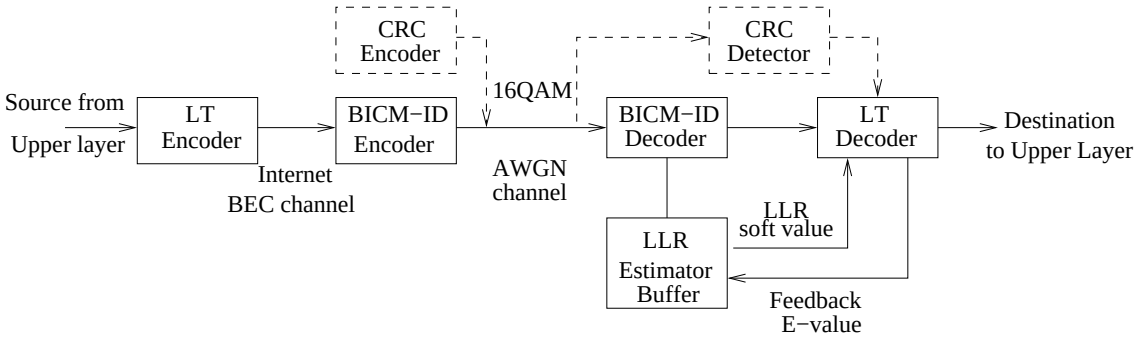


Figure 3.20: System structure of the joint design of LT and BICM-ID.

the LT decoder after a number of BICM-ID iterations. We assume here that the perfectly sychronized LT encoder and LT decoder are capable of reliably identifying the indices of the lost packets. The CRC encoder inserts CRC bits into the LT-encoded packets, while the CRC detector extracts the CRC bits and verifies the detectable errors within each packet.

3.3.2 Packet Reliability Estimation Scheme

The LT decoder using the Belief Propagation (BP) algorithm [38] is inherently sensitive to error propagation. An erroneous packet, which contains corrupted bits may propagate the errors to other LT decoded packets, when modulo-2 operations are performed during the message passing decoding [20]. By ensuring that a minimum of E redundant packets is available, we can improve the achievable decoding performance by reducing the associated error propagation. This can be achieved by discarding erroneous packets, as and when corrupted bits are detected in a packet. Subsequently we employ the CRC-12 code, which is widely used in telecommunication systems for error detection in conjunction with the generator polynomial $(x^{12} + x^{11} + x^3 + x^2 + x + 1)$ [8]. This error detection approach improves the achievable system performance, but naturally, introduces an overhead. Observe in Fig. 3.20 that the CRC encoder injects the CRC bits into the BICM-ID encoded 16-levels Quadrature Amplitude Modulation (QAM) stream, before transmitting them over the AWGN-contaminated BEC. In this section, we map four bits to the 16QAM constellation points. The received CRC bits are extracted and are used for detecting the presence of any erroneous bits.

As an alternative we will demonstrate that the joint design of LLR-based reliability estima-

tion using the BICM-ID decoder has the ability to further improve the system's achievable performance. This potentially allow us to eliminate the CRC overhead bits, while maintaining reliable packet erasure detection, especially at a high E_b/N_0. As seen in Fig. 3.20, the soft LLR output of the BICM-ID decoder is stored in the LLR estimator's buffer. The BICM-ID decoder will set a threshold value for the LLRs, in order to determine, which particular LT packet may contain erroneous bits. Based on their LLR values, each of the corresponding bits is flagged with a '0' or '1' to indicate a reliable or unreliable bit, which allows the LT decoder to mitigate the associated error propagation effects during the decoding process. As seen in Fig. 3.20 a feedback signal is provided by the LT decoder to control the LLR estimator's threshold for the sake of achieving the highest possible detection accuracy. Fig. 3.21 shows the basic BICM-ID scheme, where π and π^{-1} de-

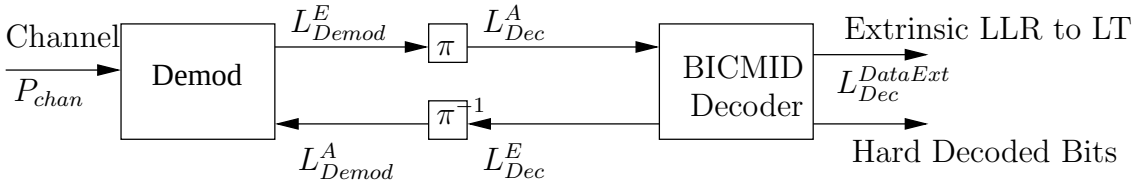


Figure 3.21: BICM-ID system structure.

note the interleaver and deinterleaver. The channel's output information is passed to the demodulator, which receives the *a priori* LLRs L^A_{Demod} from the BICM-ID decoder and provides the extrinsic LLRs L^E_{Demod} for the decoder. The channel decoder of Fig. 3.21 receives the *a priori* LLRs L^A_{Dec} from the demodulator and outputs the extrinsic LLRs L^E_{Dec} in order to perform iterative decoding by exchanging extrinsic information between the BICM-ID decoder and demodulator. The reliability of the demodulator's decisions depends on the channel's Symbol-to-Noise Ratio E_b/N_0 and on the *a priori* information L^A_{Demod}, yielding L^E_{Demod}=$f(L^A_{Demod}|SNR_{chan})$. By contrast, the reliability of the channel decoder's decisions depends solely on the *a priori* LLRs received from the demodulator, since it only has a single input and hence it is not directly dependent on the SNR [53], i.e. on the extrinsic mutual information $I^E_{Dec} = f(I^A_{Dec})$ and $I^{DataExt}_{Dec}$=$g(I^A_{Dec})$. Using the technique proposed in [53] we plotted both the EXIT chart and the corresponding BER values in Fig. 3.22, assuming that the variance σ^2 of the Gaussian distributed LLR outputs of the decoder output is known [53]. The system parameters used in this section are detailed in Table 3.3. More explicitly, Fig. 3.22 details the EXIT characteristics of both the demodulator and of the BICM-ID decoder. The nine dotted lines represent the

BICM-ID code rate	r=3/4
Modulation	16QAM
Number of source packets	10,000
Number of transmitted packets	13,000
Number of bits per source packet	165
BEC erasure probability	P_e=0.1
LT degree of distribution	Improved Robust Soliton Degree Dist. (IRSDD)

Table 3.3: System parameters of the serially concatenated LT-BICM-ID scheme, Using IRSDD for LT codes.

EXIT characteristics of the demodulator for E_b/N_0 values ranging from 1dB to 9dB using Ungerböck's Set-Partitioning (SP) based bit-to-symbol mapping strategy. The extrinsic information characteristics for BICM-ID having a coding rate of $R = 3/4$ and employing different memory lengths of m=3, 4 and 5 are shown by the $I_A(Dec)$ versus $I_E(Dec)$ curves of Fig. 3.22. The BER of the decoder is plotted in the same figure with the aid of the solid horizontal and vertical lines. More explicitly, since there is an unambiguous one-to-one relationship between I^A_{Dec} and the BER for transmission over an AWGN channel, the output characteristic of the BICM-ID decoder is independent of the channel SNR. Hence, the BER curve is based on the function of $P_b = f'[I_A(Dec)]$, as detailed in [53]. For further illustration, at $I_A(Dec)$=0.800083 for example, we have BER=0.026177, as shown in Fig. 3.22. The threshold value used by the feedback link from the LT decoder to the LLR estimator buffer seen in Fig. 3.20 determines the minimum number of redundant packets required for successful LT decoding. In our forthcoming discourse we will briefly characterize the intricate interplay between the LT code's generator matrix employing different degree of distributions and the specific portion of redundant packets providing vital information for the LLR estimation, which can be used for adjusting the LLR threshold value. The Improved Robust Soliton Degree Distribution (IRSDD) of (3.6) in Section 3.1.1 having parameters of c=0.1 and δ=0.5 [20] was used, which required 10 percent redundant packets [20]. Based on the system parameters listed in Table 3.3, we conducted a series of simulations in order to determine the LLR threshold required for operation at certain BERs at the targeted E_b/N_0 value. Fig. 3.23 shows the LLR threshold value determined by simulation for maintaining a normalised E-value of 0.1 at E_b/N_0=5.1dB, 5.2dB and 5.5dB, respectively. The horizontal section of the curves represents the specific extrapolation of LLR threshold values, when the 10% target value of E has been reached. The

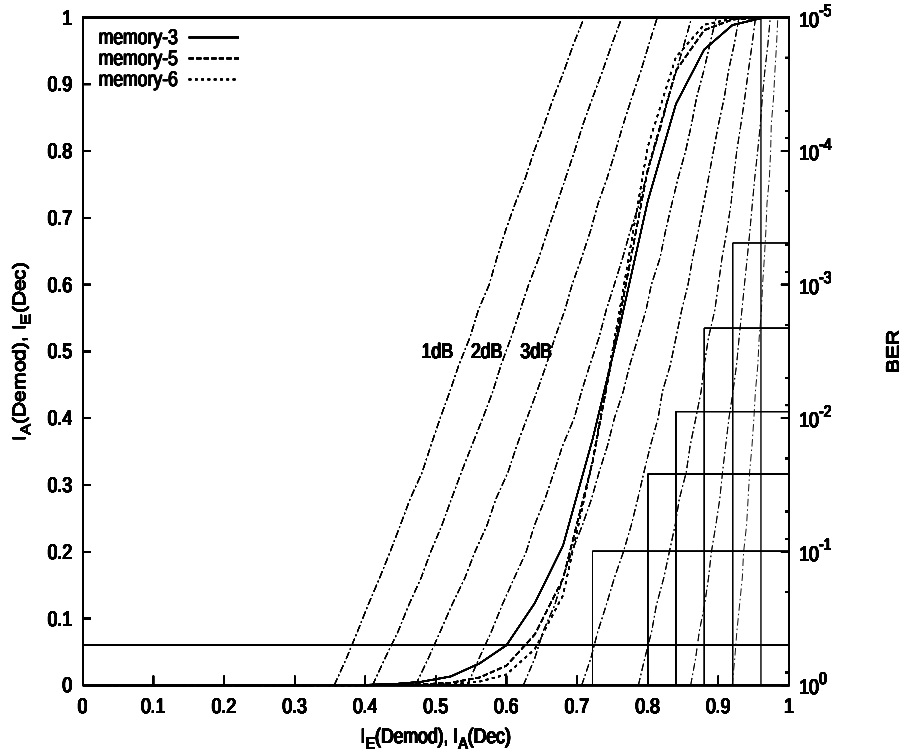


Figure 3.22: EXIT chart for BICM-ID.

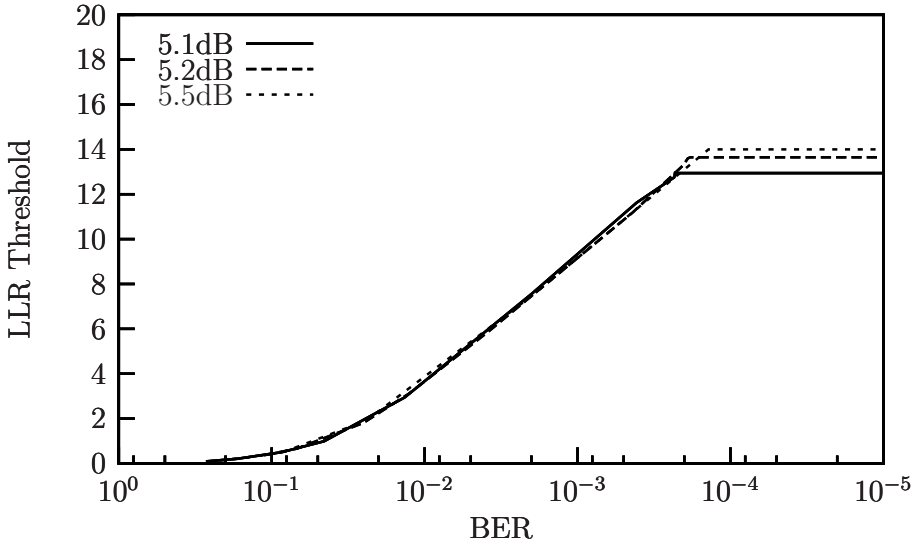


Figure 3.23: LLR threshold value necessitated for requiring E=10% redundant packets for the system of Table 3.3. The required values were averaged over 165 bit/packet for a total of 10,000 LT source packets.

LLR-based packet reliability evaluation becomes sufficiently confident, when the BER of the decoder becomes as low as $2{\cdot}10^{-4}$.

In Fig. 3.24 the LLR values are plotted for 200 bits against the bit-index taken from a random sample. The entire sample has a total of [165× 13,000× (1-0.1)= 1,930,500] bits based on Table 3.3. The circles denote the correctly decoded bits, while the crosses denote the erroneously decoded bits. Note that most of the bits that are located outside the threshold region represented by the horizontal lines are indeed likely to be near-error-free, since at the BER of (2×10^{-4}) on average a 20,000 bit segment would have a single error. By contrast, all the corrupted bits tend to be inside the threshold region, even though some LLR values within the same region may also yield correct bits, when they are carried back to the erro-free zone by a noise sample. Since we require a BER between 10^{-3} to

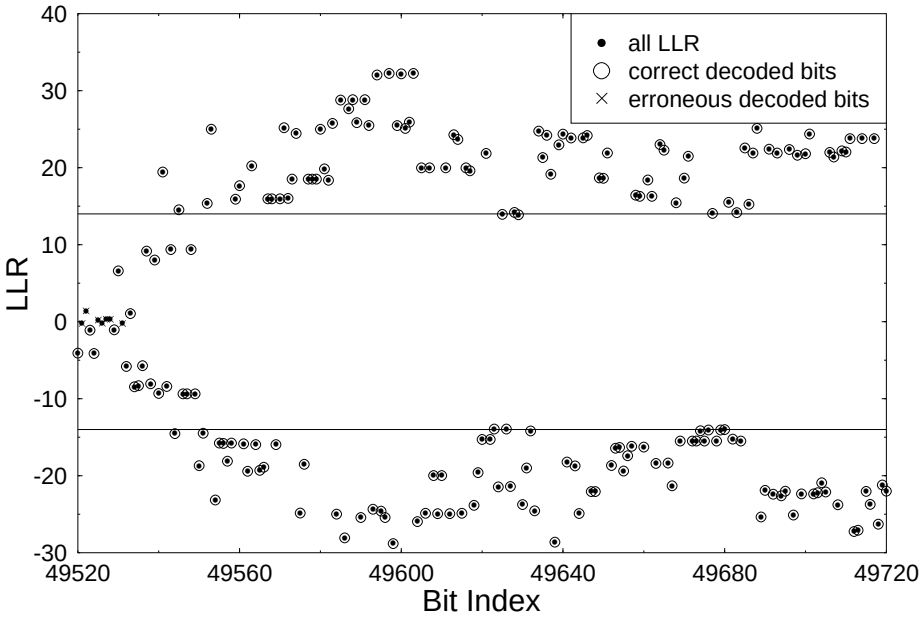


Figure 3.24: A set of 200 LLR values recorded for the system detailed in Table 3.3, given the threshold value of Fig. 3.23 at $E_b/N_0 = 5.5dB$.

10^{-4} in order to obtain a confident LLR based packet-reliability decision we observe in Fig. 3.22 that the BER of 3.10^{-4} satisfies this requirement at $I_A \approx 0.92$. This may also be associated with $E_b/N_0 > 5.0$ dB for four or more iterations.

3.3.3 Bit-by-bit LT Decoding

Again, it is widely recognized that LT packet decoding is sensitive to error propagation. Since it was originally designed for the BEC channel, the BER of the AWGN-contaminated packets may be mitigated by the BICM-ID decoder. An unsuccessfully decoded BICM-ID symbol is likely to cause error propagation to other LT packets. Owning to their modulo-2 connection, if the BEC's erasure probability P_e is high and the number of redundant LT packets is insufficient, the LT packet decoding operation will be curtailed. This problem can be mitigated by using bit-by-bit LT decoding, as outlines below with reference to Fig. 3.25. An extra advantage of this technique is that the complexity of bit-by-bit decoding reduces, because the contaminated low-reliability bits detected by the LLR estimation process are removed from the received packets and hence the total number of XOR calculations required decreases. Fig. 3.25 illustrates (a) our packet-based LT-decoding and

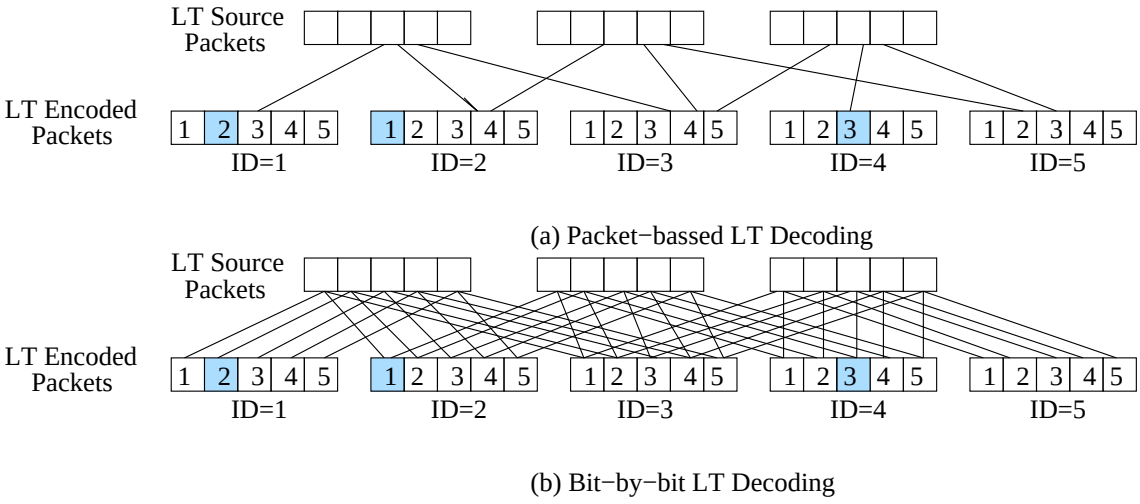


Figure 3.25: Packet-by-packet based LT decoding and bit-by-bit based LT decoding.

(b) bit-by-bit LT-decoding strategies. The corrupted bits of the packets are shown as by the gray-shaded rectangles in the figure. When using bit-by-bit decoding, the index of the corrupted bits can be identified with the aid of LLR estimation and this enables us to use only the reliable LT-encoded bits and hence to continue the LT decoding process.

3.3.4 Pseudo Random LT Generator Matrix

The design of LT codes critically depends on their so-called degree distribution [10]. It was also stated in [20, 36] that the specific degree distribution of LT codes has a strong influence on the percentage of redundant packets required for near-error-free detection, which is the E-value of the buffer feedback link of Fig. 3.20. Given a specific LT-code degree distribution, a further influential factor in determining the performance of a LT code is the particular choice of the random generator, which controls the actual assignment of the set of original source packets contributing to a specific LT-encoded packets, as shown in Fig. 3.25. Since this problem has not been addressed in the open literature, we propose a novel technique for reducing the minimum required fraction of redundant packets in the context of LT codes. The conventional technique of generating pseudo-random integers for pin-pointing the particular original information packets to be combined with the aid of modulo-2 additions to generate a given LT-encoded packet is using finite-length Shift Registers (SR). A block diagram of the LT encoding process is portrayed in Fig. 3.26. The

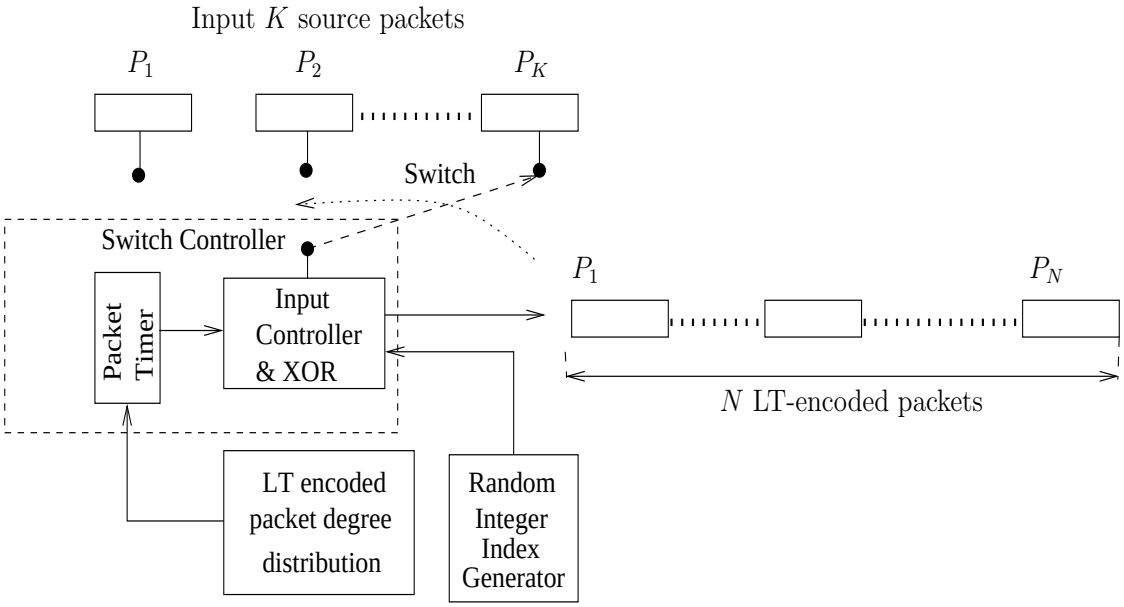


Figure 3.26: Schematic of the LT encoder.

LT encoding process is based on the long-term statistical degree distribution determining the average number of LT-encoded packets checking a specific source packet and on the random integer generator coining the specific degree of an LT source packet. As seen

in Fig. 3.26, the LT-encoded packet degrees are determined by the degree distribution previously proposed in Section 3.1.1, while the LT source input packet degree distribution is determined by the random integer index generators to be outlined in this section. The LT parity packets are created as follows. Initially, all values of the LT-encoded packets are set to logical '0'. The LT-encoded degree distribution generator of Fig. 3.26 coins a specific degree for each LT-encoded packet, where each new degree value corresponds to a sampling instant of the switch controlled by the Switch Controller of Fig. 3.26. For instance, if the LT-encoded packet degree distribution creates a LT-encoded packet having a degree of four, the Input Controller randomly picks four source packets and their XOR function becomes the current LT-encoded packet. The random integer index generator determines which particular source packets are combined to create the current LT-encoded packet using the XOR function. For example, if the random integer index generator is a pseudo-random integer generator, then the Input Controller hops across the source packets pseudo-randomly and creates the XOR function of four source packets to create the current LT-encoded packet. If this random generator was capable of creating a perfectly equiprobable distribution, then all parity packets would be created from the same number of source packets and hence all source packets would contribute to the same number of parity packets. Bellow two different random integer generators, namely the Linear Congruential Random Integer Generator (LCRIG) and the Swapping Bit Random Integer Generator (SBRIG) are detailed.

Given an m-stage SR, the number of possible integer values that may be generated is $M = (2^m - 1)$ and the next integer number I_n at the output of the SR is uniquely determined by the previous integer number $(I_n - 1)$ according to the function of $I_n = f(I_n - 1)$, which maps the finite set $\{I\}$ of integers onto itself. More explicitly, the output of the SR traverses from I_{n-1} to I_n by picking any of the $M = (2^m - 1)$ elements with the same probability. As an alternative, a sequence of integer numbers may be generated by linear congruential generators [54], which obey the following relation:

$$I_{j+1} = (aI_j + C)\textbf{mod } M, \tag{3.9}$$

where M is the modulus, while a and C are positive integers referred to as the multiplier and the increment, respectively [54]. The period of the shift register is never longer than $M = (2^m - 1)$. When the number of integers created is approaching its maximum legitimate value of $(2^m - 1)$, the sequence generated becomes more correlated, i.e. loses its pseudo-random nature. As a further feature, the linear congruential generator of (3.9) makes the

least significant bits more correlated than the most significant bits. The random integer number generator used for LT codes in [10] [39] and [20] has the length of $M = (2^{32} - 1)$, which was generated by a 32-bit SR and the process is based on the formula [55]:

$$I_n = (I_{n-1} + C)\mathbf{mod}\ M. \tag{3.10}$$

Here we opted for using the random sequence generated using (3.10) for LT encoding, when randomly choosing the modulo-2 connections between the LT source packets and the LT-encoded packets, as illustrated in Fig. 3.25. An example of the degree i.e. number of connections between the LT source and LT-encoded packets is shown in Fig. 3.27, corresponding to $(1.06 \times 2,011)$ LT-encoded packets, where the multiplier 1.06 indicates that 6% redundant packets were generated. Note from Fig. 3.27 that there are some LT

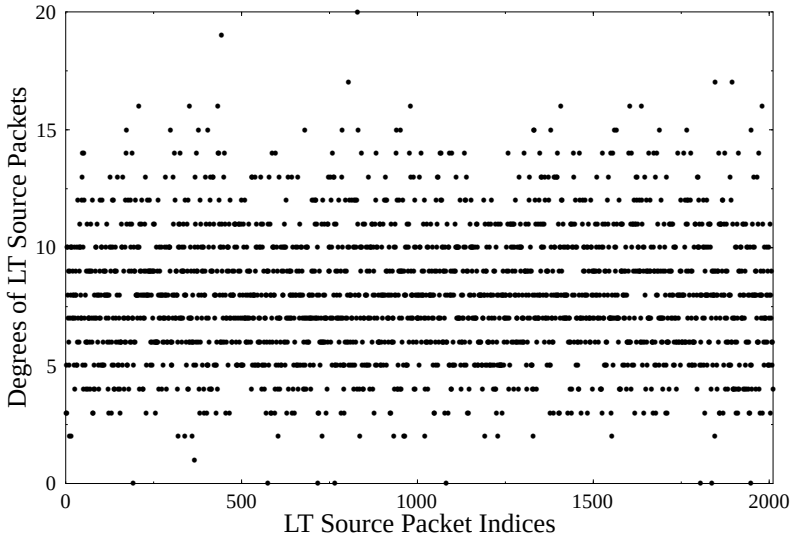


Figure 3.27: A snapshot of the integer random numbers generated by the congruential generator of (3.10) for $M = 32$.

packet indices, which are not used for generating any of the LT-encoded packets after the random selection. In other words, their degree is zero, as shown in Fig. 3.27. This implies that the absence of these LT packets after LT encoding causes the failure of the LT decoding process, since they cannot be recovered. In this case, to ensure that all source packets are represented at the decoder, we have to increase the total number of LT-encoded packets by increasing the redundant overhead. For the sake of avoiding this, we introduce a novel solution for designing a better random integer number generator for LT codes. In order to generate an uncorrelated set of integer numbers, we invoke the specific random generator relying on rearranging the bits of a shift register, as proposed in [52] [54]. We

introduce two different types of so-called random rotation generators. The first method suggests that the bits are rotated after their modulo 2 addition [52].

$$I_n = \left[(I_{n-j} + I_{n-k}) \textbf{ mod } 2^b\right] \textbf{ rot } r. \tag{3.11}$$

By contrast, the second method proposes that the bits are rearranged before their modulo 2 addition [52].

$$I_n = [(I_{n-j} \textbf{ rot } r_1) + (I_{n-k} \textbf{ rot } r_2)] \textbf{ mod } 2^b, \tag{3.12}$$

where I_n is an integer represented by b bits and the notation $I_{n-j}(\textbf{rot})r_1$ means that the bits of I_{n-j} are shifted to the right by r_1 positions, for example according to $[00001111_2\ (\textbf{rot})3 = 11100001_2]$. When using the pseudo-random integer number generator of (3.12), the LT code achieves a better performance than that using the traditional linear congruential pseudo-random integer number generator of (3.10), as seen in Fig. 3.29, Fig. 3.30 and Fig. 3.31. Fig. 3.28 shows the evolution of the degree of connections for the LT source packet indices after encoding (1.06× 2011) number of LT-encoded packets using the pseudo-random integer number generator of (3.12). Observe that this generator outperforms that of (3.10), which was characterized in Fig. 3.27, since in Fig. 3.28 no degree-zero connections are found, which can cause potential decoder failures. We further investigate

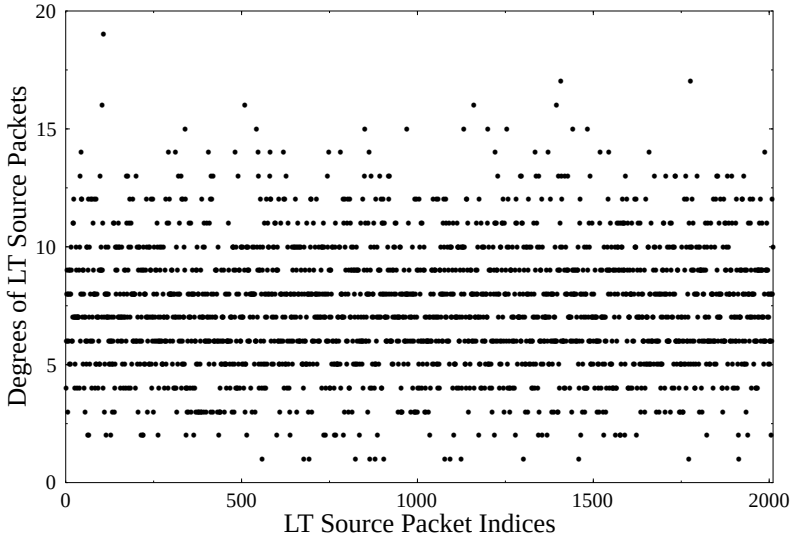


Figure 3.28: A snapshot of the integer random numbers generated by the pseudo-random generator of (3.12).

the benefits of the specific pseudo-random generator of (3.12) by comparing the normalised number of packets required for successful LT decoding. The parameters of the specific LT

degree of distribution of (3.6) used in the simulations are c=0.1 and δ=0.5 [20] and the required number of LT-encoded packets is normalized by the number of LT source packets K. We observe from Fig. 3.29 and Fig. 3.30 that the number of LT-encoded packets required in Fig. 3.29 is lower than that in Fig. 3.30. More specifically, the required number of LT-encoded packets only varies in the range spanning from 1.06 to 1.085 in Fig. 3.29, while it varies from 1.06 to 1.10 in Fig. 3.30.

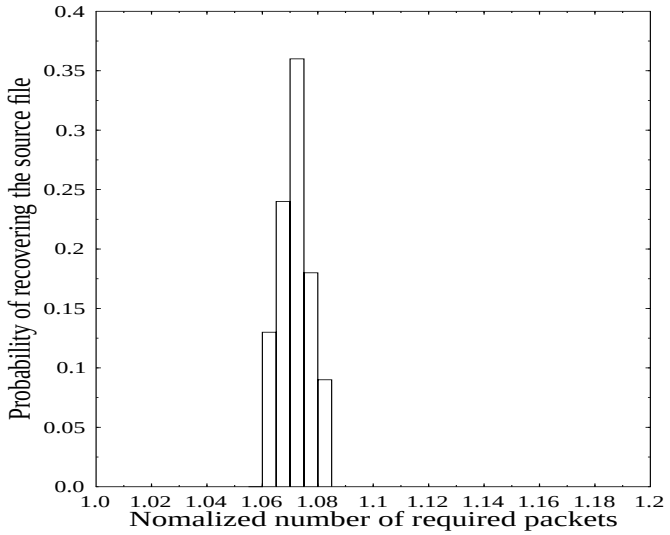


Figure 3.29: The degree histogram of the LT code using the pseudo-random integer number generator of (3.12).

3.3.5 Simulation Results

In this section we embark on quantifying the achievable performance of our proposed joint LT and BICM-ID scheme. The system parameters employed in our simulations are shown in Table 3.3. When encoding K= 2,011 source packets, Fig. 3.31 shows the BER versus the normalized number of LT-encoded packets using both the pseudo-random and the congruential pseudo-random integer number generators of (3.10) and (3.12). It can be seen from Fig. 3.31 that the random generator of (3.12) reduces the total number of LT-encoded packets compared to that of (3.10) by 3% from 15% to 12% at BER=10^{-5}. The BER performance of the BICM-ID scheme using different number of decoding iterations combined with LT coding is shown in Fig. 3.32 in comparison to that of our benchmark scheme operating without any exchange of information between the BICM-ID and LT

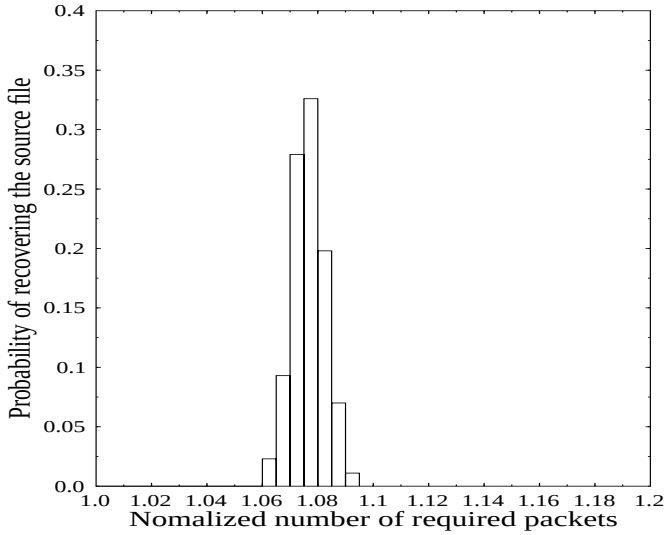


Figure 3.30: The degree histogram of the LT code using the linear congruential pseudo-random integer number generator of (3.10).

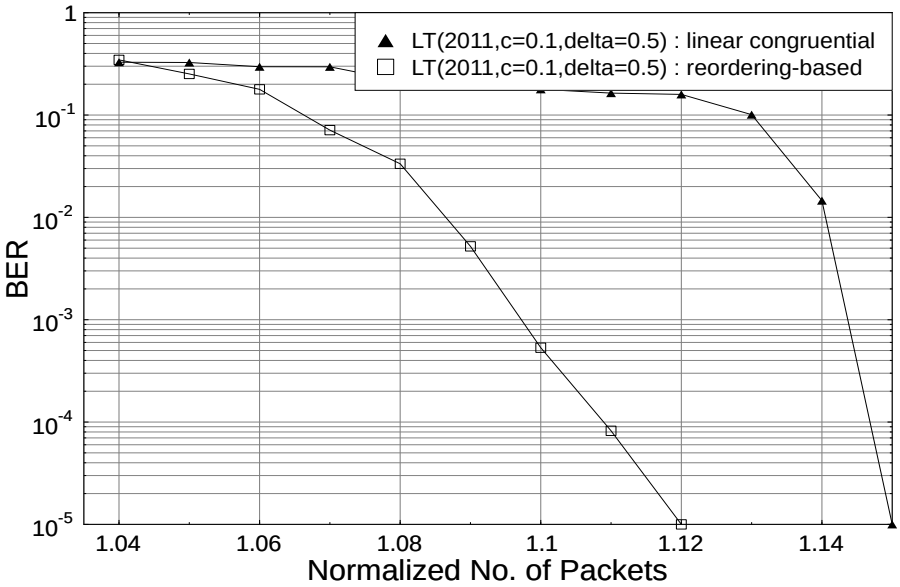


Figure 3.31: BER versus the number of LT-encoded packets, when transmitting 2011 packets over a BEC having $P_e = 0.1$ and contaminated by AWGN in conjunction with the system parameters of Table 3.3 and using c=0.1, δ=0.5.

decoder. An infinitesimally low BER is achieved for E_b/N_0 values in excess of 7.4dB. When using an exchange of soft information between the LLR buffer and LT decoder

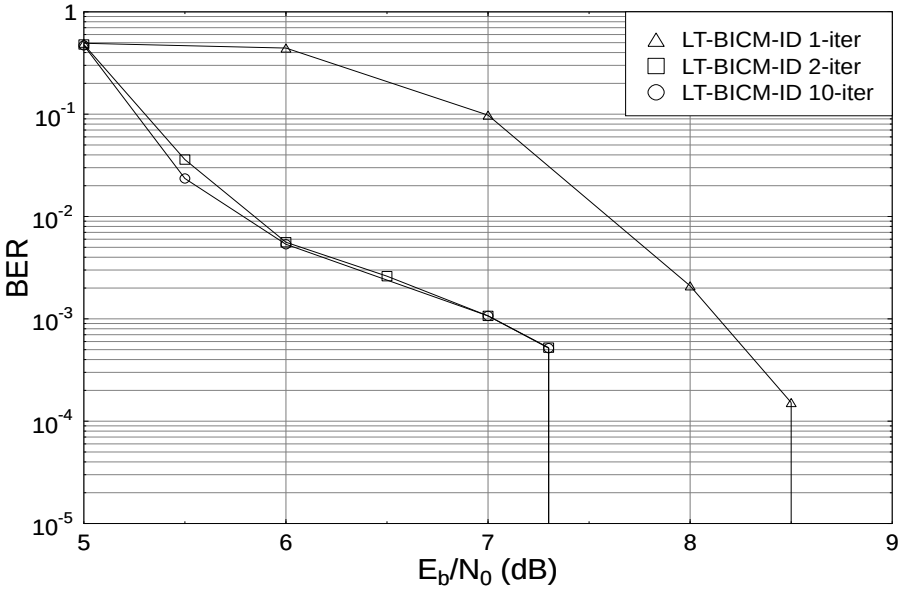


Figure 3.32: BER versus E_b/N_0 performance, when communicating over a BEC channel having $P_e = 0.1$ and contaminated by AWGN, when using the degree distribution of (3.12).

of Fig. 3.20, Fig. 3.34 shows the significant performance improvement achieved by the system. Figs 3.33 and 3.34 include the results recorded for both the packet-by-packet LT decoding as well as for the bit-by-bit LT decoding.

Finally, in Fig 3.35 we compare the bit-rearrangement based pseudo random generator of (3.12) and the traditional congruent random generator of (3.10) used for specifying the modulo-2 connections between the source packets and the LT-encoded packets. The technique of (3.12) is capable of reducing the total number of received packets required for achieving an infinitesimally low BER. More explicitly, Fig. 3.35 shows the improved BER performance recorded, when using $K = 11,000$ packets instead of the originally stated $K=$ 13,000 packets specified in Table 3.3. The novel features of the jointly designed BICM-ID and LT coding scheme included:

1. A bit-by-bit rather than packet-by-packet LT decoder;

2. An LLR-based reliability estimator designed for the sake of avoiding the classic CRC overhead;

3. The bit-rearrangement based random generator of (3.12), which resulted in an im-

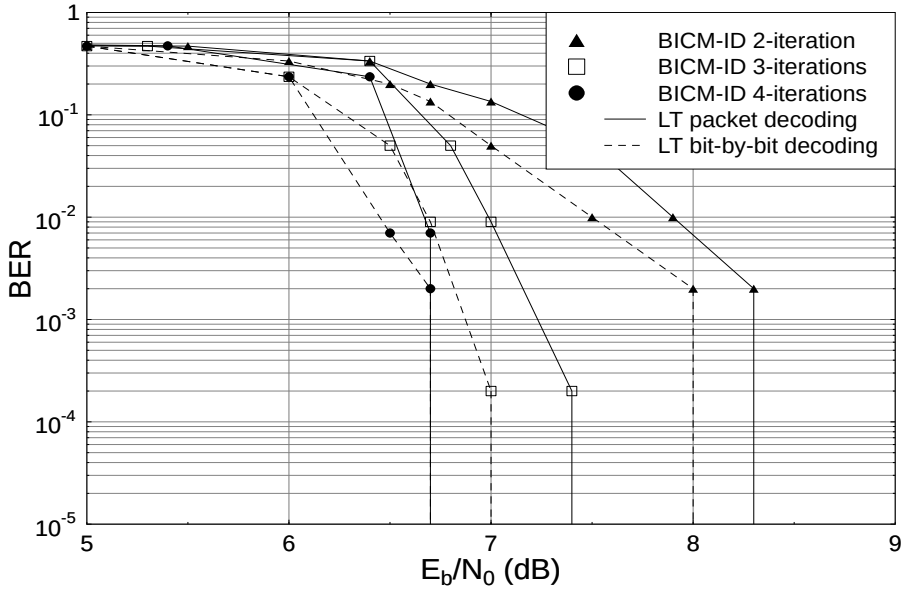


Figure 3.33: BER versus E_b/N_0 performance, when communicating over a BEC channel having $P_e = 0.1$ contaminated by AWGN using perfect CRC estimation as well as the parameters of Table 3.3.

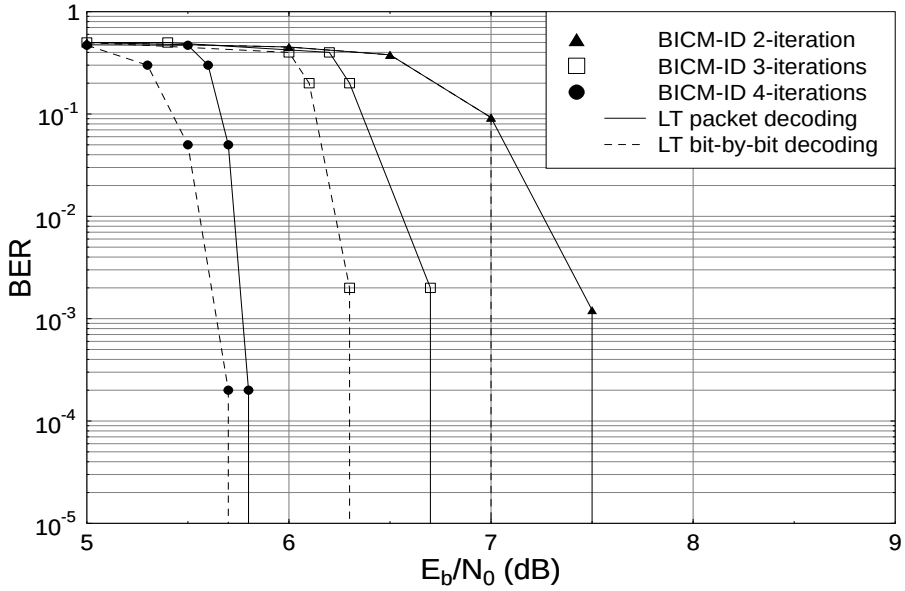


Figure 3.34: BER versus E_b/N_0 performance, when communicating over a BEC channel having $P_e = 0.1$ and contaminated by AWGN using LLR based reliability estimation and the parameters of Table 3.3.

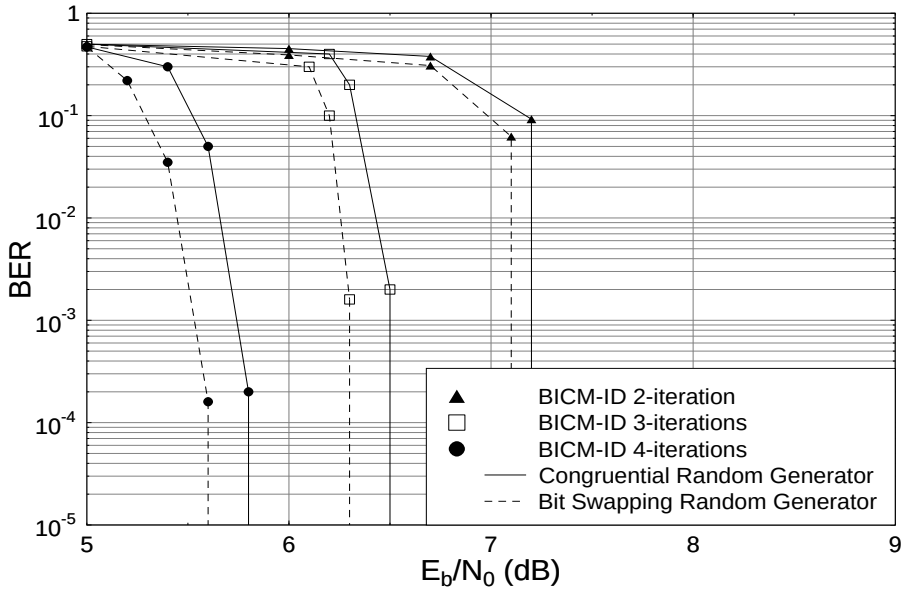


Figure 3.35: BER versus E_b/N_0 performance when communicating over a BEC channel having $P_e = 0.1$ and contaminated by AWGN using LLR based reliability estimation and the parameters of Table 3.3. The number of source packets was reduced to $K = 11,000$ from $K = 13,000$.

proved LT-encoder and hence was capable of reducing the LT-encoded packet overhead required for maintaining an infinitesimally low BER by about 3%.

3.4 LT Coding Aided Iterative Detection for Downlink SDMA Systems

As another sophisticated LT-coded system design example, in this section we consider a downlink Spatial Division Multiple Access (SDMA) system using iterative detection, which invokes a reduced-complexity near-Maximum-Likelihood (ML) Sphere Decoder (SD) [56]. The Ethernet-based Internet section of the transmission chain inflicts random packet erasures, which is modelled by the BEC, while the wireless DownLink (DL) imposes both fading and noise. The novel LLR based packet reliability metric of Section 3.3 is used for identifying the low-reliability channel-decoded packets, which are likely to be error-infested. Packets having residual errors must not be passed on to the LT decoder for the sake of avoiding LT-decoding-induced error propagation. We will demonstrate that the proposed scheme is capable of maintaining an infinitesimally low packet error ratio in the downlink of the wireless Internet for E_b/N_0 values in excess of about 3dB. In this section

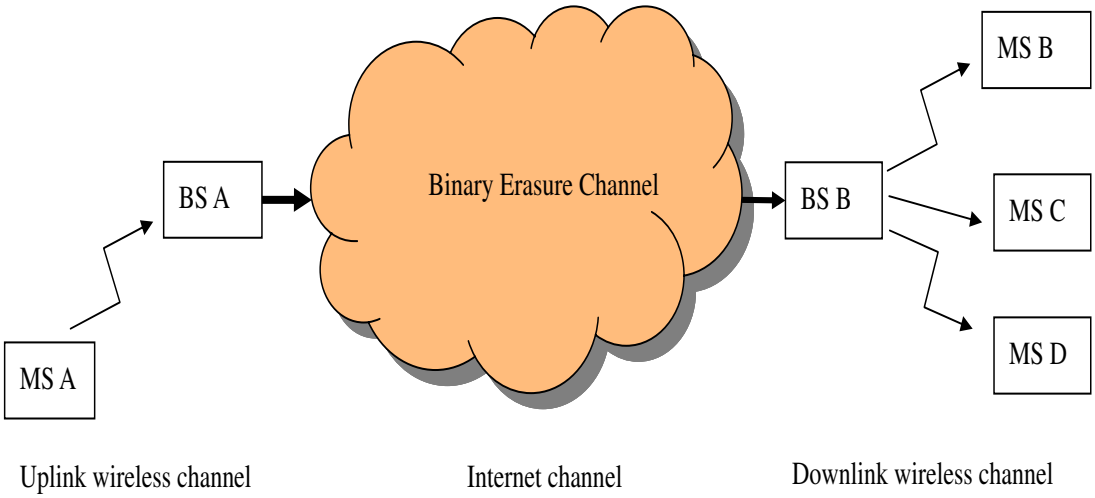


Figure 3.36: The wireless Internet.

we considered the system configuration illustrated in Fig. 3.36 and studied an LT-coding aided Down-Link SDMA system (LT-DL-SDMA). As shown in Fig. 3.36, we considered the scenario, when Mobile Station (MS) B would like to receive data packets from MS A. It may be assumed that BS A and B are connected to each other via either the conven-

tional Ethernet-based Internet or by a dedicated optical fiber backbone. In this study we consider the Ethernet-based Internet aided scenario, which inflicts packet erasures owing to statistical multiplexing-induced packet collision events. First, the data packets are generated either by a conventional Ethernet-based terminal or transmitted from MS A via the wireless uplink channel to Base Station (BS) A. Then the data packets are routed through the statistical multiplexing aided Ethernet-based BS-BS channel, where some packets may be lost. These packet-loss events can be modeled by the BEC [57], which is characterized by a single parameter, namely by its packet dropping probability P_e. Finally, BS B receives the data packets from the Internet and then transmits them via the wireless downlink channel to MS B. As shown in Fig. 3.37, the LT packets generated by MS A and received by BS A are conveyed first via the Ethernet-based, Internet modeled by the BEC channel, while BS B at the output of the BEC channel forwards the LT packets to the target user. Since we focus our attention on the DL transceiver design in this treatise, we assume that the packet source is either a fixed terminal directly connected to the conventional Ethernet or, alternatively, that no impairments are imposed by the uplink wireless channel between MS A and BS A.

In our DL transmitter design, the LT packets are forwarded to the convolutional channel encoder of Fig. 3.37 followed by an interleaver, as well as to a unity-rate inner encoder [58] and finally to the modulator. The philosophy of SDMA is that the users are differentiated with the aid of their unique, user-specific Channel Impulse Response (CIR), provided that they are sufficiently different. The modulated signal is transmitted by the multi-user DL-SDMA scheme of Fig. 3.38, which employs the Singular Value Decomposition (SVD) based spatio-temporal SDMA DL pre-processing technique proposed by Choi and Murch [59], allowing for a specific user to receive his/her dedicated signal, essentially free from Multi-User Interference (MUI) inflicted by other users. In order to maintain a high data throughput for the system, a novel detector suitable for the so-called rank-deficient scenario is considered in the proposed DL-SDMA system [60], where the number of transmitters exceeds the number of receivers [56]. While linear detectors, such as the Minimum Mean Square Error (MMSE) detector [61] were shown to perform unsatisfactorily in high-throughput rank-deficient scenarios, the novel Optimized Hierarchy Reduced Search Algorithm (OHRSA)-aided ML detection method of [62] exhibits a relatively low complexity even in highly rank-deficient scenarios and thus its employment is meritorious. In order to further increase the attainable performance, additionally a unity-rate precoder [58] is employed in the iterative decoding aided system, where extrinsic information is exchanged

between the precoder's decoder and the convolutional channel decoder of Fig. 3.39. The role of the precoder is to efficiently disperse the extrinsic information without increasing the delay and the decoding complexity, as a benefit of its Infinite Impulse Response (IIR). Furthermore, based on the soft-bit outputs of the channel decoder, we will use the packet reliability metric calculation technique of Section 3.3 for quantifying the Packet Reliability Information (PRI) forwarded to the LT decoder, which will be used for informing it as to whether the packet considered is sufficiently reliable for employment by the LT decoder. More explicitly, based on the PRI, the LT decoder erases the unreliable packets, namely those that are likely to contain errors, in order to perfectly recover all the original data packets. By contrast, unreliable packets must not be forwarded to the LT decoder, since they would inflict a high number of further propagated errors, depending on the specific degree-distribution of the LT code.

3.4.1 System Overview

Again, the system's schematic is outlined in gradually increasing detail in Figs 1 - 4. Let us consider, for example, the transmission of 13,000 LT-encoded packets, which were generated from 10,000 source packets by incorporating 30% redundancy. At a packet erasure probability of $P_e = 0.1$ we will lose 1,300 randomly positioned LT packets, when the LT packets are routed over the Internet. The LT packets, which do reach BS B will be transmitted via the wireless downlink channel to MS B of Fig. 3.36. The DL transmitter of the BS encodes the bits of the LT-encoded packets using both a convolutional encoder and a unity-rate precoder, as shown in Fig. 3.37. Again, the rationale of using the unity-rate precoder serially concatenated with the convolutional encoder is that-as a benefit of its IIR-it improves the convergence of the iterative receiver, when its decoder exchanges extrinsic information with the convolutional decoder of Fig. 3.39 [58]. When using the system parameters summarized in Table 3.4.4, the DL-SDMA system employed the SVD-based Multi-User Transmission (MUT) scheme of Fig. 3.38 for transmitting the data stream of MS B [60] in Fig. 3.36. Fig. 3.39 details the design of the MS's receiver, again, utilizing the OHRSA-MUD of [62], which is suitable for high-throughput rank-deficient scenarios. Finally, the LT decoder assisted by the PRI provided by the convolutional decoder generates the recovered source packets constituting the original data file. In our forthcoming discussions we will briefly summarize the operation of the MUT scheme [60] utilized and outline the corresponding receiver design employing iterative decoding.

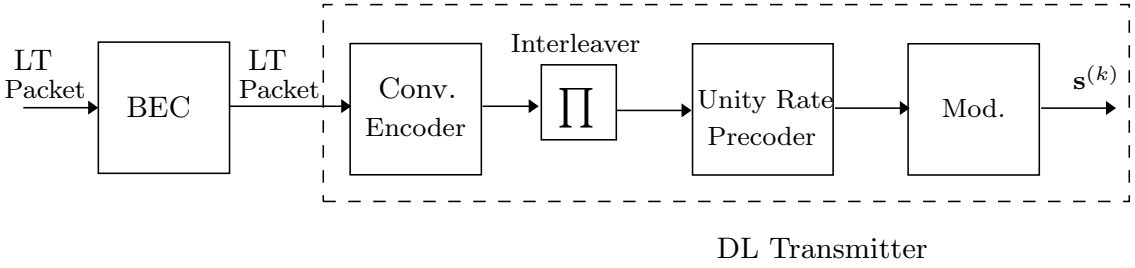


Figure 3.37: Transmitter of the k-th user.

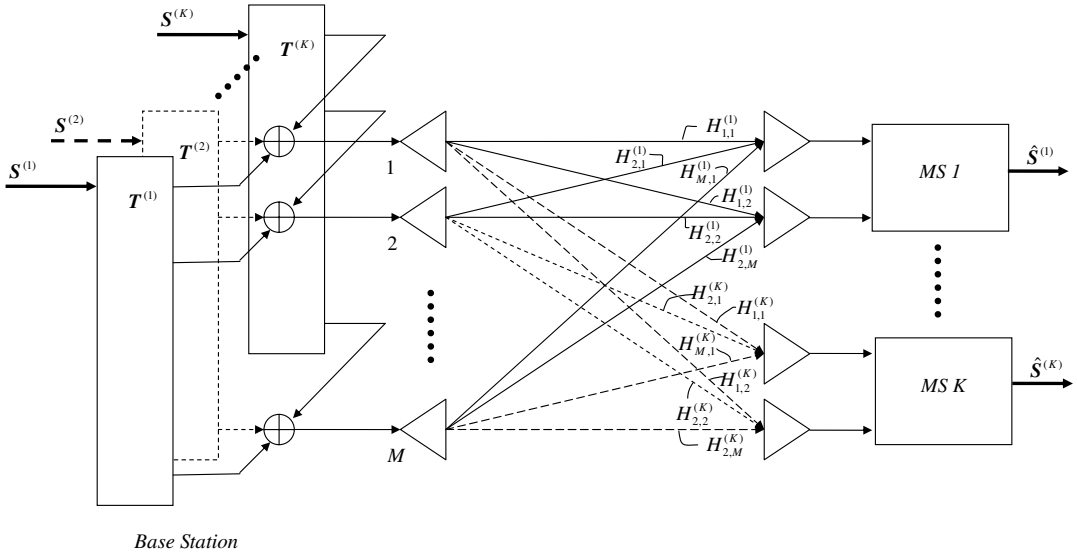


Figure 3.38: Multiuser transmission in the LT-DL-SDMA system using the single-user transmitter of Fig. 3.37.

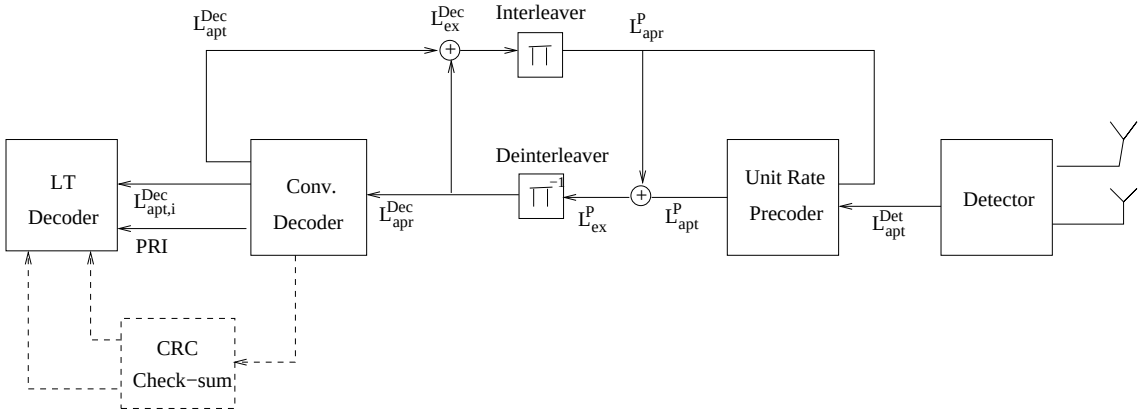


Figure 3.39: LT coding aided iteratively decoded MS receiver design.

3.4.2 Multi-User Transmission Scheme

Again, the MUT scheme considered is depicted in Fig. 3.38. More specifically, the BS employs M' transmit antennas for supporting K' MSs, each employing N'_k receive antennas. In this section we consider a flat-fading Multi-Input Multi-Output (MIMO) channel. The MIMO channel corresponding to the k-th user may be described by an $(N'_k \times M')$-dimensional complex-valued time-domain CIR matrix $\mathbf{H}'^{(k)}$. Each element of $\mathbf{H}'^{(k)}$ is characterized by a complex-valued scalar channel coefficient $H'^{(k)}_{ij}$, which represents the non-dispersive link between the i-th BS's transmit antenna and the j-th MS receiver antenna of the k-th user. More explicitly, each fading MIMO link is modeled by an i.i.d. complex Gaussian random variable having a variance of unity and a mean of zero. In order to eliminate the MUI, let us now construct the MUT preprocessor $\mathbf{T}_k$ of Fig. 3.38. As outlined in [60], $\mathbf{T}_k$ can be generated based on the null space $\mathbf{V}^{(k)}$ of $\tilde{\mathbf{H}}'^{(k)}$, calculated using the SVD [63] of $\tilde{\mathbf{H}}'^{(k)}$ expressed as:

$$\tilde{\mathbf{H}}'^{(k)} = \begin{pmatrix} \tilde{\mathbf{U}}^{(k)} & \mathbf{U}^{(k)} \end{pmatrix} \cdot \begin{pmatrix} \sum & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \cdot \begin{pmatrix} \tilde{\mathbf{V}}^{(k)H'} \\ \mathbf{V}^{(k)H'} \end{pmatrix}, \tag{3.13}$$

where

$$\tilde{\mathbf{H}}'^{(k)} = \begin{pmatrix} \mathbf{H}'^{(1)} & \cdots & \mathbf{H}'^{(k-1)} & \mathbf{H}'^{(k+1)} & \cdots & \mathbf{H}'^{(K')} \end{pmatrix}^T. \tag{3.14}$$

As advocated in [60], we will categorize the potential system design options of an $(M' \times N')$-dimensional MIMO system, where M' and N' refer to the number of transmit and receive antennas. To elaborate a little further, the total number of antennas employed by all the

K' user terminals is $K'N'_k$. Let us now introduce a measure of the normalized *system load* expressed as

$$L_s = \frac{M'}{N'}. \tag{3.15}$$

Consequently, we may distinguish three different scenarios as follows:

1. lightly-loaded scenario, for $L_s < 1$;
2. fully-loaded scenario, for $L_s = 1$;
3. rank-deficient or 'over-loaded' scenario, for $L_s > 1$.

3.4.3 LT Coding Aided Receiver Using Iterative Detection

The detector constituting the first stage of the receiver seen in Fig. 3.39 is the OHRSA-MUD [62], which is adapted for employment in our system in order to reduce the computational complexity imposed by the ML detector used by each of the MSs. The derivation of an expression for the low-complexity evaluation of the soft-bit information associated with the bit estimates of the OHRSA detector was detailed in [62].

Iterative decoding is carried out by exchanging extrinsic information between the unity-rate precoder's decoder and the channel decoder. Fig. 3.39 illustrates the iterative receiver structure, where L represents the LLRs. The super-script Det indicates the detector, P denotes the precoder, while Dec represents the channel decoder. The subscripts apr, ex and apt in Fig. 3.39 indicate *a priori*, extrinsic and *a posteriori* LLRs, respectively. First, the unity-rate precoder's decoder seen in Fig. 3.39 processes the soft-bit output L_{apt}^{Det} of the detector generated in the previous stage and the *a priori* LLR values L_{apr}^{P}, which are appropriately arranged by the interleaver $\mathbf{\Pi}$ are produced from the extrinsic information L_{ex}^{Dec} of the channel decoder. Then the extrinsic information L_{ex}^{P} to be used by the unity-rate precoder's decoder in Fig. 3.39 is obtained from L_{apt}^{P} by subtracting the *a posteriori* LLR values L_{apr}^{P}. Then, as portrayed in Fig. 3.39, the channel decoder processes L_{apr}^{Dec}, which was generated by the deinterleaver $\mathbf{\Pi}^{-1}$ from L_{ex}^{P}, and outputs the *a posteriori* LLRs L_{apt}^{Dec} to be used as a feedback for the next decoding iteration. When the iterations are curtailed, the channel decoder outputs $L_{apt,i}^{Dec}$, which represents the hard-decision based data bits.

Recall from Section. 3.3 that the PRI is calculated based on the Log Likelihood Ratio (LLR) [30] of the bits. Hence we introduced an LLR threshold, denoted as l_t. A channel

decoded bit is deemed reliable, if the absolute value of its LLR is higher than l_t , where the threshold l_t has to be carefully chosen. By contrast, if the absolute value of the LLR is lower than l_t, it is deemed unreliable. Therefore, the PRI of a packet is simply calculated as the percentage of the reliable bits in the channel-decoded packet. On this basis we can identify packets, which are likely to contain decoding errors. Let us assume for example that the total number of the LT packets generated by the LT encoder is 13,000. However, only 11,700 packets are transmitted by the BS to MS B of Fig. 3.36 owing to the 10% packet loss rate of the BEC channel. It was found in our informal experiments not detailed here that the minimum number of packets required for achieving a high probability of successful LT decoding is 11 000. Then we have to consider two decoding scenarios, namely when the number of erroneous packets is higher than $(11,700 - 11,000) = 700$, as well as when it is lower than 700. In the former specific case, the 11,000 packets chosen for LT decoding may contain some incorrectly decoded LT-encoded packets, which hence result in error propagation. However, in this scenario the best course of action is to still forward the most reliable 11,000 packets for LT decoding based on the PRI. In other words, we remove the 700 most unreliable packets, because the chances are that this way we may indeed succeed in removing all the incorrect packets and hence successfully LT decode all the source packets.

3.4.4 Simulation Results

Here we continue our discourse by characterizing the Packet Error Ratio (PER) versus packet-length performance of the inner encoder/decoder scheme of Fig. 3.40. As expected, the PER degrades upon increasing the packet-length, since the probability of having residual errors after channel decoding increases upon increasing the packet-length. Furthermore, it can be shown that increasing the packet-length increases the LT-decoding complexity. By contrast, upon reducing the packet-length the relative overhead of using a 16-bit CRC sequence increases. As a compromise, based on the trends seen in Fig. 3.40, we opted for a packet-length of 120 bits. Observe furthermore that as expected, increasing the interleaver length has the beneficial effect of reducing the PER owing to randomizing the bit-errors more efficiently.

In this section, we characterize the achievable performance of the LT-DL-SDMA system assisted by both a 16-bit CRC-based and on the proposed LLR-based PRI estimation scheme. The packet erasure rate P_e of the BEC was set to 0.1 and each LT packet con-

LT Packet size	120 bits
LT Distribution	Improved robust distribution [20]
Number of information packets	10,000
Number of redundancy packets	3,000
Erasure probability of BEC	0.1
Recursive Systematic Code Interleaver length	RSC(5,7) 10^5 bits
Modulation	4 QAM
Number of users K'	3
Number of transmit antennas M'	6 for $L_s = 1$, 8 for $L_s = 1.333$
Number of receive antennas N_k	2
Normalized Doppler frequency	$f_d = f_D \cdot T = 0.001$

Table 3.4: System Parameters of LT Coding Aided Iterative Detection for Downlink SDMA Systems.

tained 120 bits. In conjunction with the CRC overhead, a source packet accommodated 136 bits. As seen in Table 3.4.4, we employed 4-QAM protected by the half-rate Recursive Systematic Convolutional code RSC(5,7) having a memory of three and using a 10^5-bit interleaver, where (5,7) represent the generator polynomials in octal format. A slow-fading MIMO channel having a normalized Dopper frequency of 0.001 was used.

As seen in Table 3.4.4, the system configuration studied supported $K' = 3$ users and each user terminal employed two receiver antennas. Finally, for the sake of convenience, we assume that each MS is receiving the same number of independent data streams L_t from the BS, which implies that each user has the same data throughput.

Fig. 3.41 portrays the PER performance of the inner decoder scheme of Fig. 3.39 corresponding to the normalized system load of $L_s = 1.0$, i.e. when the number of transmit antennas is $M' = 6$ and the number of independent data streams transmitted to each user terminal is $[L_t = M' - (K' - 1) \cdot N'_k = 2]$. This is a rather challenging DL MUD scenario, because the CIRs of the two streams of a specific user are quite similar to each other, which results in a high interference between them. In the CRC-assisted scheme of Fig. 3.39, the LT decoder is capable of avoiding the decoding of error-infested packets. As stated above, in the example considered the BEC erases 1,300 packets out of 13,000 and

hence 11,700 LT-encoded packets will be transmitted via the wireless downlink channel. Recall that since the LT decoder requires a minimum number of 11,000 correct packets, the maximum tolerable number of channel-impaired packets is 700, still allowing the LT decoder to recover the original source packets. This implies that if the PER of the inner decoder is lower than $(700/11,700 \approx 0.06)$, the CRC-assisted system is capable of completely recovering all the original source packets with a high probability. The horizontal dashed line in Fig. 3.41 characterizes the value of PER=0.06. Observe in Fig. 3.41 that the system using $I = 4$ iterations exhibits a PER $< 6\%$ for E_b/N_0 values in excess of about 3dB and hence it is highly likely to recover all packets correctly. Furthermore, the LLR-based scheme performs similarly to the CRC-aided arrangement, but the latter suffers from a slight E_b/N_0 loss owing to its CRC overhead of about 13.3%.

In summary, we proposed a novel LT-DL-SDMA system using iterative decoding. It was

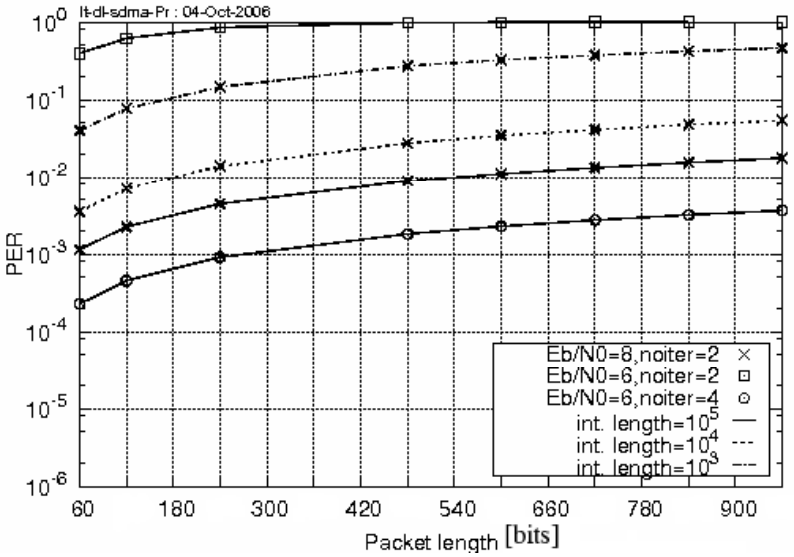


Figure 3.40: PER performance of the LT-DL-SDMA scheme having a normalized system load of $L_s = 1.333$ for different E_b/N_0 values, packet sizes and interleaver lengths. Our system supports $K' = 3$ users, where each user employs $N'_k = 2$ receives antennas.

also demonstrated that the LLR-based PRI-aided scheme slightly outperformed its CRC-based counterpart, owing to the E_b/N_0 penalty imposed by the 16-bit CRC overhead. It was shown in Fig. 3.41 that for E_b/N_0 values in excess of about 3dB an infinitesimally low PER may be achieved. Our future research will focus on improving the system's performance by exchanging soft information between the inner decoders and the outer LT decoder.

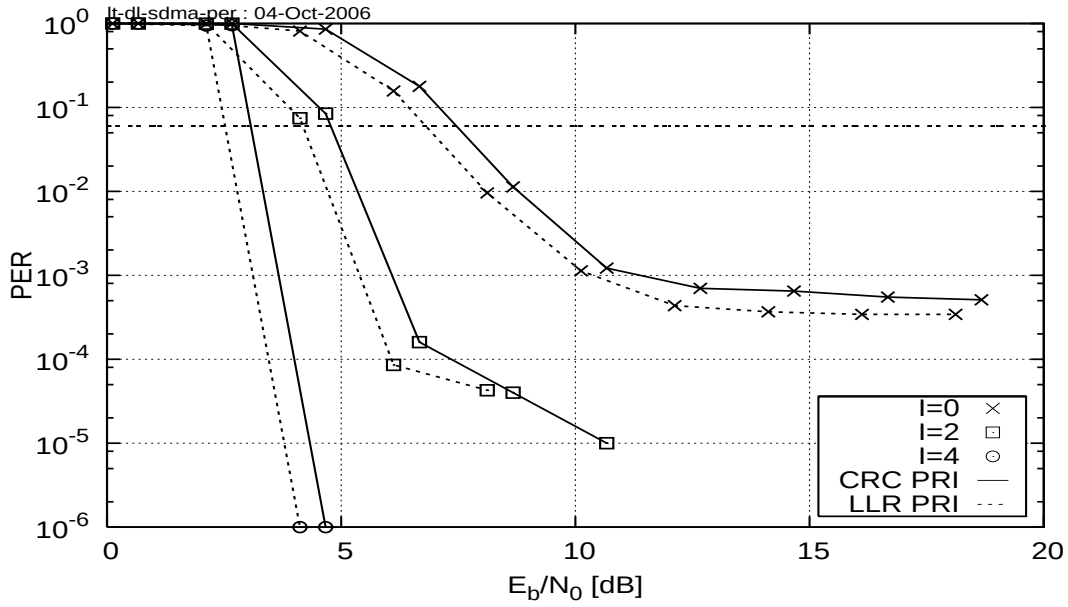


Figure 3.41: PER performance of the LT-DL-SDMA scheme system having a normalized system load of 1.0 using either the LLR-based or the CRC-based PRI for different number of iterations. This system supports $K' = 3$ users, where each user terminal employs $N'_k = 2$ receive antennas. All other system parameters are summarized in Table3.4.4.

3.5 Chapter Conclusions

Improving the degree distributions for the LT source and encoded packets helps LT codes achieve better BER performances as shown in the BER performances of the concatenated LT and BICM-ID system. Besides, the amalgamated scenarios between LT codes and other FEC codes such as General-Low Density Parity Check codes using the smark packing FEC codeword method into the LT packets helps LT codes countable the error ratio of contaminated LT packets. Knowing the error ratio of received LT packets, based on which the LT decoder can decide to choose the less erroneous packets for recovering the source packets, helps LT codes increase the potential ability of their BER performances in the combined systems for transmitting data over noisy channels. The new random integer generator designed for the degree distribution of LT input packets and the novel definition bit by bit decoding method aided by the LLRs error estimation reduce the complexity of the Luby transform codes and improve BER performances of the concatenated LT and FEC code systems in noisy channels. For the sake of improving the BER performance of LT codes we develop a soft-bit decoding algorithm and new structure designs for LT codes

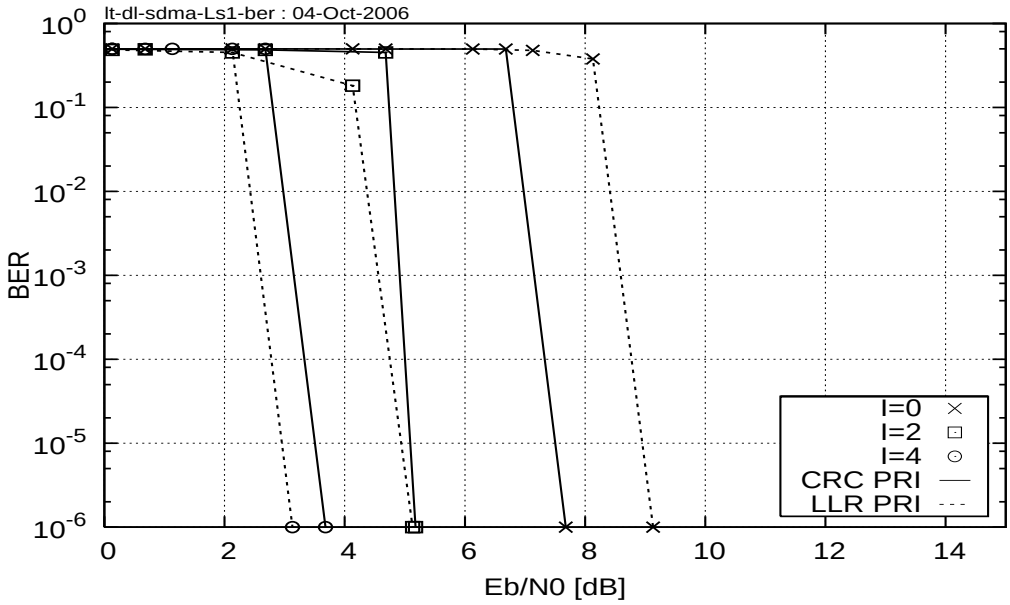


Figure 3.42: BER performance of the LT-DL-SDMA system having a normalized system load of 1.0 assisted with the PRI and CRC and compared with different number of iterations. This system supports $K' = 3$ users, where each user terminal employs $N'_k = 2$ receive antennas. All other system parameters are summarized in Table3.4.4.

in the next section.

Chapter 4

Luby Transform Codes Using Soft-Bit Decoding Algorithms

In this chapter we will design the soft-bit decoding of LT codes and evaluate their performance. Section 4.1 develops the concept of Systematic Luby Transform (SLT) codes, while Section 4.2 introduces their soft-bit decoding. A new random integer for determining the specific degree of the input LT packets and a novel truncated soliton degree distribution designed for Systematic Luby Transform codes are proposed in Section 4.3. Section 4.4 shows the BER performances of SLT codes using different degree distributions for the SLT-parity packets and different random integer generator designs for the degree distribution of SLT-source packets. Finally, Section 4.5 presents an application example in the context of an SLT coded V-BLAST system.

4.1 Systematic Luby Transform Encoding

The SLT encoder's operation may be followed with the aid of Fig. 4.1, where the original m-bit information packets are encoded by the SLT encoder, which generates n number of m-bit packets from k number of m-bit packets. Owing to its systematic encoding first the original k number of m-bit packets are directly copied into the encoder's output buffer and then $(n-k)$ number of m-bit parity packets are created according to the modulo-2 encoding rule of conventional LT codes. For the sake of a more compact graphical representation the n number of m-bit SLT encoded packets may be portrayed as a vertically stacked set of

n number of m-bit packets, as seen in Fig. 4.1. According to this representation the top k number of bits stacked vertically correspond to the bits belonging to the same bit positions in the k number of consecutive original m-bit packets, while the remaining $(n-k)$ bits at the bottom of each column represent the corresponding bits of the $(n-k)$ SLT-encoded parity packets. Hence, if we refer to the generator matrix and the parity check matrix of the SLT code as $\mathbf{G}$ and $\mathbf{H}$, then we have the syndrome equation of $(\mathbf{G} \cdot \mathbf{H}^T = \mathbf{0})$. The graphic representation of the SLT code seen in Fig. 4.1 may be interpreted as a set of m codewords of an (n, k) block code, such as an LDPC code for example, where each of the vertical (n, k) codes is represented by the generator and parity check matrices of $(\mathbf{G}_0; \mathbf{H}_0), (\mathbf{G}_1; \mathbf{H}_1), \cdots, (\mathbf{G}_m; \mathbf{H}_m)$, which obey the structure of Fig. 4.1.

Observation:

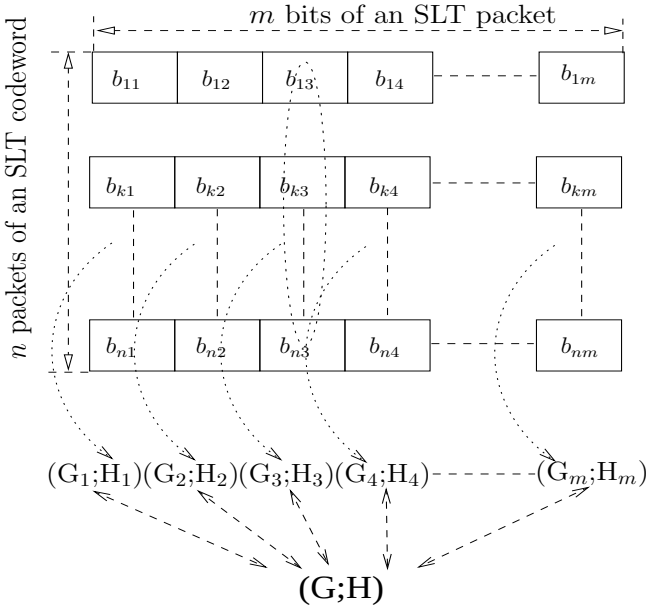


Figure 4.1: The m number of (n, k) codewords of SLT codes and their generator as well as parity check matrices.

If $\mathbf{G}$ and $\mathbf{H}$ are the generator matrix as well as the parity check matrix of a SLT code and they satisfy the syndrome equation $(\mathbf{G} \cdot \mathbf{H}^T = \mathbf{0})$, then all generator and parity check sub-matrices $\mathbf{G}_i$ and $\mathbf{H}_i$ also satisfy the equation $(\mathbf{G}_i \cdot \mathbf{H}_i^T = \mathbf{0};\ i = 1, \cdots, m)$.

Proof:

As seen in Fig. 4.1, the bits of the SLT codewords are generated by the resultant generator matrices $\mathbf{G}_i$ and all of these matrices satisfy $(\mathbf{G}_i \equiv \mathbf{G};\ i = 1, \cdots, m)$ and $(\mathbf{H}_i \equiv \mathbf{H};\ i = 1, \cdots, m)$. Since we have $(\mathbf{G} \cdot \mathbf{H}^T = \mathbf{0})$, consequently $(\mathbf{G}_i \cdot \mathbf{H}_i^T = \mathbf{0})$. From the above

proposition we infer the statement that a codeword of the SLT code has a length of n bits. Hence, we infer the following Lemma as:

SLT Decoding Algorithm:

The decoding process of a single codeword of the SLT code may be decomposed into m decoding processes of the (n, k) constituent codewords.

4.2 Soft-Bit Decoding of Systematic Luby Transform Codes

4.2.1 Preliminaries

In this section, we comment the rationale of designing novel systematic Luby transform codes. The Luby transform code proposed in [10], [36] was designed for filling packet erasures wiped out by the BEC channel, where the transmitted data is not contaminated by channel noise or fading. The Luby transform decoding process recovers source packets by using the eXclusive OR (XOR) operations between a certain number of received packets, as dictated by the generator matrix $\mathbf{G}$ of the Luby transform code. When the data is transmitted across the wireless Internet channel, where the data is affected by erasures, noise and fading events, the erasure decoding of the Luby transform code will impose severe error propagation between the LT-decoded packets, when a single contaminated received packet is used during the erasure decoding process, as portrayed in Fig. 4.2. As seen in Fig. 4.2, when a single contaminated received packet S_1 is used by the erasure decoding process, the contaminated part of this LT-packet indicated by the shaded part of the packet is propagated to all of the decoded packets. After the 5^{th} decoding cycle of Fig. 4.2, all of the LT-decoded packets are contaminated by this erroneous packet-segment at the corresponding symbol positions. For the sake of preventing this error propagation, in [64] Luby *et al.* proposed a soft-bit decoding method based on message passing between the encoded symbols and source symbols of the Raptor code, where the symbols are groups of q-bits [64]. The BER performance and practical application of this decoding algorithm is limited for the following reasons [64]:

- Consider the pairs of decoded symbols y_1 and y_2 as well as the triplet of received symbols x_1, x_2, x_3 as portrayed in [64, Fig. 4], where x_1 and x_3 have a degree of one, while x_2 has a degree of two. Assume furthermore that x_2 is connected to both y_1, y_2, while x_1 and x_3 are connected to y_1 and y_2, respectively. If the above configuration

does not exist and not all source symbols are recovered, then the decoding operation will fail because in this case the message passing algorithm cannot be implemented as suggested in [64].

- As the length of the q-bit symbols increases, the complexity of calculating the symbol probability in a large q-ary field increases exponentially.

For the sake of finding out a flexible decoding method for the Luby transform code, we managed to apply the message passing algorithm, which is used by the LDPC code, for the Luby transform code. This idea is developed as follows. By observing the encoding and decoding processes of both Luby transform codes and quasi-regular LDPC codes, we can list their similarities and differences:

- **Similarities:**
  - Both Luby transform codes and quasi-regular LDPC codes use the generator matrix $\mathbf{G}$ to generate the encoded packets and bits, respectively.
  - Both Luby transform codes and LDPC codes are represented by the *Tanner graph* and are decoded by the Belief Propagation (BP) algorithm.
- **Differences:**
  - When designing LDPC codes, the Parity Check Matrix (PCM) $\mathbf{H}$ is typically formed by randomly creating the parity checks for the rows and columns having quasi-regular weights. In other words, the weights of the rows and columns of $\mathbf{H}$ are similar. The generator matrix $\mathbf{G}$ may be calculated from $\mathbf{H}$ by exploiting that $\mathbf{G} \cdot \mathbf{H}^T = \mathbf{0}$ where $\mathbf{H}^T$ is the transpose of $\mathbf{H}$.
  - By contrast, when designing Luby transform codes, the generator matrix may be created based on the Robust Soliton Degree Distribution (RSDD) [10].
  - The LDPC code decodes based on $\mathbf{H}$ use the BP decoding method combined with the message passing algorithm, while Luby transform codes are decoded based on $\mathbf{G}$ and employ the BP decoding method combined with the erasure filling algorithm.
  - The LDPC encoding process is a bit-by-bit process, while the Luby transform encoding process is a packet-by-packet process.

Based on the above similarities and differences, the idea of calculating $\mathbf{H}$ from $\mathbf{G}$ for Luby transform codes was proposed in [25], by exploiting that we have $\mathbf{G} \cdot \mathbf{H}^T = \mathbf{0}$ for the sake of applying the BP decoding method combined with the message passing algorithm. This idea is described as follows. Rateless non-systematic LT codes were designed in [10], [11], [36] [43], where the original source information packets and the parity packets can not be directly identified. For the sake of soft-decoding LT codes by the classic Belief Propagation algorithm of LDPC codes [7], [32], [47] and [65], we have to specify their Parity Check Matrix (PCM) $\mathbf{H}$. The Generator Matrix (GM) $\mathbf{G}$ of non-systematic LT codes might be rewritten as follows [66]:

$$\mathbf{G}_{(K\times N)} = [(\mathbf{B}^T.(\mathbf{A}^T)^{-1})_{K\times M}; \mathbf{I}_{(K\times K)}], \tag{4.1}$$

where $\mathbf{A}$ and $\mathbf{B}$ are the component matrices of the PCM $\mathbf{H}$, while the superscript T represents the transpose matrices. Given $\mathbf{A}$ and $\mathbf{B}$, the PCM $\mathbf{H}$ may be expressed as follows:

$$\mathbf{H}_{(M\times N} = [\mathbf{A}_{(M\times M)}; \mathbf{B}_{(M\times K)}]. \tag{4.2}$$

From (4.1) and (4.2) we can calculate the PCM of the non-systematic LT code by the coresponding inverse process as follows:

- We commence by finding out a non-singular matrix $\mathbf{A}$ having the size of $(K \times K)$ elements within the GM $\mathbf{G}$. If there is no such matrix $\mathbf{A}$ in $\mathbf{G}$, we failed to construct the PCM $\mathbf{H}$ of the non-systematic LT code.
- If existing a non-singular matrix $\mathbf{A}$ in $\mathbf{G}$. Re-arranging the generator $\mathbf{G}$ and we have $\mathbf{G}$=[$\mathbf{A}_{(K\times K)}$;$\mathbf{B}_{(K\times M)}$] . Calculating the product of $[\mathbf{B}^T.(\mathbf{A}^T)^{-1}]_{M\times K}$.
- The PCM $\mathbf{H}$ is calculated as follows:

$$\mathbf{H}_{(M\times N)} = [(\mathbf{B}^T.(\mathbf{A}^T)^{-1})_{M\times K}; \mathbf{I}_{(M\times M)}]. \tag{4.3}$$

Unfortunately, the resultant PCM $\mathbf{H}$ calculated from the GM of the non-systematic LT code contains many zero-columns. Therefore, the achievable performance of non-systematic LT codes using belief-propagation based soft-bit decoding method is poor. Another reason for the poor soft decoding performance of non-systematic LT codes is that we cannot readily find the optimal PCM matrix for non-systematic LT codes. Given the PCM $\mathbf{H}$, we soft-decode the bits of the LT coded packets received and as always, this process will impose error propagation to other source packets as examplified in Fig. 4.2 and

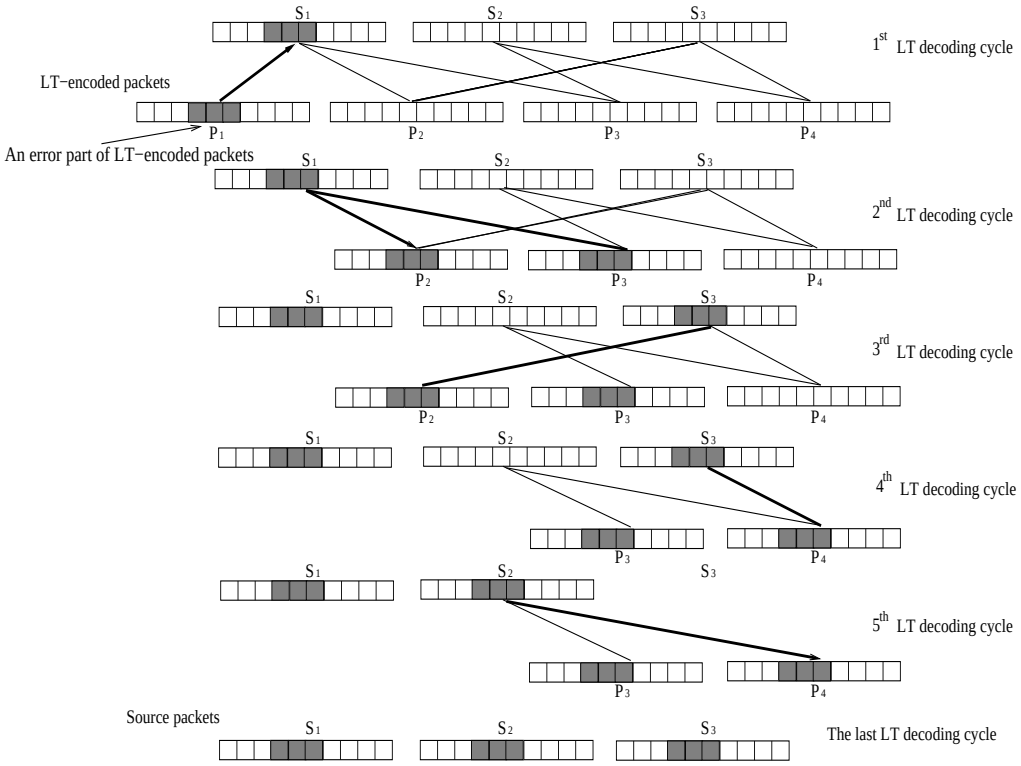


Figure 4.2: An example of error propagation in the LT hard-decoding process, when the LT decoder receives packets affected by the channel error

detailed in the next section. A different LT soft-decoding method was proposed in [64], which also uses the message passing method based on the GM $\mathbf{G}$, but the decoding process is suspended if it can not find out the set of packets that satisfy the specific configuration as analysed in [64, Fig. 4] and in [27].

4.2.2 Systematic Luby Transform Codes and Their Soft-Decoding

LT codes were originally designed for hard-decoding [10] in the context of the BEC. The error propagation phenomenon of the LT decoding process is portrayed in Fig. 4.2. This error propagation is described as follows. Assuming that the LT decoder need to recover three source packets S_1, S_2, S_3 from four received LT packets $(P_1, \cdots, P_4)$, as seen in Fig. 4.2 and the received LT degree-one packet P_1 contains three erroneous bits contaminated by the noise, which are the fourth, fifth, sixth bits. At the first LT decoding cycle, the bits of the source packet S_1 having a connection to P_1 are foisted the corresponding bit's values of the received LT packet P_1 and then this connection is erased. At

this step, three erroneous bits plotted by grey square in Fig. 4.2 are propagated to three corresponding position bits of S_1. At the second LT decoding cycle, the LT decoder finds in the rest received LT packets P_2, P_3 and P_4 which packet has a connection to the source packet S_1. As seen in Fig. 4.2, there are only P_2 and P_3 having connections to S_1. The LT decoder implements XOR operation S_1 with P_2 and S_1 with P_3, then foists these results to P_2 and P_3 and clears the connections from S_1 to P_2 and P_3. Again, the LT decoder propagates erroneous bits from S_1 to P_2 and P_3. Continuosly, after 5 decoding cycles the LT decoding process propagates erroneous bits from single received contaminated LT packet to all the source LT packets, as seen in Fig. 4.2. For the sake of avoiding this detrimental inter-packet error propagation effect, they have been combined with various FEC codes [20] [67]. These combined schemes substantially mitigated the effects of error propagation. However, the attainable performance improvements of these schemes were still limited owing to the employment of hard-decision aided LT decoding. For the sake of circumventing this deficiency, we introduce the novel concept of soft LT decoding based on the classic belief-propagation technique applying Tanner Graphs. We commence our discourse by introducing the concept of 'single-bit packets'. The LT codes having larger packets will be considered as the generalized scenario. The soft LT decoding process is based on the classic concept of the LDPC decoding. Given the generator matrix $\mathbf{G}$ of the LT code, we calculate the Parity Check Matrix (PCM) $\mathbf{H}$ of the LT code similarly to that of a classic LDPC code, namely by dividing the LT code's generator matrix into two matrices, where $\mathbf{A}$ and $\mathbf{B}$ have a size of $(K \times K)$ and $(K \times M)$, respectively. We choose the non-singular matrix $\mathbf{A}$ based on the conventional LT decoding process. Then following from (4.3) the PCM is calculated as follows:

$$\mathbf{H}_{(M\times N)} = [(\mathbf{B}^T \cdot (\mathbf{A}^T)^{-1})_{(M\times K)} | \mathbf{I}_{(M\times M)}]. \tag{4.4}$$

The LT decoding process may be implemented as follows. An LT code's PCM can be represented by a classic Tanner graph [67]. To elaborate a little further, the filled circles and the filled squares of Fig. 4.3 represent the LT variable nodes and the LT check nodes, respectively, while the horizontal lines connected to the variable nodes represent the intrinsic information provided by the channel's output. Let us assume that the circular node at the top of Fig. 4.3 represents the K^{th} variable node in the block of N number of single-bit LT encoded packets, which is also termed as the root or information node [68]. The root node receives information from the check nodes it is connected to at the level seen below it in Fig. 4.3 and those check nodes also receive information from the variable nodes they

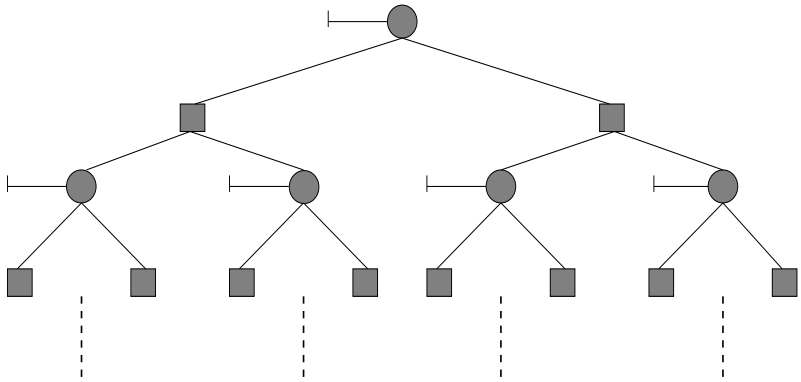


Figure 4.3: A tree-based Tanner-graph representation of LT code

are connected to at the next level down, etc. The dotted lines in Fig. 4.3 indicate that the above process is repeated further by expanding the tree. The number of connections associated with a variable node of the LT code - excluding the line representing the intrinsic information - indicates the column weight of this particular message node, while the number of connections associated with an LT check node represents the corresponding row weight. The column weight and row weight of the LT code's PCM are related to the degree distribution of LT packets.

The LT decoding process is implemented in the same way as the classic LDPC decoding procedure. Initially, the LT decoder's soft values are set to a value corresponding to the demodulator's soft output. The decoder's soft values of $R^a_{i,j}$ and $Q^a_{i,j}$ which denote the LLRs passed from the check nodes to the variable nodes and vice versa are then iteratively updated after each decoding iteration as follows [69]:

$$tanh(\frac{R_{i,j}}{2}) = \prod_{n\in\{C_i\},n\neq i} tanh(\frac{Q_{n,j}}{2}), \tag{4.5}$$

where we have

$$tanh(x/2) = \frac{e^x - 1}{e^x + 1}. \tag{4.6}$$

After each iteration, the LT decoder outputs its tentative hard-decision and checks, whether the product of the corresponding codeword and the transpose of the PCM $\mathbf{H}$ is equal to zero according to ($\mathbf{C} \cdot \mathbf{H}^T = \mathbf{0}$) i.e whether a legitimate codeword was produced. If not, the LT decoding process will be continued in an iterative fashion, until the output codeword becomes legitimate or the maximum affordable number of iterations is exhausted. For the sake of improving the performance of LT codes at high E_b/N_0 values, we invoke hard-decoding after the last LT decoding cycle in order to erase the low-confidence LT

packets, namely those, which have low Logarithm Likelihood Ratios (LLRs). We employ the low-complexity packet-reliability evaluation technique of Section 3.3 based on evaluating the average LLR of the packet. However, the soft LT decoding process based on the LT code's PCM exhibits some deficiencies owing to the following two reasons:

- the LT code's PCM contains many zero-columns because of absenting the identity information component part in $\mathbf{G}$ of LT codes, which degrades the performance of the LT code, when using the above mentioned soft decoding process.
- The conventional non-systematic LT code will impose error propagation, when using the above-mentioned hard-decoding process for the sake of recovering the original information packets from the erroneously decoded LT-encoded packets, as seen in Fig.4.2.

Hence, for the sake of improving the LT code's performance in hostile wireless channels, we specifically design the LT code's degree distribution by expanding its generator matrix with the aid of a unity matrix having a size of $(K \times K)$ elements, which results in a SLT

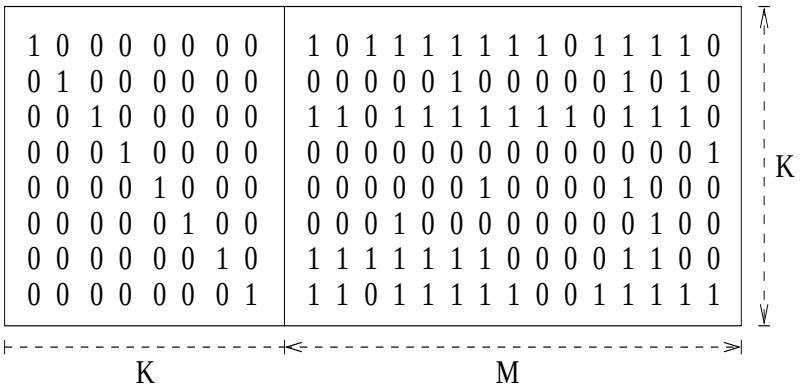


Figure 4.4: The systematic LT code's generator matrix

code. The resultant relationship of the SLT generator matrix $\mathbf{G}$ and the PCM $\mathbf{H}$ may be stated as follows:

- If we have a generator matrix $\{\mathbf{G}_{K\times N} = [\mathbf{I}_{K\times K}|\mathbf{A}_{K\times M}]\}$, where $\mathbf{I}$ is an identity matrix having a size of $(K \times K)$ and $\mathbf{A}$ is a non-singular matrix having a size of $(K \times M)$, then the PCM obeys the construction of $\{\mathbf{H} = [\mathbf{A}^T|\mathbf{I}']\}$, where $\mathbf{A}^T$ is the transpose of $\mathbf{A}$ and $\mathbf{I}'$ is an identity matrix having a size of $(M \times M)$ [7], where $(N = K + M)$ is the number of columns in $\mathbf{G}$ and K is the number of rows in $\mathbf{G}$.

4.2.3 EXIT-Chart Analysis of SLT Codes

Extrinsic Information Transfer (EXIT) charts [31] [70] were proposed by ten Brink and his collaborators. As mentioned in Section 4.2, SLT codes may be decoded by belief propagation. Hence, we can analyse SLT codes based on EXIT charts [31] [71]. Since SLT codes have a specific PCM construction, which contains the check bits in the corresponding rows and the message bits in the corresponding columns, while the second part of the PCM is constituted by a unity matrix having a size of $(M \times M)$ elements containing the check bits in both its rows and columns. Hence, we have to re-define the passing of the LLR messages during the SLT decoding process as follows:

- The LLR messages are passed between the SLT message nodes and the SLT check nodes.

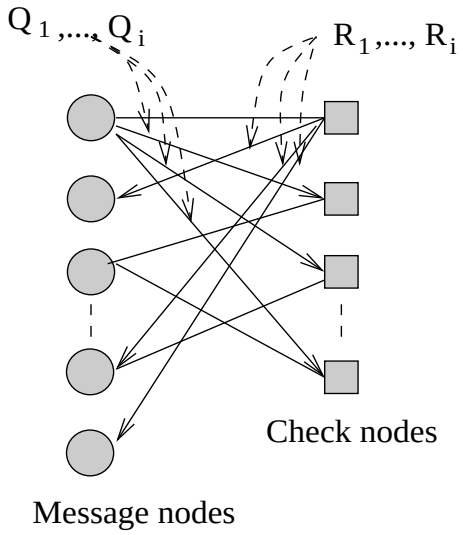


Figure 4.5: The LLR message passing between message nodes and parity nodes of SLT codes.

Let d_m represent the degree of the message nodes, d the degree of the encoded parity packets and d_c the degree of the check nodes. Furthermore, let K denote the number of message nodes and N the number of variable nodes. Then the number of check nodes is given by M=(N-K). The degree distribution $D(d)$ of the encoded parity packets was

defined in Section 3.1.1 as follows:

$$D(d) = \begin{cases} 0 & d = 0, \\ \frac{1+S+\nu}{Z \cdot K} & \text{if } d = 1, \\ \frac{1}{Z}\left[\frac{1}{d(d-1)} + \frac{S}{K} \cdot \frac{1}{d}\right] & \text{if } 2 \leq d < \frac{K}{S}, \\ \frac{S}{Z \cdot K}\left[\log\frac{S}{\delta} + \frac{1}{(\frac{K}{S}-1)}\right] & \text{if } d = \frac{K}{S}, \\ \frac{1}{Z}\left[\frac{1}{d \cdot (d-1)}\right] & \text{for } K > d > \frac{K}{S} \end{cases} . \tag{4.7}$$

where S, ν were defined in the context of (3.5) of Section 3.1.1 [20], We assume that the LT encoding process of [20] using the ideal random integer generator is employed. Then the degree distribution of the message nodes $D(d_m)$ is defined as:

$$D[d_m(\overline{d_c})] \quad = \quad r,$$

where $r = \frac{K}{N}$ is the code rate of the SLT code and $d_c = (d + 1)$. Let us now consider the messages passed between the SLT message nodes and the check nodes in Fig. 4.5, when using the LLR representation of the messages. Let Q as well as R denote the LLR information passed from the message nodes to the check nodes and that passed from the check nodes to the message nodes, respectively, as seen in Fig. 4.5. As seen in Fig. 4.5, the extrinsic LLR information passed from the check nodes to the message nodes is defined by:

$$Q = \sum_{i=0}^{d_m-1} R_i, \tag{4.8}$$

where the initial soft channel-output message associated with the variable nodes is given by ($R_0 = \frac{4}{N_0} \cdot y$) and y denotes the output of an Additive White Gaussian Noise (AWGN) channel. Furthermore, as seen in Fig. 4.5 $\{R_i$; i= 1 ... $(d_v - 1)\}$ represents the LLR information arriving from the check nodes to the message nodes, except from that particular check node to which the LLR information message Q was sent. The extrinsic LLR information R passed from the check nodes to the message nodes is defined as:

$$tanh(\frac{R}{2}) = \prod_{i=1}^{d_c-1} tanh(\frac{Q_i}{2}), \tag{4.9}$$

where $\{Q_i$; i= 1,..., (d_c-1)$\}$ represents the LLR information arriving from the message nodes, except from that particular message node to which the LLR information message R was sent. Let m_R, m_{R_0} and m_Q denote the mean of R, R_0 and Q. Then, from (4.8) we have:

$$m_Q^{(l)} = m_{R_0} + (d_m - 1) \cdot m_R^{(l-1)}, \tag{4.10}$$

where l is the $l'th$ iteration and ($m_{R_0} = 4 \cdot \frac{E}{N_0}$), while E is the transmitted bit energy, while $\frac{N_0}{2}$ is the one-side power spectral density of the noise. We assume that R is Gaussian distributed. Hence, m_R is can be updated according to:

$$m_R^{(l)} = J^{-1}\left[I(X; R^{(l)})\right], \tag{4.11}$$

where $J(m_R)$ is defined as follows:

$$\begin{aligned} J(m_R) = I(X; R^l) &= \\ &= \int \frac{1}{\sqrt{4\pi m_R}} e^{-\frac{(l-m_R)}{4m_R}} (1 - log_2(1 + e^{-}l))dl. \end{aligned} \tag{4.12}$$

The mutual information was calculated in [72] as follows:

$$I(X; R) = \frac{1}{ln2} \sum_{i=1}^{\infty} \frac{1}{2i(2i-1)} [E(T_Q^{2i})]^{d_c - 1}, \tag{4.13}$$

where [72]:

$$\begin{aligned} \phi_i(m_Q) &= E(T_Q^{2i}) \\ &= \int_{-1}^{+1} \frac{2t^{(2i)}}{(1-t^2)\sqrt{4\pi m_Q}} e^{-\frac{(\ln(\frac{1+t}{1-t}) - m_Q)^2}{4m_Q}} dt. \end{aligned} \tag{4.14}$$

Finally, we arrive at the required update formulae for the means m_Q and m_R as follows:

$$m_Q^{(l)} = m_{R_0} + (d_m - 1) m_R^{(l-1)} \tag{4.15}$$

$$m_R^{(l)} = J^{-1}(\frac{1}{ln2} \sum_{i=1}^{\infty} \frac{1}{2i(2i-1)} [E(T_{m_{Ql}}^{2i})]^{d_c - 1}. \tag{4.16}$$

Based on (4.15) we arrive at the variable node's EXIT function:

$$\begin{aligned} I_{E_v} = I(X; Q) &= I(X; R_0, R_1, \ldots, R_{d_m - 1}) \\ &= f\left[I(X; R_0), I(X; R)\right] = f(I_{ch}, I_{A_v}). \end{aligned} \tag{4.17}$$

Similarly, from (4.13) we derive the check node's EXIT function as follows [72]:

$$\begin{aligned} I_{E_c} = I(X; R) &= I(X; Q_1, \ldots, Q_{d_c - 1}) \\ &= f(I(X; Q)) = f(I_{A_c}) = \frac{1}{\ln 2} \sum_{i=1}^{\infty} \frac{1}{(2i-1)(2i)} [\phi_i(J^{-1}(I_{A_c})]^{d_c - 1}, \end{aligned} \tag{4.18}$$

where $I_{ch} = I(X; R_0)$ is the average channel output information, $I_{A_v} = I(X; R)$ is the average a priori information at the input of the message node decoder and I_{E_v} is the average extrinsic information at the output of the message node decoder.

The SLT code has the degree distributions $D(d_m)$ and $D(d_c)$ formulated in (4.7) and (4.8). Therefore, the EXIT functions of the message and check nodes are given in (4.17) and (4.18), which depend on the degrees of the message nodes and check nodes, respectively. Again, we assumed that the LT encoding process of [20] using the ideal random integer generator is employed. Hence, all the LT message nodes have the same degree, namely a degree of $(\overline{d_c} - 1)$. Finally, we arrive at the EXIT function of the message nodes and the check nodes for the SLT code expressed in the following form:

- Message node EXIT function [72]:

$$\begin{aligned} I_{E_v} &= f(I_{ch}, I_{A_v}) \\ &= J\left[J^{-1}(I_{ch} + (j-1)J^{-1}(I_{A_v})\right]. \end{aligned} \tag{4.19}$$

- Check node EXIT function [72]:

$$\begin{aligned} I_{E_c} &= f(I_{A_c}) \\ &= \sum_{j=2}^{d_c} d_{c_j} I_{E_{c_j}} = \sum_{j=2}^{d_{c_{max}}} d_c(j) \cdot \frac{1}{ln2} \sum_{i=1}^{\infty} (\frac{1}{2i(2i-1)})[\phi_i(J^{-1}(I_{A_c})]^{j-1}. \end{aligned} \tag{4.20}$$

The EXIT chart of the SLT code is portrayed in Fig. 4.6.

The associated parameters are:

- The check bits obey the same degree distribution as the LT packets, namely the Improved Robust Soliton Degree Distribution (IRSDD) proposed of Section 3.1.1 in [20].
- All of the message bits have a degree of one.
- A total of 500 LT-encoded packets are transmitted and each of them has 1,000 bits. The rate of the SLT code was set to $r = \frac{1}{3}$.

The SLT code's message-node-related EXIT curve recorded at $E_b/N_0 = 1$ dB intersects the check-node-related curve relatively close to the point of perfect convergence, namely the (1,1) point of Fig. 4.6, which was made more visible by expanding the top right corner of Fig. 4.6 in Fig. 4.7. The point of intersection is marked by a bold dot in Fig. 4.7. Owing to the intersection between the message-node's EXIT curve and the check-node's EXIT curve, the SLT decoder is unable to reach an infinitesimally low BER at $E_b/N_0 = 1$ dB. By contrast, observe in Fig. 4.7 that the SLT decoder becomes capable of approaching the (1,1) point at $E_b/N_0 = 6$ dB after 10 iterations.

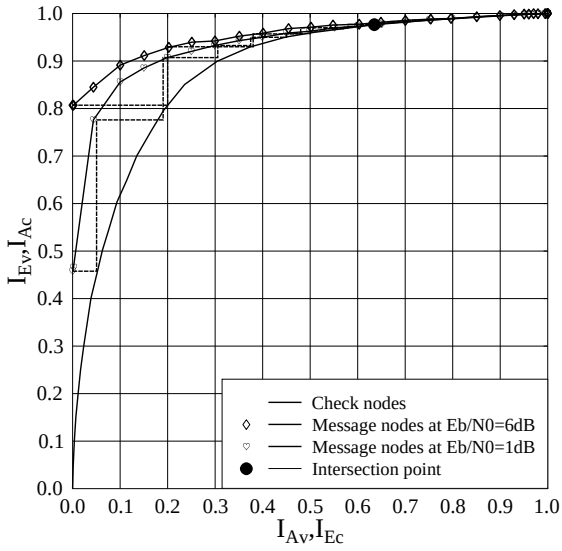


Figure 4.6: EXIT chart of the SLT(1,000, 3,000) code.

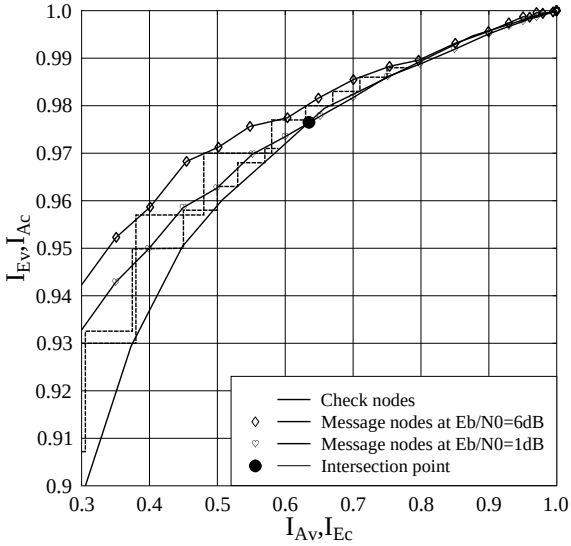


Figure 4.7: Expanded view of the SLT(1,000, 3,000) code's EXIT chart.

4.2.4 Simulation Results

Erasure probability P_e	0.0; 0.1
LT code parameters in Eq.(4.24)	δ= 0.5; c= 0.1
The number of source SLT packets	500
SLT packet size	1000 bits
SLT code rates r	1/3
Modulation	QPSK
Noise channel	AWGN
Number of the SLT code's iteration	0, 2, 4, 6

Table 4.1: System parameters of SLT codes.

Fig. 4.8 shows the attainable BER performance of the SLT(1,000, 3,000) code using the message-passing decoding technique, when communicating over the AWGN channel, having an erasure probability of P_E= 0. Fig. 4.8 may be contrasted to Fig. 4.9, where transmission over the Wireless Internet associated with different E_b/N_0 values and $P_e = 0.1$ is considered. Fig. 4.10 shows the performance of the SLT code for transmission over the BEC having different erasure probabilities P_e, where the abscissa axis was scaled in terms of the values of $(1 - P_e)$. Again, the Improved Robust Soliton Degree Distribution (IRSDD) of (3.6) in Section 3.1.1 was used for the variable nodes of the SLT(1,000, 3,000) code characterized in Fig. 4.8, Fig. 4.9 and Fig. 4.10. The degree distribution parameters of c= 0.1 and δ = 0.5 were used of (3.6) in Section 3.1.1 as defined in [20]. We can see in Fig. 4.8 that for P_e= 0 the BER becomes as low as 10^{-5} at E_b/N_0= 3.5 dB. By contrast, when the erasure probability is P_e= 0.1, the decoder reaches BER= 10^{-5} at E_b/N_0= 4.5 dB. The proposed scheme has the potential of maintaining a lower BER at a given E_b/N_0 and capable of tolerating higher P_e values than the system of [20], because the hard-bit LT decoding used in [20] is very sensitive with the contaminated received packets and it can cause error propagation from a single contaminated received packet, as exampled in Fig. 4.2. When we increase the size of the SLT codes from SLT(1,000, 3,000) to SLT(10,000, 30,000), their performance is improved, as seen by contrasting in Fig. 4.11 to Fig. 4.8, because the length of the SLT codeword is increased. More specifically, a BER of 10^{-5} may be attained at $E_b/N_0 = 3.0$ dB after 6 iterations of the SLT decoder. Finally, observe in Fig. 4.10 that the SLT decoder is capable of maintaining an infinitesimally low

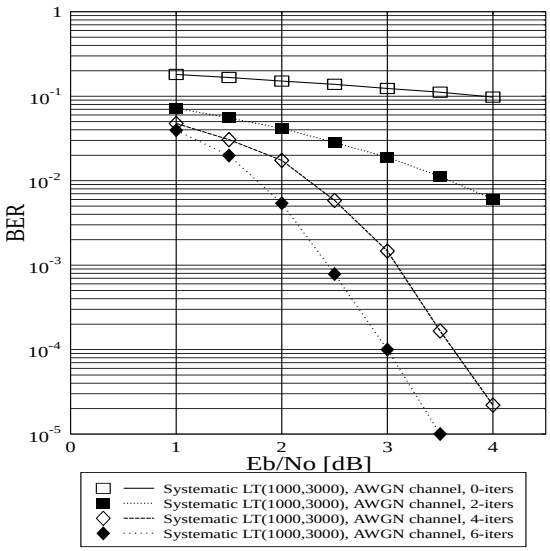


Figure 4.8: BER versus Eb/N0 performance of the SLT(1,000, 3,000) code of Table 4.1 in AWGN channels using BPSK modulation and no erasures.

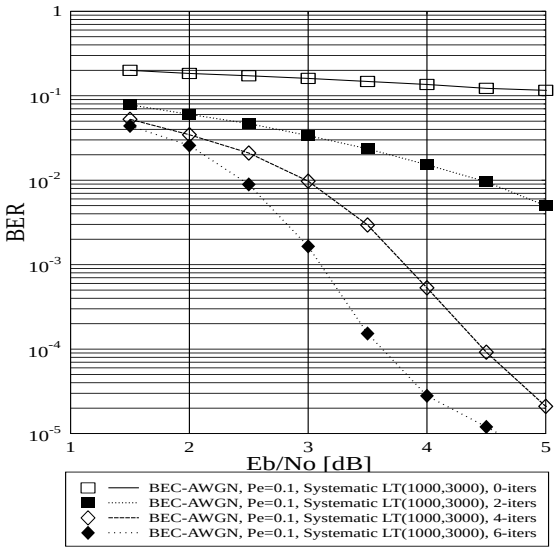


Figure 4.9: BER versus E_b/N_0 performance of the SLT(1,000, 3,000) code of Table 4.1 for transmission over the AWGN-contaminated BEC associated with ($P_e = 0.1$).

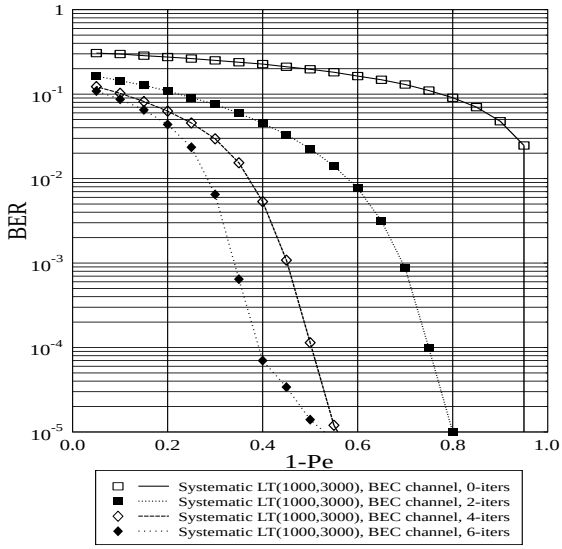


Figure 4.10: BER versus $1 - P_e$ performance of the SLT(1,000, 3,000) code of Table 4.1 in the noiseless BEC channel.

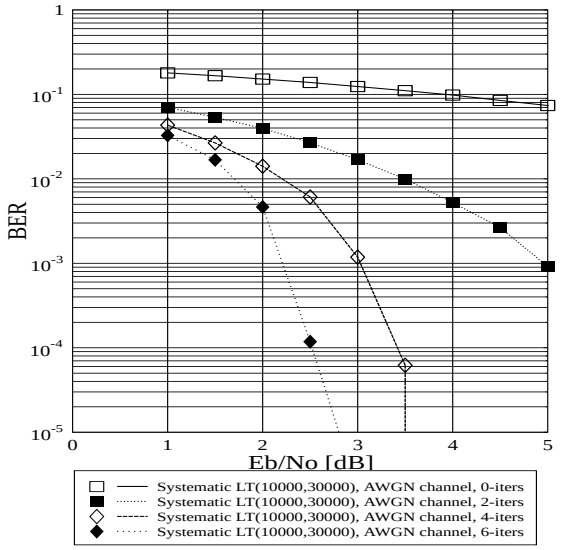


Figure 4.11: BER versus E_b/N_0 performance of the SLT(10,000, 30,000) code of Table 4.1 for transmission over the AWGN-contaminated BEC associated with $P_e = 0$.

BER for erasure probabilities as high as $P_e = 0.4$ in the BEC.

4.2.5 Conclusions

The concept of SLT codes was introduced with the aid of appropriately modifying the RSDD of Section 3.1.1 [10] and [20]. The SLT coding concept of Section 4.1 facilitated the employment of the classic belief-propagation based soft decoding of the resultant SLT codes. The main benefit of the proposed scheme is that it is capable of mitigating the effects of catastrophic error propagation across the LT-encoded packets. For the sake of improving the performance of the SLT codes in the next section, we consider their degree distribution and design a random integer generator for coining the specific degree of the individual LT packets.

4.3 An Improved Degree Distribution and A Random Integer Generator for SLT Codes

4.3.1 Introduction

When designing LT codes, there are three important factors, which determine the attainable performance of the LT code, namely the degree distribution, the integer random generator used for coining a particular packet's degree and the total number of source packets to be transmitted [10], [11], [36], [25]. However, as mentioned in [10], [11] and [36] the degree distribution design is the most influential factor. The original distribution proposed in [10], [11] and [36] for SLT codes is no longer suitable, when using soft bit decoding. Hence, the novel contribution of this section is that of designing an improved degree distribution for SLT codes. This distribution is referred to as the Truncated Degree Distribution (TDD), which will be outlined in Section 4.3.2. Furthermore, we also introduce a novel random integer generator for determining the specific degree of each packet during the LT encoding process. This random generator is termed as the conditional random integer generator, which will be detailed in Section 4.3.3. In Section 4.1 the soft bit based SLT is discussed, while in Section 4.3.4 its performance is analysed by using EXIT charts. Finally, in Section 4.3.5 the achievable BER performance is detailed and our conclusions are offered in Section 4.3.6.

4.3.2 The Truncated Degree Distribution Designed for The SLT's Parity Packets

For the sake of finding an improved degree distribution, we commence our discourse by revisiting the original RSDD $\mu(d)$ of Section 2.2.5.1 [10], [20]. The RSDD $\mu(d)$ is composed of two parts, namely $\rho(d)$ and $\tau(d)$ formulated as follows [10]:

$$\rho(d) = \begin{cases} 1/K & \text{for } d = 1, \\ \frac{1}{d(d-1)} & \text{for } d = 2, 3, ..., K, \end{cases} \tag{4.21}$$

and

$$\tau(d) = \begin{cases} \frac{S}{K}\frac{1}{d} & \text{for d=1, 2, ..., } \frac{K}{S} - 1, \\ \frac{S}{K}\log(\frac{S}{\delta}) & \text{for } d = \frac{K}{S}, \\ 0 & \text{for } d > \frac{K}{S}, \end{cases} \tag{4.22}$$

which are combined as:

$$\mu(d) = \frac{\rho(d) + \tau(d)}{Z'}, \tag{4.23}$$

where K, S, d are the number of input packets, the number of packets having a degree-one and the degree of packets, respectively. These distributions were plotted in Fig. 2.13 of Section 2.2.5.1. Finally, we have $Z' = \sum_d [\rho(d) + \tau(d)]$. The number of degree-one packets generated by this distribution is [10] $S = \left\lceil c\sqrt{K}\log_e(K/\delta) \right\rceil$, where c and δ constitute the tuneable parameters of the distribution. There are two conditions, which have to be satisfied by all input packets for them to be recovered from the received LT-encoded packets. Firstly, the number of LT-encoded packets N has to satisfy [10], where $N \geq K + 2S\log_e(\frac{S}{\delta})$ and secondly, the number of packets having the highest degree must obey $d \geq \frac{K}{S}$ [10], [36]. These are the prerequisites for designing the original LT code of Section 2.2.5.1 [36]. When we design the SLT code of Section 4.2.2 [25], these are still sufficient conditions for the decodability of SLT codes having a code rate lower than $R = \frac{1}{2+\epsilon}$, where $\epsilon = 2\log_e(\frac{S}{\delta})$ is the relative packet-level overhead of the original LT codes. However, these conditions are no longer sufficient for SLT codes having a code rate higher than $R = \frac{1}{2+\epsilon}$, because the density of both the parity and generator matrices of the SLT codes are low. For the sake of improving the density distribution of both the parity and generator matrices of the SLT codes, we re-define the degree distribution used for generating the parity check

part of SLT codes as follows [26], [27]:

$$\Omega(d) = \mu(d,\gamma,\nu) = \begin{cases} 0 & \text{for } d = 1, \\ \frac{1}{Z}\left[1 + \frac{S}{k} + \nu\right] & \text{for } d = \gamma, \\ \frac{\gamma}{Z}\left[\frac{1}{d\cdot(\frac{d}{\gamma}-1)} + \frac{S}{k}\frac{1}{d}\right] & \text{for } d = 2\gamma, 3\gamma, \cdots, \frac{k\cdot\gamma}{S} - 1, \\ \frac{S}{Z\cdot k}\left[\log_e(\frac{S}{\delta}) + \frac{1}{(\frac{k}{S}-1)}\right] & \text{for } d = \frac{\gamma\cdot k}{S}, \\ \frac{\gamma}{Z}\left[\frac{1}{d\cdot(\frac{d}{\gamma}-1)}\right] & \text{for } k > d > \frac{\gamma\cdot k}{S}. \end{cases} \tag{4.24}$$

where K is the total number of original information source packets contributing to the SLT code constituted by m number of (n,k) codes, S is the number of packets having a specific degree γ, where the variables satisfy the condition of [36] $\left[S = c\sqrt{K}\log_e(K/\delta)\right]$ and γ is an integer number higher than unity. Furthermore, ν represents the extra fraction of redundant packets required for the Improved Robust Soliton Degree Distribution (IRSDD) of Section 3.1.1 [20] to ensure decodability. Still referring to (4.24), we have $\{Z = \sum_d [(\rho(d) + \tau(d)) + \nu]\}$. Maintaining the maximum packet degree of $\left(d_{max} = \frac{\gamma\cdot K}{S}\right)$ ensures that all original input packets will be represented in the set of SLT-encoded parity packets at least γ times [10], [11], [36]. Hence, the resultant packet degree distribution may be referred to as a truncated distribution having a maximum abscissa value of $\left(d_{max} = \frac{\gamma\cdot K}{S}\right)$ and an ordinate value step size of γ in the fictitious degree-histogram not included here. The probability that upon generating a new SLT-encoded parity

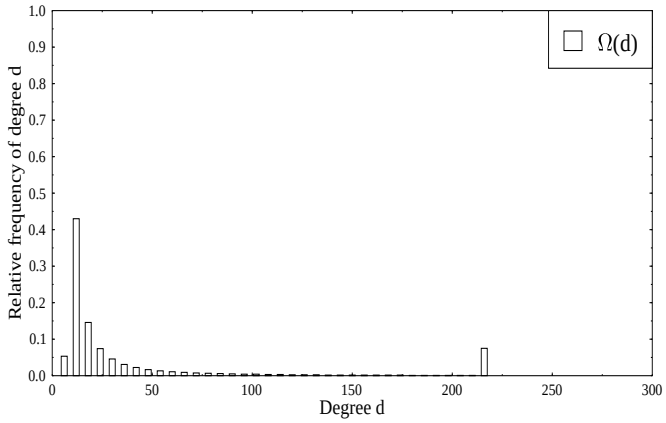


Figure 4.12: An example of the truncated RSDD designed for the SLT-parity packets having parameters of $K = 10,000, \delta = 0.5$, $c = 0.1$, $\gamma = 2$.

packet having a degree of γ, $(K - \gamma)$ number of SLT input information packets still remain

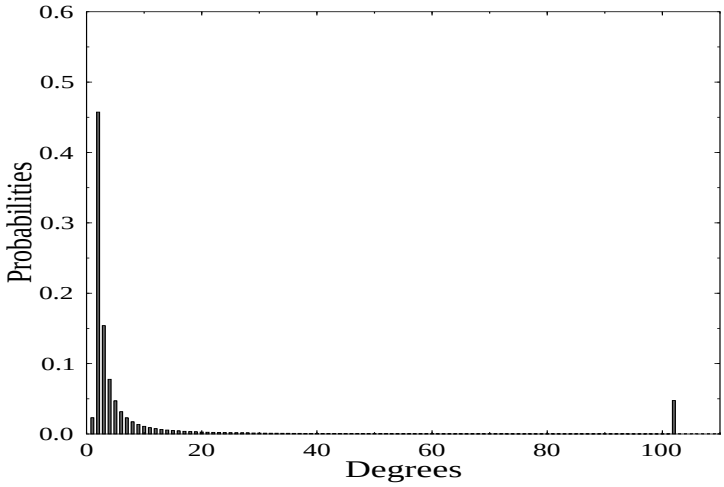


Figure 4.13: The original RSDD $\mu(d)$ of (2.16) for the case of K=10,000, c=0.1, δ=0.5

unrepresented up until the instant of generating the SLT-encoded parity packet in the encoder's output buffer was termed as the Degree Release Probability (DRP) [10], which was formulated as [11]:

- $p(\gamma, K-\gamma) = 1$;
- $p(d, L) = \frac{d(d-1)\cdot L\cdot \Pi_{j=0}^{d-1}(K-(L+1)-j)}{\Pi_{j=0}^{d-1}(K-j)}$ for $d = 2\gamma, 3\gamma, \cdots, \frac{\gamma\cdot K}{S}$ and $L = K-d+1, \cdots, \gamma$;
- for all other d and L values we have $p(d, L) = 0$,

where L is the number of still unrepresented input information packets, when a new SLT-encoded parity packet having a degree d is generated. Hence, given the degree distribution $\Omega(d)$ of (4.24), the probability that there are still L unrepresented input packets, when a new SLT-encoded parity packet is generated, is given by $[P(d, L) = \Omega(d) \cdot p(d, L)]$, where $p(d, L)$ is the DRP of an SLT-encoded parity packet having a degree of d.
Finally, the average degree of the SLT-encoded packets is calculated as follows [26], [27]:

$$D = \sum_{d}^{K} \frac{d \cdot (\rho(d) + \tau(d) + \nu(d))}{Z} + 1. \tag{4.25}$$

It can be readily shown that this average SLT-encoded packet degree gives the average row weight in the parity check matrix of the SLT codes designed in [25]. Having derived a novel degree distribution for SLT codes, in the next section we contrive a novel conditional

integer random generator, which is used for determining the specific degree of each SLT-source packet during the SLT encoding process.

4.3.3 Conditional Random Integer Generator Designed for The Degree Distribution of The SLT's Source Packets

For the sake of random equally generating the integer index for coining the specific source SLT packet in the SLT encoding process, two types of random integer generators have been proposed in [22], which are the Linear Congruential Random Integer Generator (LCRIG) and the Bit Swapping Random Integer Generator (BSRIG). Given the current integer I_j and the parameters C and M, the next integer generated by the LCRIG for coining the index of the next chosen SLT-source packet is given as follows [22]:

$$I_{j+1} = (aI_j + C) \textbf{ mod } M, \tag{4.26}$$

where M is the basis of the modulo function used, while a and C are positive integers referred to as the multiplier and the increment, respectively.
There are two types of the BSRIGs [22]. The first type suggests that the bits representing the integer $(I_{n-j} + I_{n-k})$ **mod** 2^b are rotated i.e. circularly shifted by r bit-positions according to:

$$I_n = [(I_{n-j} + I_{n-k}) \textbf{ mod } 2^b] \textbf{ rot } r, \tag{4.27}$$

where '**rot**' denotes the rotation function. By contrast, the second type proposes that the bits representing the integer I_{n-j} and I_{n-k} are rotated by r_1 and r_2 positions, respectively before their modulo 2 addition

$$I_n = [(I_{n-j} \textbf{ rot } r_1) + (I_{n-k} \textbf{ rot } r_2)] \textbf{ mod } 2^b, \tag{4.28}$$

where I_n is an integer represented by b bits and the notation $(I_{n-j}$ **rot** $r_1)$ means that the bits of I_{n-j} are shifted to the right by r_1 positions, as exemplified by $\{00001111_2$ (**rot**) $3 = 11100001_2\}$. As detailed in [22], the employment of a LCRIG has a limited benefit, because the probability of having different packet degrees is different. More explicitly, Fig. 4.14 portrays the packet degree distribution produced by the LCRIG of [22] for the SLT code of (1200, 1800, $c = 0.1$, $\delta = 0.5$), where c and δ are two parameters of the SLT code of [25]. The horizontal axis quantifies the specific degrees of the original source information packets, while the vertical axis represents the relative frequency of the specific degrees having corresponding values along the horizontal axis. As we can see in Fig. 4.14, the relative

frequency of the source information packet's degrees spans the range from degree-1 to degree-12.

By contrast, the relative frequency of the degrees created by the bit swapping random integer generator of [22] characterized in Fig. 4.14 is concentrated to a slightly narrower degree distribution range, most of which range from $d_m = 2$ to $d_m = 6$. When designing the random integer generator used for encoding SLT codes, we aim for making this range as narrow as possible, since this allows us to maintain a near-constant probability of the original information packets being represented by the parity packets. Consequently, the resultant probability that an original information packet is recovered from the parity packets also becomes near-constant. Hence, we may conclude from the above interpretation of Fig. 4.14 that the performance of SLT codes using the BSRIG is expected to be better than that of the LCRIG of [22].

Nonetheless, the degree of SLT source packets still varies over quite a wide range even

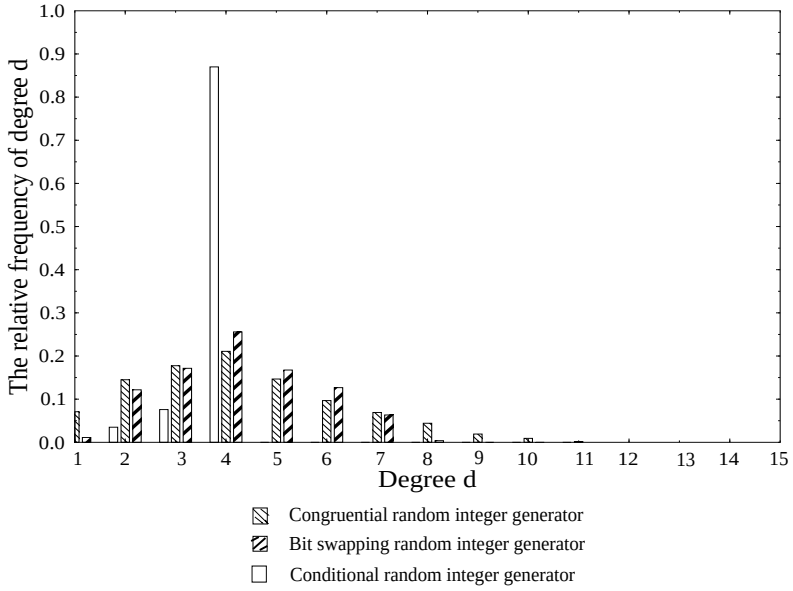


Figure 4.14: Three degree distributions for the source SLT packets created by two different random integer generators of [22] and the conditional random integer generator.

for the BSRIG in Fig. 4.14, which degrades the performance of the resultant SLT codes. Hence, for the sake of generating the desired 'Dirac-delta-like' degree distribution ideally associated with a single degree value for all input packets, we specifically design a novel random integer generator, which we refer to as the conditional random integer generator, in order to improve the BSRIG of [22]. More specifically, we improve the BSRIG of [22],

while satisfying the condition of $d_m \leq \overline{D}_m$ for the output degree of the message nodes, where $\overline{D}_m$ is the mean degree of the message nodes, which is calculated as follows:

$$\overline{D}_m = \left\lceil \frac{1-R}{R} \cdot (\overline{d}_c - 1) \right\rceil . \tag{4.29}$$

Furthermore, $\overline{d}_c$ is the average degree of the $(n-k)$ check nodes, which is calculated as follows:

$$d_c = d + 1, \tag{4.30}$$

where d is generated by the degree distribution $\Omega(d)$ of (4.24), while $R = \frac{k}{n}$ is the code rate of the SLT code. When using this conditional random integer generator, we arrive at the degree distribution marked in Fig. 4.14 by the hollow bars. Observe that most of the output degree values are concentrated around $d_m = 4$. This is expected to improve the attainable performance of the resultant SLT, as we will demonstrate in Section 4.3.5.

4.3.4 EXIT Chart Analysis of SLT Codes

As argued in Fig. 4.5 of Section 4.2.2 [25] the EXIT chart of SLT codes is constituted by a message passing function between the message nodes and parity check nodes specified by the PCM **H**. Let us now consider the EXIT chart of our novel SLT code, which employs the degree distributions mentioned in Section 4.3.2 for transmission over the AWGN channel. At the beginning of the decoding process, the LLRs of the SLT-encoded bits are calculated from the output of the matched filter in the demodulator according to:

$$\log\left(\frac{Pr_{(b_i=0/r)}}{Pr_{(b_i=1/r)}}\right) = \log\left(\frac{\sum_{s_i \in S_0} e^{(-\frac{|r-s_i|^2}{N_0})}}{\sum_{s_i \in S_1} e^{(-\frac{|r-s_i|^2}{N_0})}}\right), \tag{4.31}$$

where $s_i \in S_0$ denotes a modulated symbol with the ith bit being equal to zero. Initially, the *a priori* LLRs of the parity check nodes are set to the above-mentioned matched filter output LLRs. The output LLRs of the check nodes are set to zero at this stage. At the first iteration, the LLR messages $R(j)$ passed from the jth check nodes to the message nodes are updated as follows:

$$R(j) = 2 \cdot \operatorname{arctanh}\left(\prod_{i}^{d_c} Q(i)\right), \tag{4.32}$$

where $Q(i)$ represents the associated LLR message passed from the ith message node to the check nodes and d_c is the degree of the parity check node, which obeys our truncated

degree distribution proposed in Section 4.3.2. The extrinsic LLRs $Q(i)$ passed from the ith message node to the jth check node are updated according to:

$$Q(i) = \sum_{j}^{d_v} R(j), \tag{4.33}$$

where $R(j)$ is the associated LLR message received from the jth check node and d_v is the degree of the message node.

From the above extrinsic LLRs we generated the EXIT curves for the SLT(1200,3600)

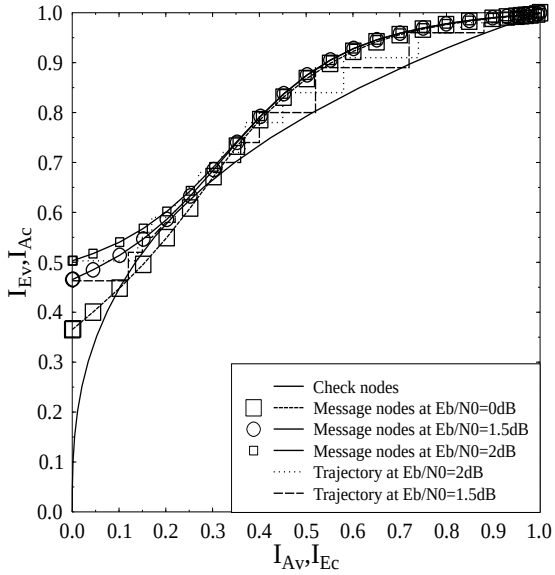


Figure 4.15: EXIT chart of the SLT(1200,3600) code of Table 4.2.

code, which is seen in Fig. 4.15. The message-node-related EXIT curve of the $R = 1/3$-rate SLT(1200,3600) code recorded at $E_b/N_0 = 0$dB intersects the check-node-related curve. Consequently, at $E_b/N_0 = 0$dB the EXIT curves of the SLT(1200,3600) code indicate that we cannot achieve an infinitesimally low BER. By contrast, at $E_b/N_0 = 1.5$dB the EXIT curves of the SLT code no longer intersect and the decoding trajectory matches well the EXIT curves, hence after about $I = 20$ decoding iterations an infinitesimally low BER may be attained as indicated by the bit-by-bit decoding trajectory printed in dotted line. By contrast, at $E_b/N_0 = 2$dB only $I = 12$ decoding iterations are necessitated. When gradually changing the factor γ from 2 to higher values such as 3, 4, the BER performance of the SLT(1200,3600) code degrades, as seen in Fig. 4.16. This happens because the density of the PCM $\mathbf{H}$ is increased, which decreases the Hamming distance

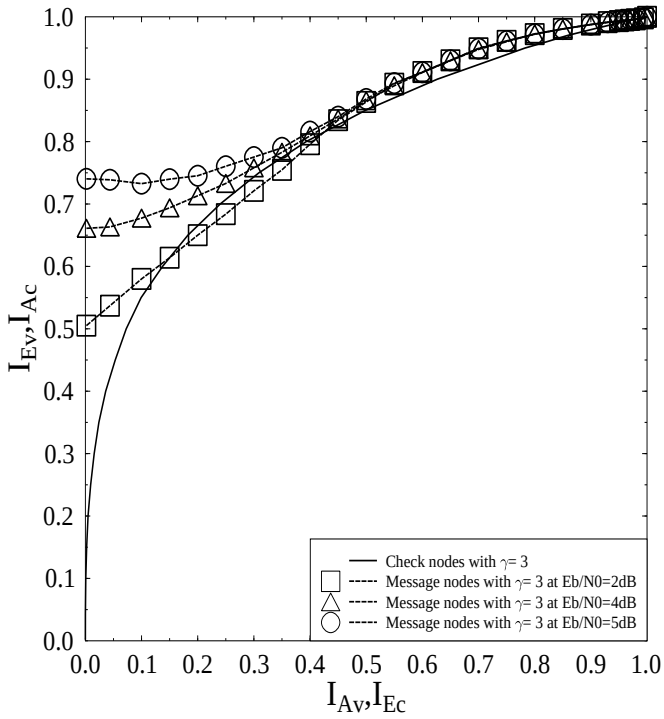


Figure 4.16: EXIT chart of the SLT(1200,3600) code using the CRIG and TDD of (4.24) in conjunction with $\gamma = 3$.

between the columns of **H** and hence causes error propagation during the decoding process. By observing the EXIT charts plotted in Fig. 4.15 and Fig. 4.16 we may conclude that $\gamma = 2$ is the best choice for the TDD of the SLT(1200,3600) code. Finally, our simulation-based BER results are provided in the following section.

4.3.5 Simulation Results

An AWGN-contaminated BEC typically encountered in line-of-sight wireless Internet scenarios was assumed, where the statistical multiplexing-induced random uniformly distributed packet dropping events had a probability of $P_e = 0.1$. The parameters of the system investigated are summarised in Table 4.2.

The BER performance comparisons between SLT codes and LDPC codes are given in Fig. 4.17 and Fig. 4.18. As seen in Fig. 4.17, at BER $= 10^{-5}$ the SLT coded systems having a code rate of $R = 1/2$ require an approximately 1 dB lower E_b/N_0 value than the quasi-regular LDPC coded benchmarker having the same code rate r, when transmitting

The TDD's parameters in (4.24)	δ= 0.5 c= 0.1
The total number of source information packets	1200 × 100
SLT packet size	165 bits
SLT(1200,2400), SLT(1200,3600) SLT(1200,1800)	$\gamma = 2$ $\gamma = 3$
LDPC(1200,$\frac{1200}{R}$) with code rate R	1/3, 1/2, 2/3
Maximum number of iterations	20
The erasure probability P_e	0.0 and 0.1
Modulation	QPSK
Channel types	BEC, AWGN and uncorrelated non-dispersive Rayleigh channels

Table 4.2: System parameters.

over the Additive White Gaussian Noise (AWGN) channel. By contrast, an E_b/N_0 value of about 2.5 dB required in for maintaining BER$\leq 10^{-5}$ Fig. 4.18, when transmitting over the uncorrelated Rayleigh fading channel. The BER performance of the SLT code using the truncated degree distribution of Eq.(4.24) and the conditional random integer generator of Section 4.3.3 is shown in Fig. 4.19 for transmission over the AWGN channel using a QPSK modulator. Fig. 4.20 characterizes the BER performance of SLT codes for transmission over the BEC-AWGN channel having an erasure probability of $P_e = 0.1$. The BER performance of SLT codes is shown in Fig. 4.21 for transmission over the uncorrelated Rayleigh fading channel. Finally, Fig. 4.22 shows the attainable BER performance of SLT codes for transmission over the wireless Internet channel, where the data is contaminated by both uncorrelated non-dispersive Rayleigh fading and erased owing to statistical multiplexing induced collisions.

4.3.6 Conclusions

Recall from Fig. 4.17-Fig. 4.22 the SLT codes using both the proposed truncated degree distribution of (4.24) and the conditional random integer generator of (4.28) achieved a high performance for transmission over various combinations of the BEC, AWGN and Rayleigh fading channels. The system using the SLT(1200,3600) code of Table 4.2 requires low E_b/N_0 values for attaining output BERs lower than 10^{-5}, when communicating over

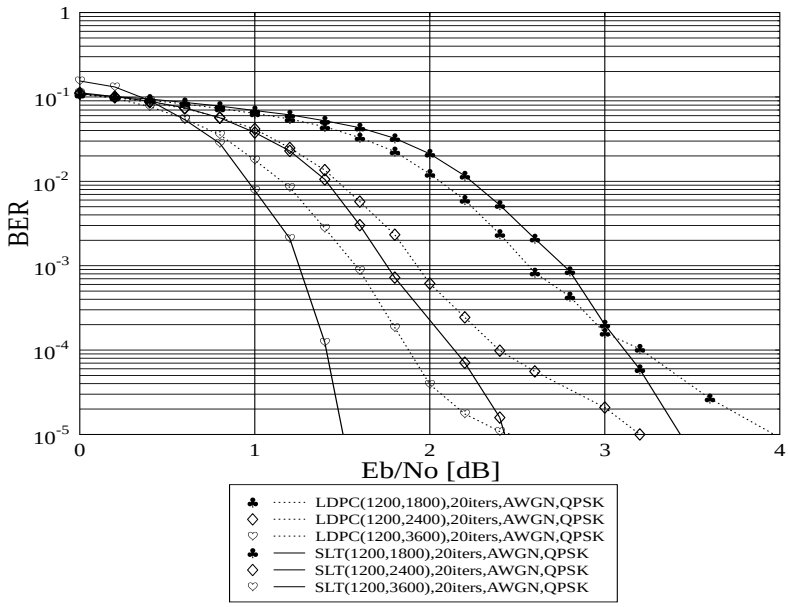


Figure 4.17: BER performances of the SLT and LDPC codes for transmission over the AWGN channel, when using a QPSK modulator and $P_e < 6.10^{-5}$. All other parameters were summarized in Table 4.2.

the different channels. Quantitatively, observe in Fig. 4.19 that this system is capable of achieving BER $\leq 10^{-5}$ for an E_b/N_0 value around 1.5 dB for transmission over the AWGN channel and at 3.5 dB over the uncorrelated, non-dispersive Rayleigh fading channel for $P_e < 6.10^{-5}$. By contrast, the corresponding benchmark system using the Low Density Parity Check code LDPC(1200,3600) requires an E_b/N_0 value of about 2.5 dB, as observed in Fig. 4.17 to achieve BER $\leq 10^{-5}$ over the AWGN channel. For the sake of improving the achievable of various MIMO systems, SLT codes are used for supporting the operation of different MIMO systems, such as Space Time Coded (SPC) [73] and V-BLAST [74] in the next section.

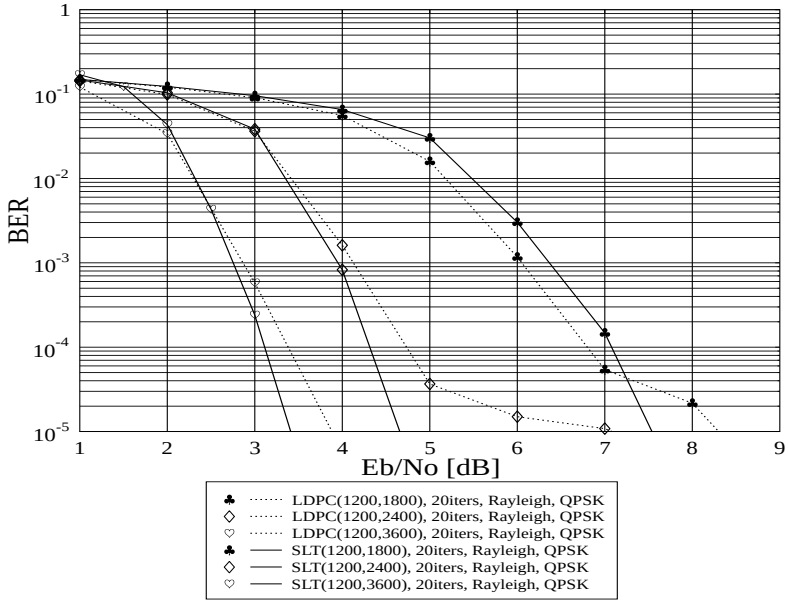


Figure 4.18: BER performance of the SLT and LDPC codes for transmission over the uncorrelated non-dispersive Rayleigh channel, when using a QPSK modulator and $P_e < 6.10^{-5}$. All other parameters were summarized in Table 4.2.

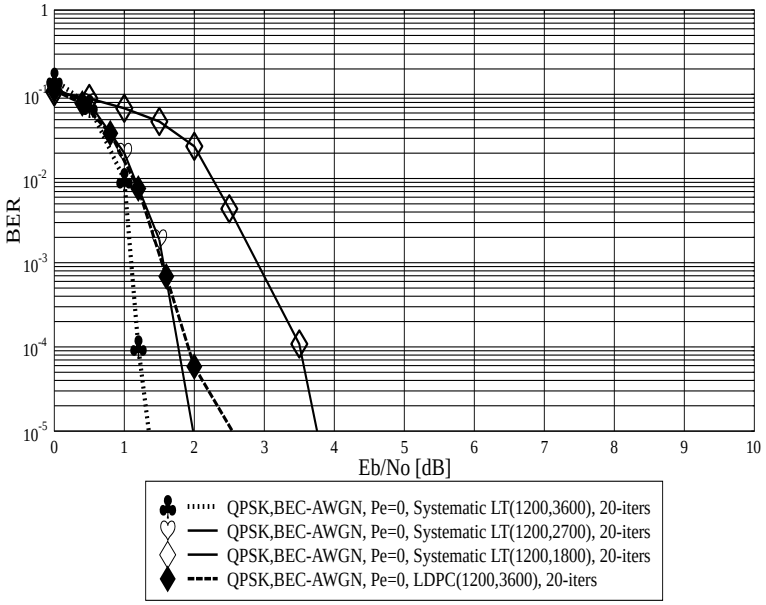


Figure 4.19: BER performance of the SLT codes over the AWGN channel, when using a QPSK modulator and the parameters of Table 4.2.

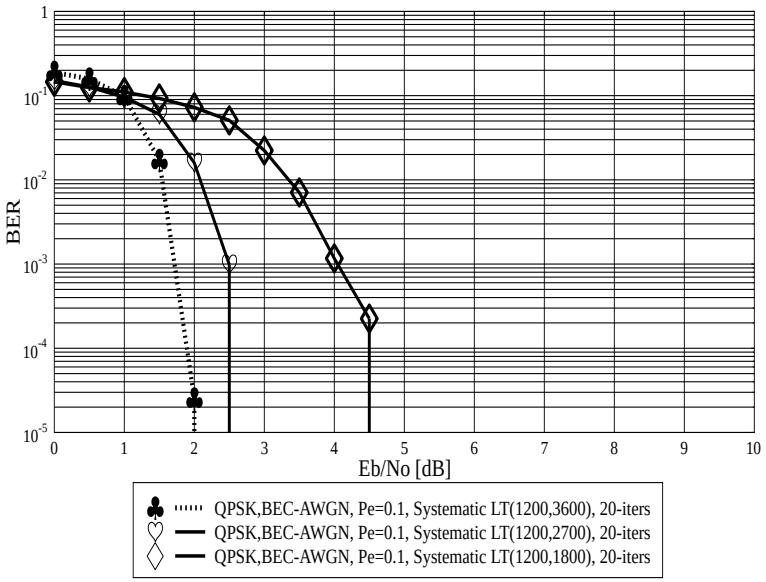


Figure 4.20: BER performance of the SLT codes over the BEC-AWGN channel associated with $P_e = 0.1$, when using a QPSK modulator and the parameters of Table 4.2.

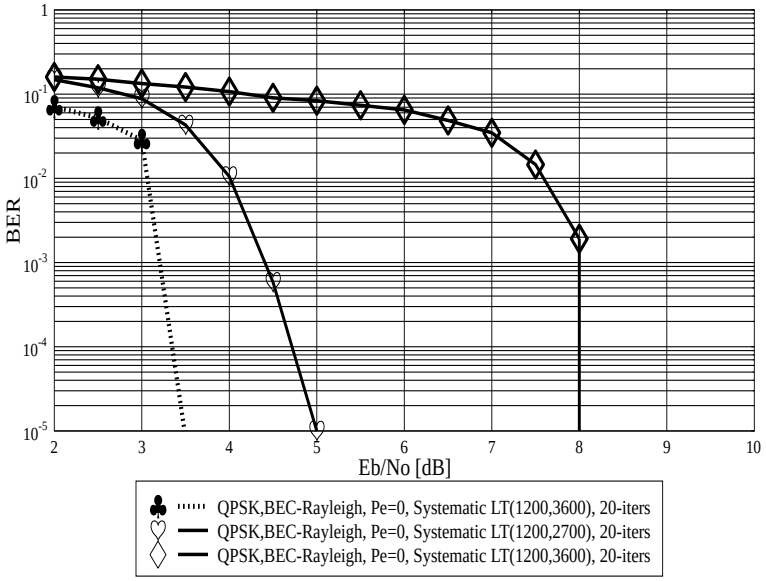


Figure 4.21: BER performance of the SLT codes over the uncorrelated non-dispersive Rayleigh channel, when using a QPSK modulator and the parameters of Table 4.2.

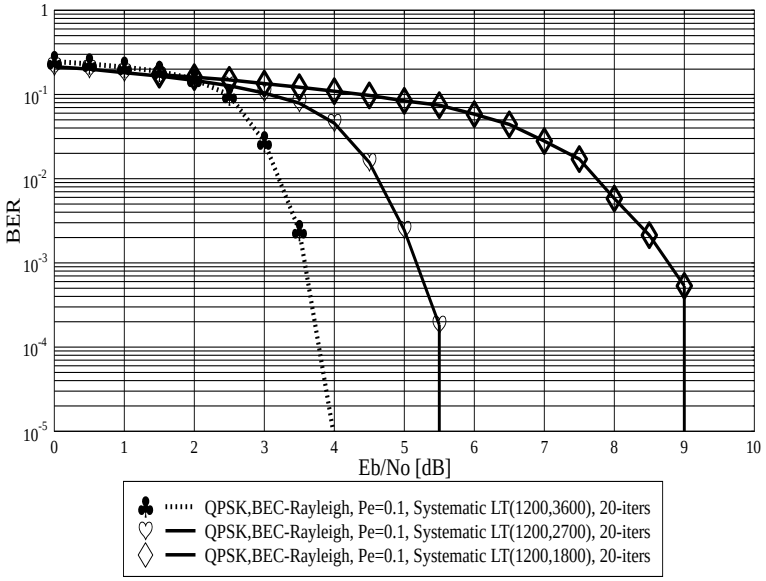


Figure 4.22: BER performance of the SLT codes over the Wireless Internet channel inflicting uncorrelated, non-dispersive Rayleigh fading and an erasure probability of $P_e = 0.1$, when using a QPSK modulator and the parameters of Table 4.2.

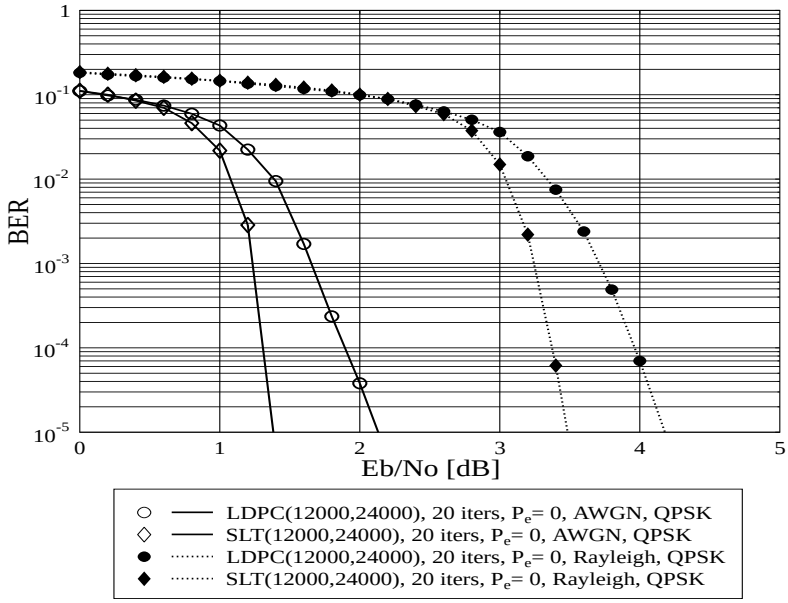


Figure 4.23: BER performances of SLT (12000, 24000) and quasi-regular LDPC (12000, 24000) codes for transmission over AWGN and uncorrelated non-dispersive Rayleigh channels, having a packet erasure probability of P_e= 0.0, when using a QPSK modulator.

4.4 SLT Code Design

This section analyses BER performances of SLT codes using different degree distributions of the SLT-parity packets and using different random integer generating algorithms for designing the degree distribution of SLT-source packets. There are three main types of degree distributions that are the IRSDD, the Poisson distribution and the TTD designed for the parity degree distribution of SLT codes. These distributions were mentioned in Chapter 3 and the previous Sections of this chapter. On the other hand, for designing the degree distribution of SLT-source packets, there are also three types of the random integer generating algorithms. These random integer generating algorithms are the LCRIG, the BSRIG and the CRIG. The BER performances and the complexity of SLT codes depend closely on these degree distributions and random integer generating algorithms as analysing as below.

4.4.1 Different Degree Distributions Designed for SLT-Parity Packets and BER Performances of SLT Codes

The SLT-parity degree distribution is designed based on three main distributions which are the Improved Robust Soliton Degree Distribution (IRSDD), the Possion Degree Distribution (PDD) and the Truncated Degree Distribution (TDD). These degree distribution are proposed in Chapter 2 and Chapter 3. Again, the functions of these degree distributions follows:

* The Improved Robust Soliton Degree Distribution [20]:

$$D(d) = \begin{cases} 0 & K > d > \frac{K}{S}, \\ \frac{1+S+\nu}{Z \cdot K} & \text{if } d = 1, \\ \frac{1}{Z}\left[\frac{1}{d(d-1)} + \frac{S}{K} \cdot \frac{1}{d}\right] & \text{if } 2 \leq d < \frac{K}{S}, \\ \frac{S}{Z \cdot K}\left[\log\frac{S}{\delta} + \frac{1}{(\frac{K}{S}-1)}\right] & \text{if } d = \frac{K}{S}, \end{cases} \tag{4.34}$$

where S is the number of degree-one SLT parity packets and $S \equiv c \cdot \log_e(K/\delta)\sqrt{K}$, Z is the normalized factor and $\{Z = \sum_d [(\rho(d) + \tau(d)) + \nu]\}$ and K is the number of SLT source packets

* The Truncated Degree Distribution [20]:

$$\Omega(d) = \mu(d,\gamma,\nu) = \begin{cases} \frac{1}{Z}\left[1 + \frac{S}{K} + \nu\right] & \text{for } d = \gamma, \\ \frac{\gamma}{Z}\left[\frac{1}{d\cdot(\frac{d}{\gamma}-1)} + \frac{S}{K}\frac{1}{d}\right] & \text{for } d = 2\gamma, 3\gamma, \cdots, \frac{K\cdot\gamma}{S} - 1, \\ \frac{S}{Z\cdot K}\left[\log(\frac{S}{\delta}) + \frac{1}{(\frac{K}{S}-1)}\right] & \text{for } d = \frac{\gamma\cdot K}{S}, \\ 0 & \text{for } d > \frac{\gamma\cdot K}{S} \text{ and } d = 1, \end{cases} \tag{4.35}$$

where S is the number of degree-γ SLT parity packets and $S \equiv c \cdot \log_e(K/\delta)\sqrt{K}$.

* The Poisson Degree Distribution [20]:

$$\Omega_d(\lambda) = \begin{cases} \frac{\mu\cdot\lambda}{1+\mu} \cdot \frac{1}{\Omega(\lambda)} & \text{for } d = 1, \\ \frac{\lambda^d}{d(d-1)} \cdot \frac{1}{\Omega(\lambda)} & \text{for } d = 2, 3, ..., D, \end{cases} \tag{4.36}$$

where d is the variable degree, $(D+1)$ is the maximum degree and $[D = \frac{4(1+\varepsilon)}{\varepsilon}]$, ε is the real number bigger than zero, λ is the real number and $\{\lambda \in [\delta, 1]\}$, $(\delta = \frac{1}{D})$ and $[\mu = (\frac{\varepsilon}{2}) + (\frac{\varepsilon}{2})^2]$. The function $\Omega(\lambda)$ is defined as follows

$$\Omega(\lambda) = \frac{1}{1+\mu}(\mu\cdot\lambda + \frac{\lambda^2}{1\cdot 2} + \frac{\lambda^3}{2\cdot 3} + \cdots + \frac{\lambda^D}{(D-1)\cdot D} + \frac{\lambda^{D+1}}{D}). \tag{4.37}$$

In this section we analyse the BER performance of SLT codes using the Truncated Poisson Degree Distribution (TPDD) which were optimized in [11] as follows:

$$\begin{aligned} \Omega_1(d) &= 0.007969d^1 + 0.5161d^2 + 0.166220d^3 + 0.072646d^4 + 0.082558d^5 + \\ &\quad 0.056058d^8 + 0.037229d^9 + 0.055590d^{19} + 0.025023d^{65} + 0.003135d^{66}; \quad (4.38) \\ \Omega_2(d) &= 0.007544d^1 + 0.5033d^2 + 0.166458d^3 + 0.071243d^4 + 0.084913d^5 + 0.049633d^8 \\ &+ 0.043365d^9 + 0.045231d^{19} + 0.010157d^{20} + 0.010479d^{66} + 0.017365d^{67}, \quad (4.39) \end{aligned}$$

where IRSDD denotes the Improved Robust Soliton Degree Distribution defined in (4.34) with c= 0.1 and δ= 0.5, IRSDD1 denotes the Improved Robust Soliton Degree Distribution defined in (4.34) with c= 0.2 and δ= 0.05, TDD2 denotes the Truncated Degree Distribution defined in (4.35) with γ= 2, TDD3 denotes the Truncated Degree Distribution defined in (4.35) with γ= 3, TPDD1 denotes the Truncated Poisson Degree Distribution defined in (4.38) and TPDD2 denotes the Truncated Poisson Degree Distribution defined in (4.39). LCRIG stands for the Linear Congruential Random Integer Generator, SBRIG stands for the Swapping Bit Random Integer Generator and CRIG stands for the Conditional Random Integer Generator used for designing the SLT source packet degree distributions.

The BER performance of SLT codes recorded for transmissions over the AWGN channel using different distributions for designing the SLT parity packets and the LCRIG for the source packet degree distribution is portrayed in Fig. 4.24. By contrast, Fig. 4.25 shows the BER performance of SLT codes using different distributions for designing the SLT parity packets and the SBRIG for the source packet degree distribution. Finally, Fig. 4.26 plots the BER performance of SLT codes using different distributions for designing the SLT parity packets and the CRIG for the source packet degree distribution. Observe in Figs. 4.24, 4.25, 4.26 that SLT codes using the TDD2 of (4.35) with $\gamma = 2$ for designing the SLT parity packet degree distribution and employing the three random integer generators LCRIG, SBRIG, CRIG for coining the specific source packets outperform the identical SLT codes using other degree distributions, despite employing the same three random integer generators. Also as seen in Figs. 4.24, 4.25 and 4.26, SLT codes using the TPDD1 of (4.38) has a worse BER performance than SLT codes using other SLT parity packet degree distributions.

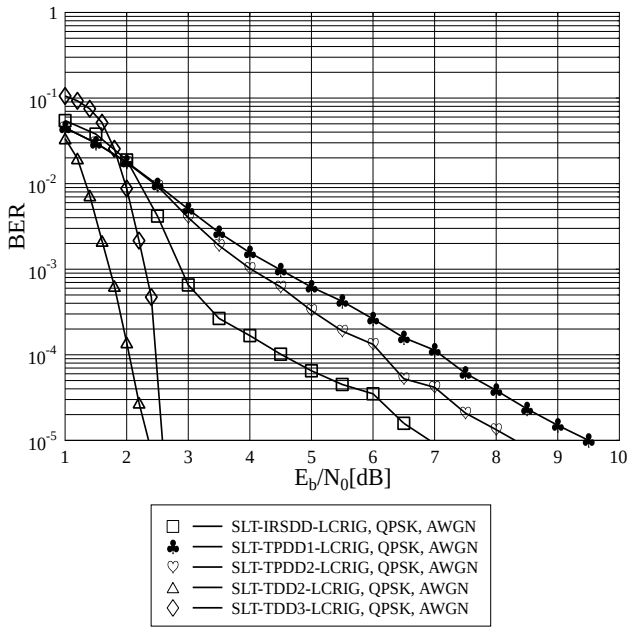


Figure 4.24: BER performance of the SLT codes over the AWGN channel using different distributions for designing the LT parity packets and the LCRIG for the source packet degree distribution,

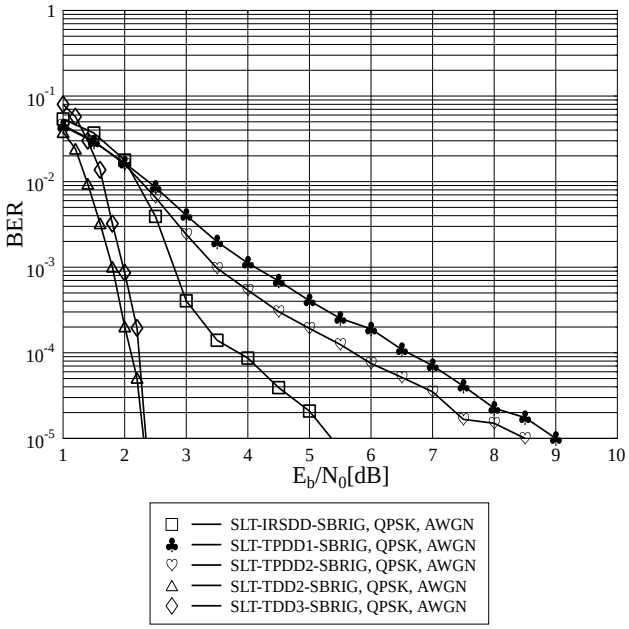


Figure 4.25: BER performance of the SLT codes over the AWGN channel using different distributions for designing the LT parity packets and the SBRIG for the source packet degree distribution.

4.4.2 Random Integer Generators and Their BER Performance

There are three types of random integer generators used for coining the SLT source packets which are randomly chosen to create the SLT parity packets in the SLT encoding process. These random integer generators proposed in Section 3.3.4 are the LCRIG, the SBRIG and the CRIG. The BER performances of SLT codes using different random integer generators are plotted in Figs. 4.27, 4.28, 4.29, 4.30 and 4.31. We can see in Figs. 4.27, 4.28 and 4.29 that the BER performances of the SLT codes using IRSDD, TPDD1 and TPDD2 of (4.34), (4.38) and (4.39) for designing the parity packet degree distribution are fairly different, when using different random integer generators for designing the source degree distribution. By contrast, as shown in Figs. 4.30 and 4.31 when employing the TDD2 and TDD3 of (4.35) with (γ= 2 and 3) for designing the SLT parity packet degree distribution, the BER performances of the SLT codes employing different random integer generators for designing the source degree distribution are only marginally different.

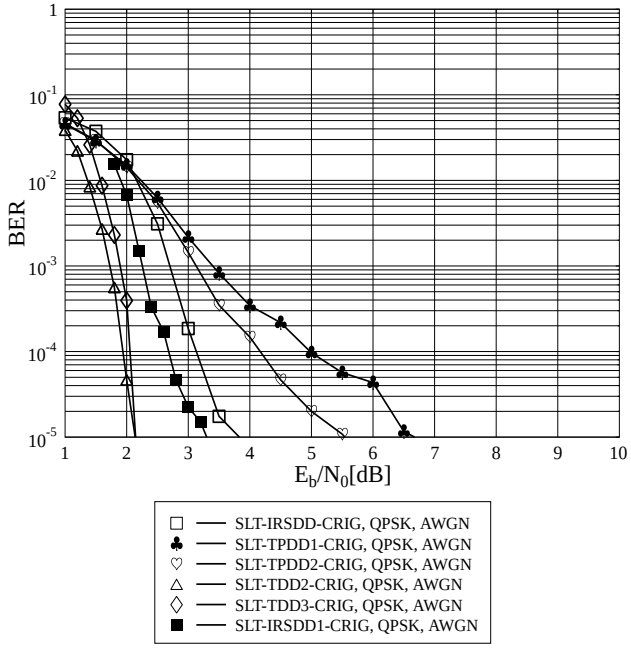


Figure 4.26: BER performance of the SLT codes over the AWGN channel using different distributions for designing the LT parity packets and the CRIG for the source packet degree distribution.

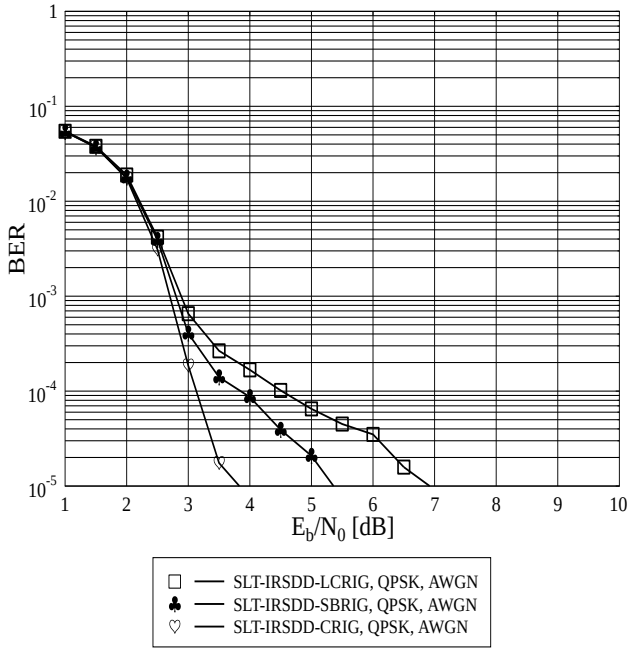


Figure 4.27: BER performance of the SLT codes over the AWGN channel using the IRSDD for designing the LT parity packets and different random integer generators for the source packet degree distribution.

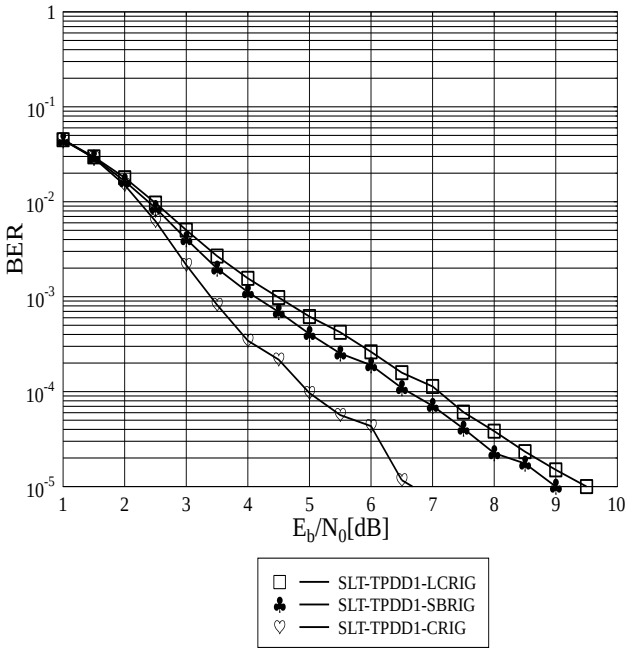


Figure 4.28: BER performance of the SLT codes over the AWGN channel using the TPDD1 for designing the LT parity packets and different random integer generators for the source packet degree distribution.

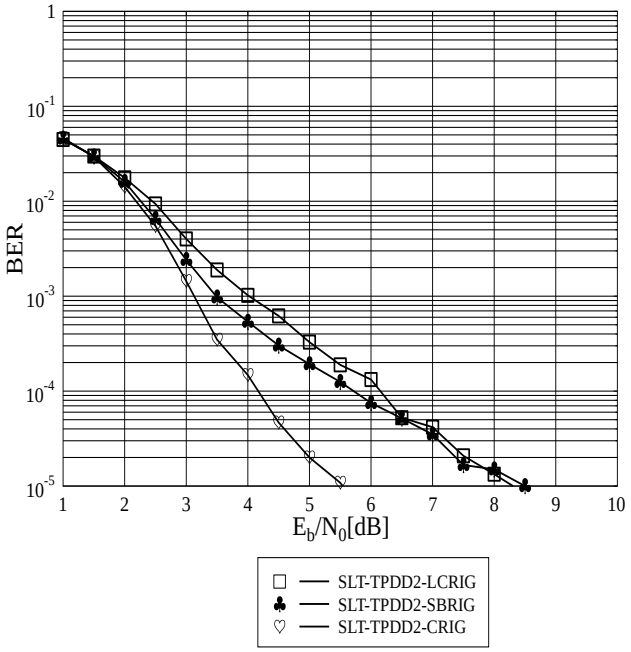


Figure 4.29: BER performance of the SLT codes over the AWGN channel using the TPDD2 for designing the LT parity packets and different random integer generators for the source packet degree distribution.

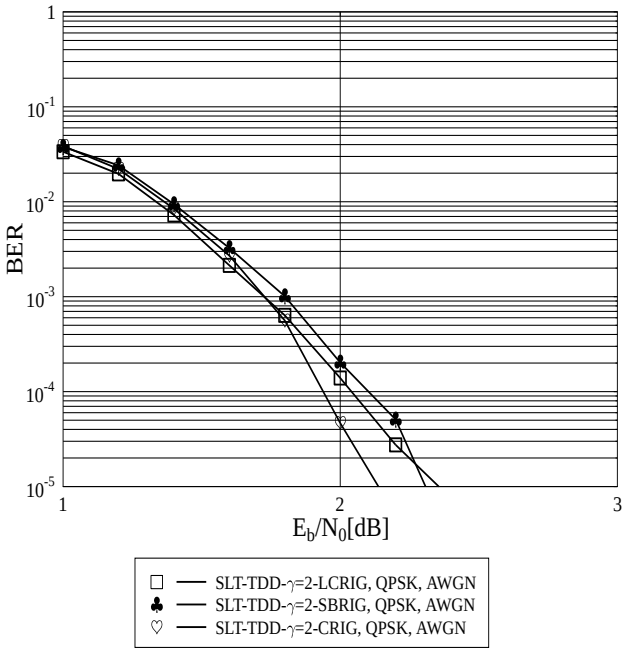


Figure 4.30: BER performance of the SLT codes over the AWGN channel using the TDD2 for designing the LT parity packets and different random integer generators for the source packet degree distribution.

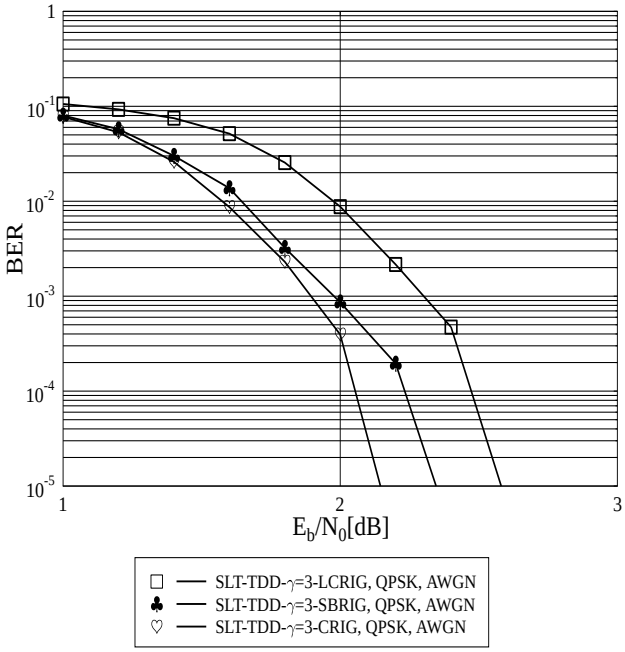


Figure 4.31: BER performance of the SLT codes over the AWGN channel using the TDD3 for designing the LT parity packets and different random integer generators for the source packet degree distribution.

4.4.3 Conclusions

SLT codes using the TDD2 and TDD3 can achieve BER $\leq$ 10^{-5} at E_b/N_0 in excess of 2dB with all types of random integer generators applied for designing the SLT source packet degree distribution, while SLT codes using other distributions such as the IRSDD needs E_b/N_0 in exceed of 6dB with the LCRIG as seen in Fig. 4.24, in exceed of 5dB with the SBRIG as plotted in Fig. 4.25 and in exceed of 3dB with the CRIG as portrayed in Fig. 4.26. The SLT codes using the Truncated Poisson Degree Distribution (TPDD) type 1 and type 2 (TPDD1,TPDD2) require E_b/N_0 in exceed of 8dB as seen in Fig. 4.24 and Fig. 4.25 to get BER $\leq$ 10^{-5}. However, the PCM's density of the SLT codes using the TDD is higher than the one of the SLT codes using the IRSDD and TPDD, hence, the SLT codes using the TDD2 has the complexity higher than two times the one of the SLT codes using the IRSDD and about four times the one of the SLT codes using the TPDD1 and TPDD2. When the density of the SLT codes using the TDD is increased by raising the factor γ in (4.35) from 2 to 3, the BER performances of the SLT codes slightly decrease. This happens because the PCM's column and row weights, which relates to the size of SLT code words as well as the density of SLT codes' PCM, reach their upper limit.

4.5 A Systematic Luby Transform Coded V-BLAST System

4.5.1 Introduction

The fundamental limitations of reliable wireless transmissions are imposed by the time-varying nature of typical multipath fading channels, which may be efficiently circumvented by sophisticated transceiver design [75] employing multiple antennas at both the transmitter and the receiver. Recent information theoretic studies [76] [77] have revealed that employing a Multiple-Input Multiple-Output (MIMO) system significantly increases the capacity of the system. In [74], Wolniansky *et al*. proposed the popular multi-layer MIMO structure, known as the Vertical Bell Labs Layered Space-Time (V-BLAST) scheme. In V-BLAST systems, each transmit antenna simultaneously transmits an independent data stream within the same carrier frequency band. At the receiver side, provided that the number of receive antennas is higher than or equal to the number of transmit antennas, a low-complexity Successive Interference Cancellation (SIC) based detection algorithm may be applied for detecting the transmitted data [56]. The V-BLAST receiver is capable of providing a tremendous increase of a single user's effective bit-rate without the need for any increase in the transmitted power or the system's bandwidth. However, its impediment is that it was not designed for exploiting transmit diversity and the decision errors of a particular antenna's detector propagate to other bits of the multi-antenna symbol, when erroneously cancelling the effects of the sliced bits from the composite MIMO signal. SLT codes were proposed in Section 4.3 [25] for exploiting that soft-bit decoding algorithms are capable of providing good BER performances, while communicating over both the traditional BEC modelling statical multiplexing-induced packet loss event of the Internet and noise contaminated channels. The algorithm of Section 4.3 was further developed for achieving an improved performance in single-antenna aided systems, when communicating over a wide class of channels such as the BEC channel, the AWGN and the uncorrelated Rayleigh fading channels. The SLT codes advocated in Section 4.3 outperform conventional quasi-regular LDPC codes in the wireless channel.

In this section we study the performance of a novel SLT aided V-BLAST system communicating over a MIMO channel. The proposed scheme employs set partitioning [78], [79] based bit-to-QPSK mapping and at the receiver side iterative extrinsic information exchange is used between the SLT decoder and the QPSK demapper. The SLT coded scheme is compared to a Recursive Systematic Convolutional (RSC) coded and Unity Rate Coded

scheme. We will demonstrate that the SLT coded system outperforms its benchmark scheme.

The organization of this section is as follows. In Section 4.5.2.1 we describe the basic concept of the V-BLAST architecture, while in Section 4.5.2.3 the SLT coded V-BLAST system. In Section 4.5.3 we analyse the SLT coded V-BLAST system using EXIT charts. Finally, in Section 4.5.4 we present our BER performance results followed by our conclusions in Section 4.5.5.

4.5.2 System Architecture

4.5.2.1 V-BLAST

Again, V-BLAST [80] provides a high throughput, at the cost of a modest diversity gain. Let $\mathbf{x}^T = [x_1 \quad x_2 \quad x_3 \quad x_4]$ denote the vector of QPSK symbols to be transmitted by the four antennas during a symbol interval. Then the corresponding received vector can be represented as

$$\mathbf{r}_t = \mathbf{H} \cdot \mathbf{x}_t + \mathbf{n}_t, \tag{4.40}$$

where $\mathbf{r}_t$ is the received signal vector, $\mathbf{H}$ is the $(n_r \times n_t)$-element Channel Impulse Response (CIR) matrix, where n_t is the number of transmit antennas, n_r is the number of receive antennas and h_{ij} represents the CIR coefficients between transmit antenna j and receive antenna i, while $\mathbf{n}_t$ denotes the noise vector at time instant t. V-BLAST detection may be carried out using for example SIC or the Zero Forcing (ZF) algorithm [74].

4.5.2.2 Systematic Luby Transform Codes

The SLT codes of Section 3.1 [20] were based on the Improved Robust Soliton Distribution (IRSD)[1] of Section 3.1.1 [20], which were invoked for the sake of designing the degree distribution of the parity part of the generator matrix.

[1]The degree distribution of a SLT code determines the specific number of source packets, which contribute to an SLT-encoded packet. A further important code parameter is the particular random number, which determines the specific degree of a given SLT-encoded packet.

4.5.2.3 SLT Coded V-BLAST System Overview

The SLT coded V-BLAST system's structure is portrayed in Fig. 4.32, which employed four transmit and four receive antennas. At the transmitter side, the source information

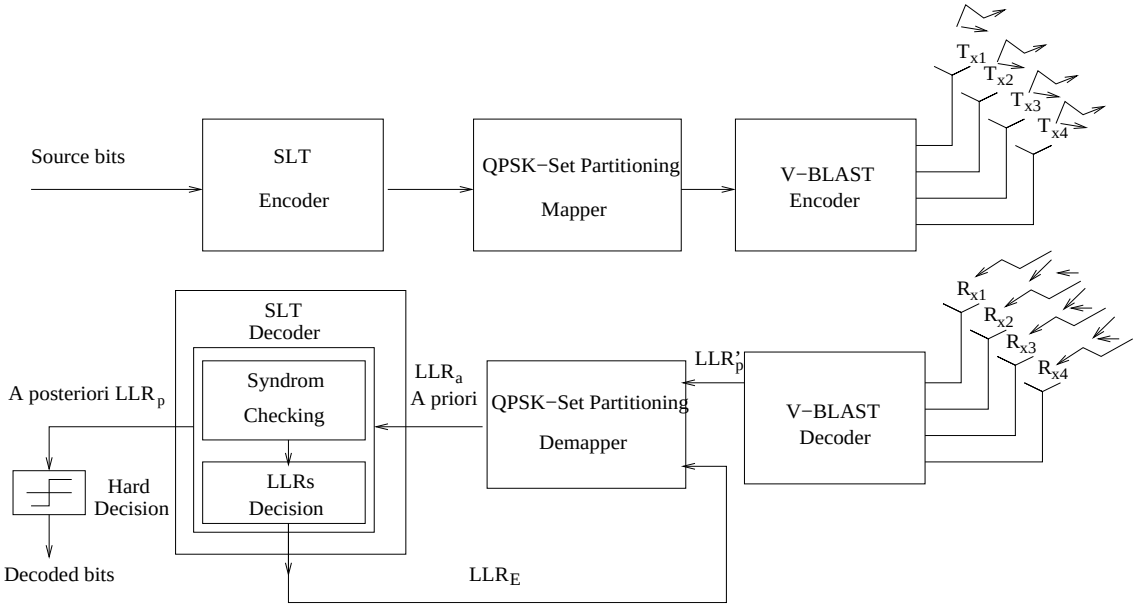


Figure 4.32: The block diagram of the SLT coded V-BLAST system.

bits are encoded by the SLT encoder, which are then mapped to the QPSK symbols of the modulator employing set partitioning based mapping [81]. The symbols at the output of the QPSK modulator are encoded by the V-BLAST encoder and transmitted by the four antennas over a correlated narrow-band Rayleigh fading channel having a normalised Doppler frequency of 0.01.

At the receiver side of Fig. 4.32, the received signal is decoded by the V-BLAST decoder, as described briefly in Section 4.5.2.1. The output of the V-BLAST decoder is then passed to the QPSK demapper, which also receives *a priori* information from the SLT decoder. The extrinsic LLR values at the output of the QPSK demapper are passed to the SLT decoder as *a priori* information. The SLT decoding process is assisted by the syndrome-checking block of Fig. 4.32. The output LLRs of the SLT decoder directly correspond to the *a posteriori* LLRs, when the syndrome checking block detects the syndrome $S = C \cdot H^T = 0$, where C is a legitimate codeword of the SLT code and H^T is the transpose of the PCM $\mathbf{H}$ of the SLT code. However, when the syndrome becomes $S \neq 0$, then the output LLRs of the SLT decoder no longer constitute the *a posteriori* LLRs. Instead, they constitute the extrinsic LLR information, which is calculated by subtracting

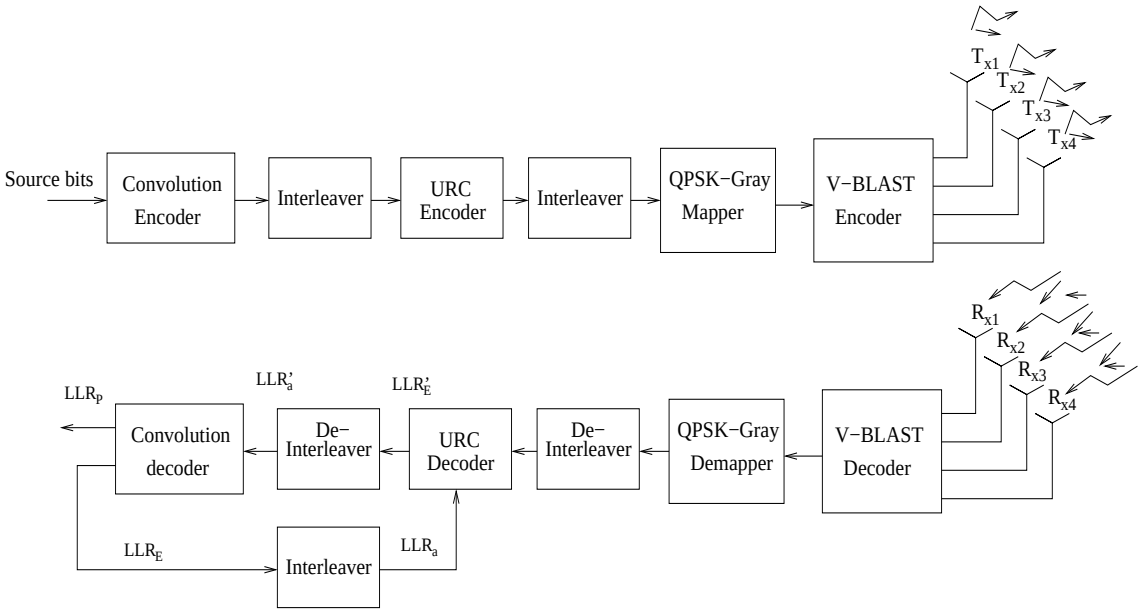


Figure 4.33: The block diagram of the RSC-URC coded V-BLAST system.

the *a priori* LLRs from the *a posteriori* LLR values.

As seen in Fig. 4.32, the output LLRs of the SLT decoder are fed back to the QPSK demapper as *a priori* LLRs. The extrinsic information aided iterative decoding process is then continued between the SLT decoder and the QPSK demapper, until all syndromes S at the SLT decoder become equal to zero or the number of iterations reaches the maximum affordable value. During the last iteration, the *a posteriori* LLR values generated at the output of the SLT decoder are passed to the hard decision block for the sake of recovering the original information bits.

The benchmark scheme considered in this paper consists of a Recursive Systematic Convolutional (RSC) code used as the outer code and a Unity Rate Code (URC) as the inner code, as shown in Fig. 4.33. The benefit of using the RSC code combined with the URC is that the system's impulse response has an infinite memory, which assists the system in efficiently exploiting the extrinsic information even at relatively low interleaver delays. In the benchmark scheme, the information bits are firstly encoded by the RSC encoder and then precoded by the URC encoder, before they are Gray mapped by the QPSK modulator and transmitted using the V-BLAST scheme. The benchmark receiver carries out decoding iterations between the RSC decoder and the URC decoder.

4.5.3 EXIT Chart Analysis

In the SLT code aided V-BLAST system of Fig. 4.32, the SLT code constitutes the outer code and the QPSK-set partitioning based mapper is the inner decoder.

The iteration process is implemented by passing *extrinsic* LLRs between the SLT decoder and the QPSK-set partitioning based demapper at the receiver. In Fig. 4.34 we plot the inverted EXIT curve of the SLT decoder as well as the EXIT curves of the QPSK demapper for various E_b/N_0 values. As seen in Fig. 4.34, the interleaver length is $L = 2,400$ bits. The EXIT tunnel between the demapper and the SLT decoder is quite wide at $E_b/N_0 = 3dB$, when we use the set partitioning based mapper, but the bit-by-bit stair-case-shaped decoding trajectory does not accurately match the EXIT curves of the inner and outer codes. This decoding trajectory mismatch is because for the short-duration 2400-bit interleaver length the LLRs are no longer Gaussian distributed, although this assumption is exploited by the EXIT chart[2]. Hence, the BER performance of the system employing an $L = 2,400$-bit interleaver does not match accurately the predictions of the EXIT chart. By contrast, the decoding trajectory does match the EXIT curves for $E_b/N_0 > 6$ dB quite accurately. Observe that two EXIT curves are plotted for the SLT code in Fig. 4.34, one of them is drawn using a continuous line and the other using dashed line. The dashed curve represents the EXIT curve of the SLT decoder dispensing with the syndrome checking block of Fig. 4.32, while the other does employ the syndrome checking block. In other words, when using syndrome checking at the SLT decoder, the achievable BER performance improves and the size of the open EXIT-chart area S seen in Fig. 4.34 is characteristic of the achievable BER gain. For the sake of improving the BER performance of the SLT coded V-BLAST scheme let us now increase the interleaver length to $L = 24,000$ bits. The system now becomes capable of achieving convergence at $E_b/N_0 = 5dB$ after $I = 2$ iterations, as seen in Fig. 4.35. Increasing the interleaver length will improve its BER performance, but the complexity of the SLT coded V-BLAST system will also increase, as discussed in the next section.

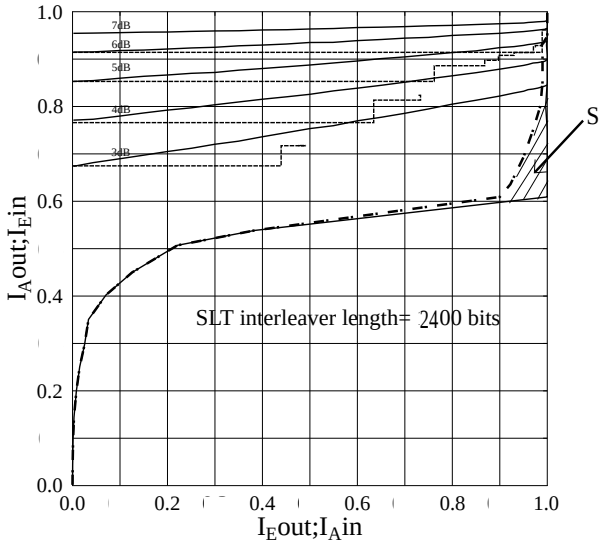


Figure 4.34: EXIT charts and decoding trajectories of the SLT coded V-BLAST system for L= 2,400 bits with all parameters of Table 4.3.

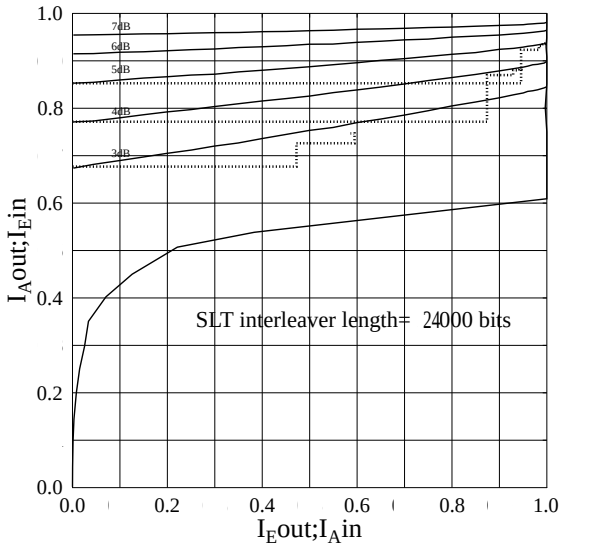


Figure 4.35: EXIT charts and decoding trajectories of the SLT coded V-BLAST system for L= 24,000 bits with all parameters of Table 4.3.

Parameters in (4.24)	δ= 0.5 c= 0.1
SLT code rates r	1/2
γ	2
The maximum number of inner iterations $Imax_{inner}$ of SLT codes	30
Modulation	QPSK-set partition mapping
Parameters V-BLAST	
Number of transmit antennas T_x	4
Number of receive antennas R_x	4
Iterative lengths	2,400 or 24,000

Table 4.3: System parameters.

4.5.4 Performance Analysis

In this section we characterise the BER performance of the SLT coded V-BLAST scheme against that of the benchmark scheme of Fig. 4.33. The benchmark scheme employs a 1/2-rate memory-2 RSC code having octally represented generator polynomials of $G_r = 7$ and $G = 5$, where G_r denotes the feedback generator polynomial and G denotes the feedforward generator polynomial. The URC code has a memory of 1. All simulation parameters are listed in Table 4.2. Fig. 4.36 shows the BER performance of the SLT coded V-BLAST system and that of the RSC-URC coded V-BLAST system, when using an interleaver length of L= 2,400 bits and L= 24,000 bits. When employing an L=2,400-bit interleaver, the SLT coded V-BLAST system achieves BER $\leq 10^{-6}$ at E_b/N_0= 6.5 dB, while the RSC-URC coded V-BLAST arrangement requires an E_b/N_0 in excess of 11 dB to achieve the same BER. When we increase the interleaver length to $L = 24,000$ bits, as seen in Fig. 4.36, the BER performance of the RSC-URC coded V-BLAST system improves more substantially than that of the SLT coded V-BLAST system, but still requires an E_b/N_0 value in excess of 5dB for achieving BER $\leq 10^{-6}$. However, the SLT coded V-BLAST system still achieves BER$\leq 10^{-6}$ at E_b/N_0= 4.5 dB.

The complexity of the two systems mainly depends on that of the SLT decoder and on

[2]Observe in Fig. 4.34 that the stair-case-shaped decoding trajectory is unable to reach the point of perfect convergence to an infinitesimally low BER, namely the point of $(I_A, I_E) = (1, 1)$.

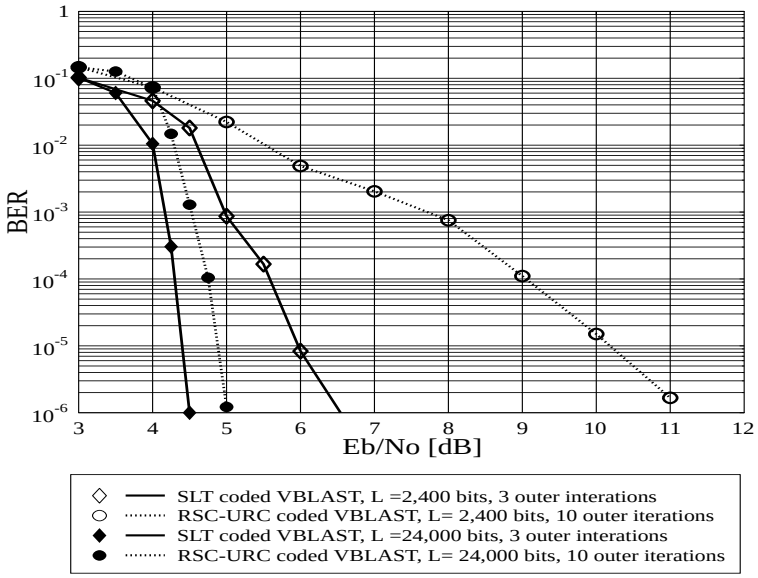


Figure 4.36: BER performance of the proposed and the benchmark systems at an interleaver length of $L = 2,400$ and $L = 24,000$ with all parameters of Table 4.3.

that of the RSC-URC decoders, when calculated based on the method proposed in [82]. We employed the Max-Log-MAP decoding algorithm [30] for the RSC and URC decoders, while the message passing technique for the SLT decoder. Let us consider an interlever length of $L = 2,400$ bits. Recall furthermore that $R = 1/2$ is the code rate of the SLT and RSC codes. The number of outer iterations used by the RSC-URC coded V-BLAST system was set to $I_{RSC-URC_{outer}} = 10$, while $m_1 = 2, m_2 = 1$ represents the memories of the RSC and URC codes, respectively. The message passing algorithm exchanged information between the message nodes and parity nodes. Hence, if $i = 8$ is the number of binary ones in a row of the PCM, then $j = i + 1 = 9$ is the number of binary ones in a column of the PCM of the SLT(1200,2400) code using the TDD distribution of (4.24). Hence, the PCM of the SLT has $i = 8$ and $j = 9$. The maximum number of inner iterations used by the SLT code is $Imax_{inner} = 30$ and the number of outer iterations is $I_{SLT_{outer}} = 3$. The complexity Λ_{SLT} of the SLT decoder is calculated in terms of the number of Add-Compare-Select (ACS) arithmetic operations as follows [82]:

$$\begin{aligned} \Lambda_{SLT} &= I_{SLT_{outer}} \times (Imax_{inner} \times 4 \times i \times L + Imax_{inner} \times j \times L) \\ &= 3 \times (30 \times 4 \times 8 \times 2400 + 30 \times 9 \times 2400) \\ &= 8,856,000 \ ACS. \end{aligned} \tag{4.41}$$

On the other hand, the complexity $\Lambda_{RSC-URC}$ of the benchmark scheme is calculated as follows [82]:

$$\begin{aligned}\Lambda_{RSC-URC} &= I_{RSC-URC_{outer}} \times L \times (10 \times 2^{m_1} \\ &\quad +10^{m_2} + 2 \times 4 \times 2^{m_1} + 2 \times 2^{m_2} \\ &\quad +2 \times 4 \times 2^{m_2} + 4^{m_1} - 2 + 4^{m_2} - 2) \\ &= 10 \times 2400 \times (10 \times 2^2 + 10 \times 2^1 \\ &\quad +2 \times 4 \times 2^2 + 2 \times 4 \times 2^1 \\ &\quad +4 \times 2^2 - 2 + 4 \times 2^1 - 2) \\ &= 3,072,000 \ ACS. \end{aligned} \tag{4.42}$$

From (4.41) and (4.42) we calculate the complexity ratio Λ_{ratio} between the SLT coded and the RSC-URC coded V-BLAST systems as follows:

$$\Lambda_{ratio} = \frac{\Lambda_{SLT}}{\Lambda_{RSC-URC}} = \frac{8,856,000}{3,072,000} = 2.88. \tag{4.43}$$

The number of inner iterations imposed by the SLT decoder when aiming for BER$\leq 10^{-5}$ is related to the E_b/N_0 value and the syndrome checking process. In other words, as E_b/N_0 increases, the number of inner iterations required in order to arrive at a syndrome of $S = 0$ decreases. Hence, the complexity of the SLT aided system obeys $\Lambda_{SLT} \leq 2.88 \times \Lambda_{RSC-URC}$.

4.5.5 Conclusions

In this section we proposed a SLT coded V-BLAST scheme for the sake of improving the V-BLAST system's performance. EXIT charts were used to analyse the system's performance. Observe by comparing Fig. 4.36 that the SLT coded system outperformed the benchmark scheme by about 5 dB for an interleaver length of L= 2,400 bits and by about 0.5 dB for L= 24,000 bits. When communicating over the correlated MIMO channel using the parameters of Table 4.3, the complexity of the SLT coded scheme was deemed to be 2.8 times higher than that of the RSC-URC coded scheme.

4.6 Chapter Conclusions

SLT codes using the soft-bit decoding algorithm have better BER performances in comparison with the original LT codes using the hard-bit decoding algorithm. LT codes alone

will propagate error when decoding contaminated received packets, which are transmitted across complicated channels such as the wireless Internet channel, where the transmitted packets are affected by erasure as well as noise and fading events. SLT codes using the soft-bit decoding algorithm combine more flexibly with other FEC codes than the original LT codes in the amalgamated systems as analysed above. For the sake of improving the BER performance of SLT codes in the combination schemes, we can implement the iteration between SLT decoders and other components of the systems such as demappers, other FEC decoders. Improving in designing the degree distributions for input and parity packets of SLT codes helps SLT codes out perform the other benchmark FEC codes such as the quasi-regular LDPC codes.

Chapter 5

Systematic Luby Transform Codes in H-ARQ Aided Iterative Receivers

5.1 Hybrid-Automatic Repeat reQuest Aided Coded Modulation

5.1.1 Introduction

In Automatic Repeat reQuest (ARQ) aided protocol [83] the source message is splitted into packets, each of which is provided with an information header H, as seen in Fig. 5.1. The integrity of the received packets is checked by the Cyclic Redundancy Checking (CRC) scheme for detecting any residual errors. When the receiver detects errors, it will use a feedback channel to request the transmitter to retransmit the packets containing errors. There are three classic ARQ schemes using different protocols, namely the stop-and-wait [84], the go-back-N [85] and the selective-repeat protocols [86]. The stop-and-wait protocol [84] is formulated as follows:

- The stop-and-wait protocol is the heart of the original Internet protocol [87], where the transmitter transmits one packet at a time and the receiver sends an ACKnowledgment (ACK) message, whenever it receives a correct packet. The transmitter then retransmits the packet after timeout, unless it received an ACK message from

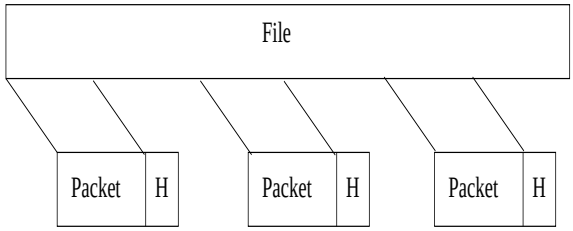


Figure 5.1: The source message is partitioned into transport packets in the ARQ protocol.

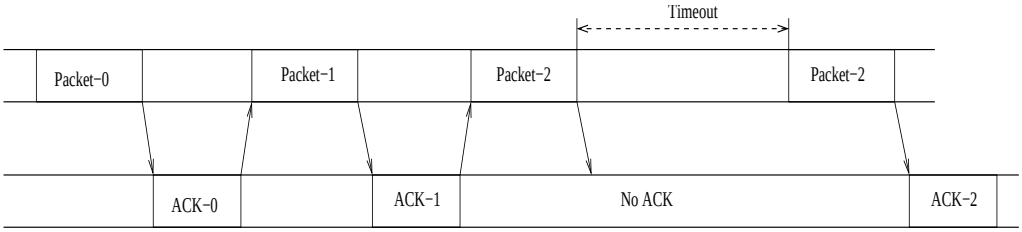


Figure 5.2: Timing issues in the stop-and-wait protocol.

the remote receiver, as seen in Fig. 5.2.

- The transmitter numbers the packets using Sequence Numbers (SN) and the receiver uses the so-called Request Numbers (ReqN) when managing the received packets. For instance, the receiver requests the retransmission of the $(j-1)^{st}$ packet by sending the message ReqN= j to the transmitter.
- The transmitter is idle while waiting for an ACK, as seen in Fig. 5.3.
- This protocol does not require the storage of packets.
- However, the protocol becomes inefficient, when the propagation delay is longer than the packet duration.
- The efficiency E of this protocol is limited by the round-trip delay and may be formulated according to Fig. 5.3 as follows:

$$E = D_{TP}/S, \tag{5.1}$$

where $S = D_{TP} + 2 \cdot D_P + D_{TA}$, D_P is the prop delay, D_{TA} is the ACK transmission duration and D_{TP} is the packet transmission duration.

The go-back-N protocol is described as follows [84]:

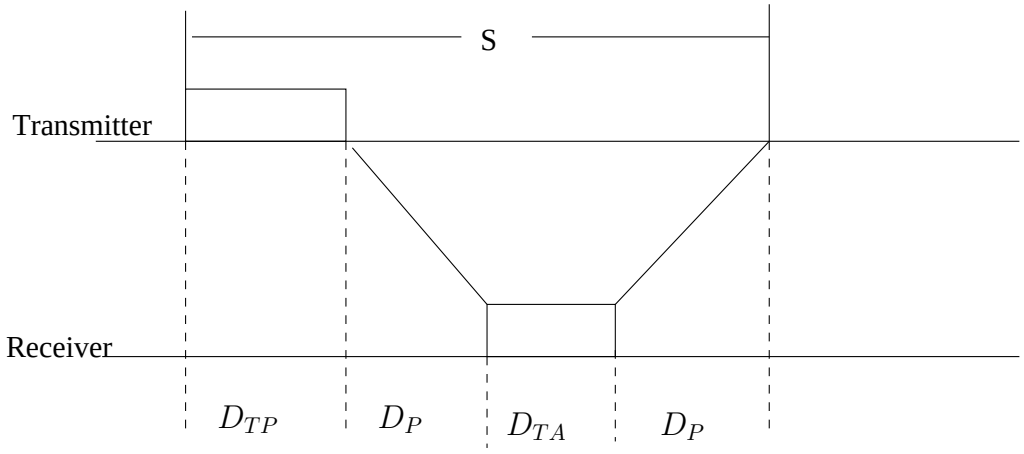


Figure 5.3: Timing issues in the stop-and-wait protocol.

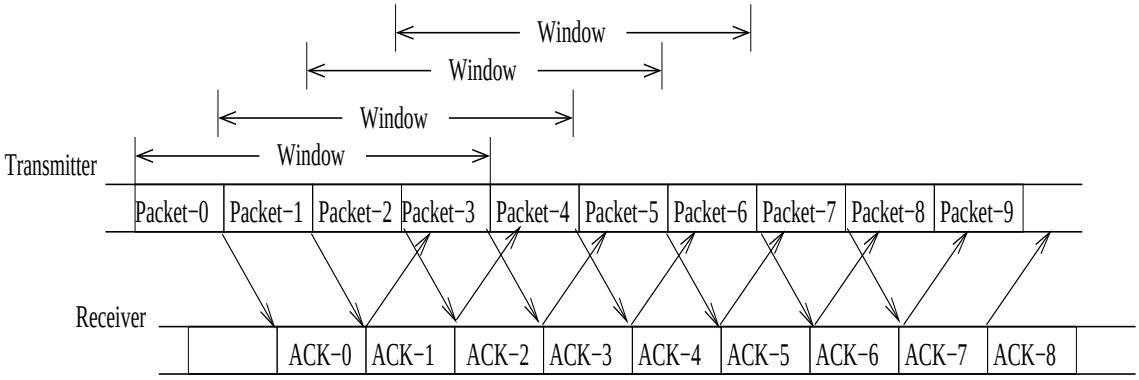


Figure 5.4: The go-back-N protocol.

- The go-back-N protocol uses a window-based mechanism, where the transmitter can send packets that are within a "window" of packets, allowing the transmission of new packets before a number of earlier ones are acknowledged. Fig. 5.4 shows the transmission process of the go-back-N protocol.

- If the window size is N packets, then the transmitter cannot send packet $i+N$, until it has received the ACK for packet i.

- The receiver operates as in the stop-and-wait protocol and it cannot accept packets out of their original sequence. When the receiver sends Receive Number (RN)= $(i+1)$, this acknowledges all packets in the window.

- The go-back-N protocol requires that the size N of the window is set to a sufficiently

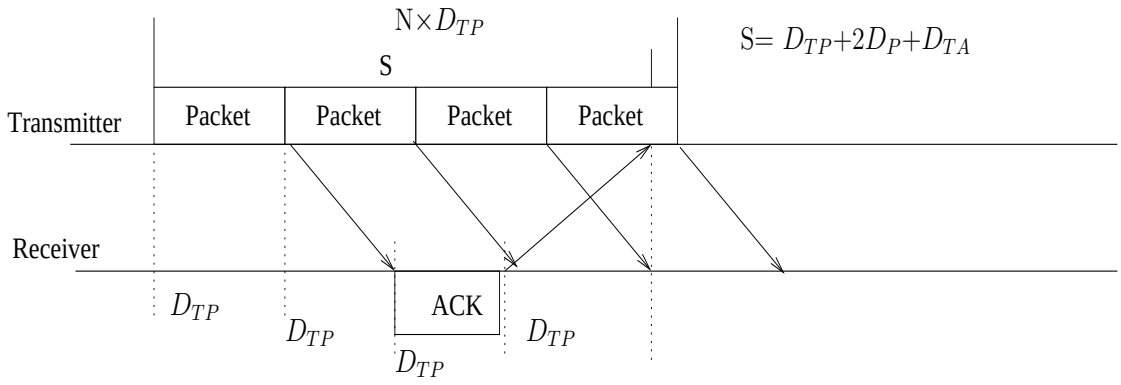


Figure 5.5: Timming issues in the go-back-N protocol.

high value to allow continuous transmission, while waiting for an ACK of the first correctly received packet within the window, where N has to satisfy:

$$N > S/D_{TP}. \tag{5.2}$$

- The go-back-N transmission protocol is portrayed in Fig. 5.5.
- In the absence of transmission errors the efficiency of the go-back-N protocol is

$$E = \min\{1, N \times D_{TP}/S\}. \tag{5.3}$$

- The go-back-N protocol does not require buffering of packets at the receiver.
- The transmitter has to buffer up to N packets while waiting for their ACK.
- The transmitter has to retransmit all the packets in the entire window in the event of an error.
- The receiver can only receive packets in their original order. In other words, the receiver cannot receive packet $(i + 1)$ before packet i.

Finally, the selective-repeat protocol may be described as follows [86]:

- It attempts to retransmit only those packets that are actually lost due to errors.
- The receiver must be able to accept packets out of order. Hence, the receiver of this protocol has to use a buffer.

- The receiver acknowledges every error-free packet, packets that are not acknowledged before a time-out are assumed to be lost or to be in error.
- An explicit NAK (selective reject) message can be transmitted to request the retransmission of a single packet.
- The window-based protocol is identical to that of the go-back-N protocol and the window size is W.
- In the ideal case, only erroneously received packets are retransmitted when the efficiency of the selective-repeat protocol may be calculated as:

$$E = 1 - P, \tag{5.4}$$

  where P is the probability of a packet error.

The proposed Hybrid-ARQ scheme is based on the selective-repeat protocol combined with FEC coding. There are two basic types of H-ARQ schemes, namely H-ARQ type I and H-ARQ type II [50], [88]. The transmitter of the H-ARQ type I scheme retransmits both the information part and the parity part of corrupted packets, when the transmitter receives a repeat request from the remote receiver. This process is typically implemented within the Media Access Control (MAC) layer. Naturally, retransmitting both the information and parity part of the packets reduces the achievable effective throughput. Hence, the H-ARQ type I [50] scheme is often replaced by the H-ARQ type II [88] arrangement. In the H-ARQ type II scheme, the information part and the parity part are sent together during the first transmission attempt. However, during the second transmission attempt additional parity information is transmitted. The receiver then uses the parity received during all transmissions to correct the information received.
Hybrid-Automatic Repeat reQuest (H-ARQ) conveniently amalgamates packet retransmission techniques with error-correction and error-detection algorithms in order to improve the performance of wireless communication systems over hostile channels [89], [90]. For the sake of achieving a low BER, H-ARQ aided modulation schemes were employed in [91], [8] which invoked various coded modulation techniques [92], [93] and [78]. Iterative decoding was used for exchanging extrinsic information between the FEC decoder and the bit-to-symbol demapper of the demodulator.

The SLT codes of Section 4.3 were proposed in [25], [26], [27], which achieved significant BER performance improvements for transmission over both noisy and fading channels.

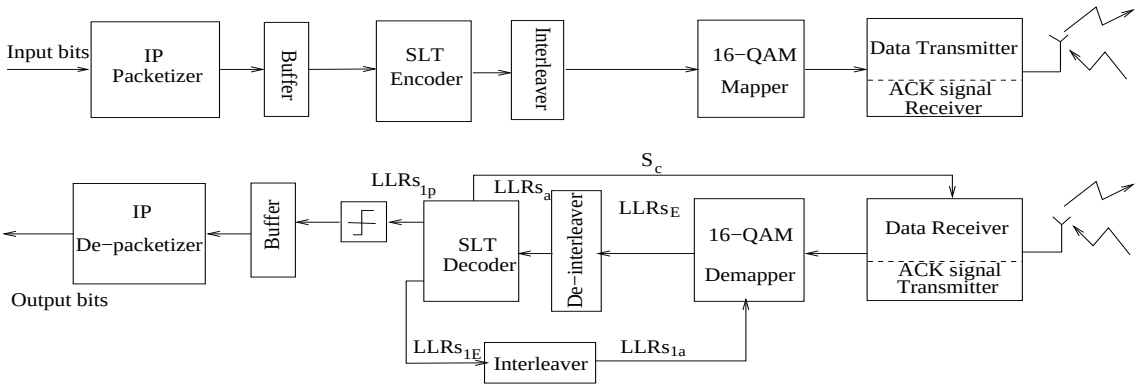


Figure 5.6: The block diagram of H-ARQ-SLT coded modulation scheme.

Against this background, the novel contribution of this section is an amalgamated H-ARQ SLT coded modulation scheme, which outperforms the classic H-ARQ aided coded modulation schemes of [94], [95] with the aid of using CRC to detect the erroneously received packets.

We used a modified version of the well-known H-ARQ Type II [88]. In the classic H-ARQ Type II scheme the transmitter implements the retransmission process by initially puncturing some bits of the parity and then incrementally transmitting additional portions of the punctured parity, whenever it receives a request for further parity from the receiver. The receiver then uses both the previously and the currently received parity portions in order to repeat the decoding process with a higher chance of decoding success. If the receiver is still unable to generate error-free decoded information, this process is still repeated until the maximum number of retransmissions is exhausted. By contrast, our simplified H-ARQ Type II scheme was implemented as follows. The receiver invokes the syndrome checking technique of [28] for detecting the legitimate codewords at the output of the SLT decoder. When an illegitimate codeword is deemed to be present at the output of the SLT decoder, the receiver requests the transmitter to retransmit the entire parity part of the illegitimate SLT codeword. The SLT decoder retains the previous LLRs of the information bits generated by the message passing between the information part and the previous parity part of the parity matrix H. The new parity part of the parity matrix H continues its message passing algorithm with the aid of the updated LLRs, until a legitimate SLT codeword is arrived at or the maximum number of iterations is exhausted.

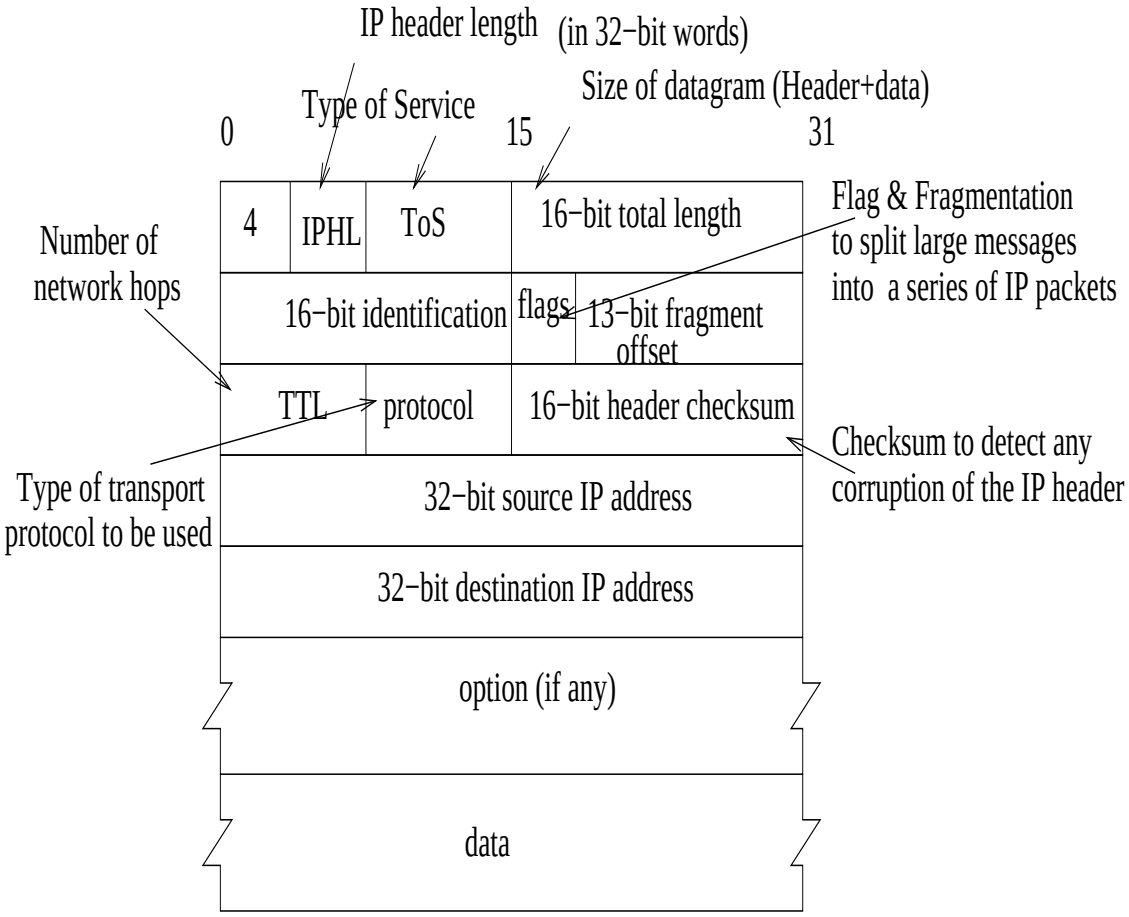


Figure 5.7: The IP structure.

5.1.2 H-ARQ-SLT Coded Modulation Scheme

As seen in Fig. 5.6 and Fig. 5.7, the source data is packetized by the Internet Protocol (IP) packetizer at the transmitter [96], before being passed to the SLT encoder block. A frame of k IP packets is buffered and then passed to the SLT encoder. Due to the fact that the number of bits in the IP header part is lower than in the data part, the IP header bits may be protected by strong SLT codes without substantially increasing the overheads. An example of the relation between the SLT packets and IP packets is shown in Fig. 5.8. Here, k bits of each SLT packet are arranged in the horizontal dimension and K bits of each IP packet are assigned in the vertical dimension. Each IP packet contains an L-bit header part and an S-bit data part, while K source SLT packets are formed from k IP packets. An SLT codeword is constituted by K SLT source bits representing an IP

source packet plus M number of SLT parity bits. The SLT codewords at the output of the SLT encoder are interleaved, where the interleaver has a length of N bits and then passed to the mapper for forwarding in terms of 4-bit symbols to the 16-Quadrature Amplitude Modulation (16-QAM) modulator, where N is the SLT codeword's total length. We term this serially concatenated system as the Hybrid-ARQ aided Systematic Luby Transform coded modulation (H-ARQ-SLT) scheme. The mapper of the 16-QAM modulator employs different types of mapping schemes, namely Gray mapping and set-partitioning based mapping [97], [98], as seen in Fig. 5.9(a), 5.9(b). The mappers are designed to attain the largest minimum Euclidean distance between phasor points having a Hamming distance of one and the smallest average number of nearest neighbor signal points [99]. Viewing the constellation from a different perspective, the average number of bits N_{min} that differs between two closest phasor points in the constellation is calculated as follows [99]

$$N_{min} = \frac{1}{L \cdot N_0} \sum_{s \in S} \sum_{s' \in S_s} d_H(s, s'), \tag{5.5}$$

where s denotes the specific phasor considered in a two-dimensional signal set S, s' represents a phasor in the neighboring signal subset S_s of the phasor s, $L = 2^m$ denotes the number of phasor points in the constellation, with m being the number of bits in a symbol of the constellation and finally, $d_H(s, s')$ denotes the Hamming distance between s and s'. Still referring to (5.5), N_0 physically represents the average number of nearest neighbor phasor points in the constellation and it is calculated as follows:

$$N_0 = \frac{1}{L} \sum_{s \in S} N_s, \tag{5.6}$$

where N_s is the number of nearest neighbor phasor points adjacent to s, i.e. the number of phasors in the subset and N_{min}= 1, when using the Gray mapper seen in Fig. 5.9(a).

Again, the modulation scheme is 16-QAM, hence each modulated symbol is represented by four bits. The transmitter is also equipped with an ACKnowledgement (ACK) receiver, while the receiver has an ACK transmitter in order to feed back the ACK signal to the transmitter. The ACK transmitter employed at the receiver side is controlled by the SLT decoder and the syndrome checking block of Fig. 5.6, as proposed in [28] in order to identify the error-free legitimate SLT codewords. When a legitimate codeword is detected at the output of the SLT decoder, the SLT decoder generates a signal S_c for the ACK transmitter of the receiver, which is then conveyed to the system's transmitter. By contrast, if the output of the SLT decoder is an illegitimate codeword, the SLT decoder activates the ACK transmitter of the receiver to send a Negative-ACK (NACK) signal to the system's

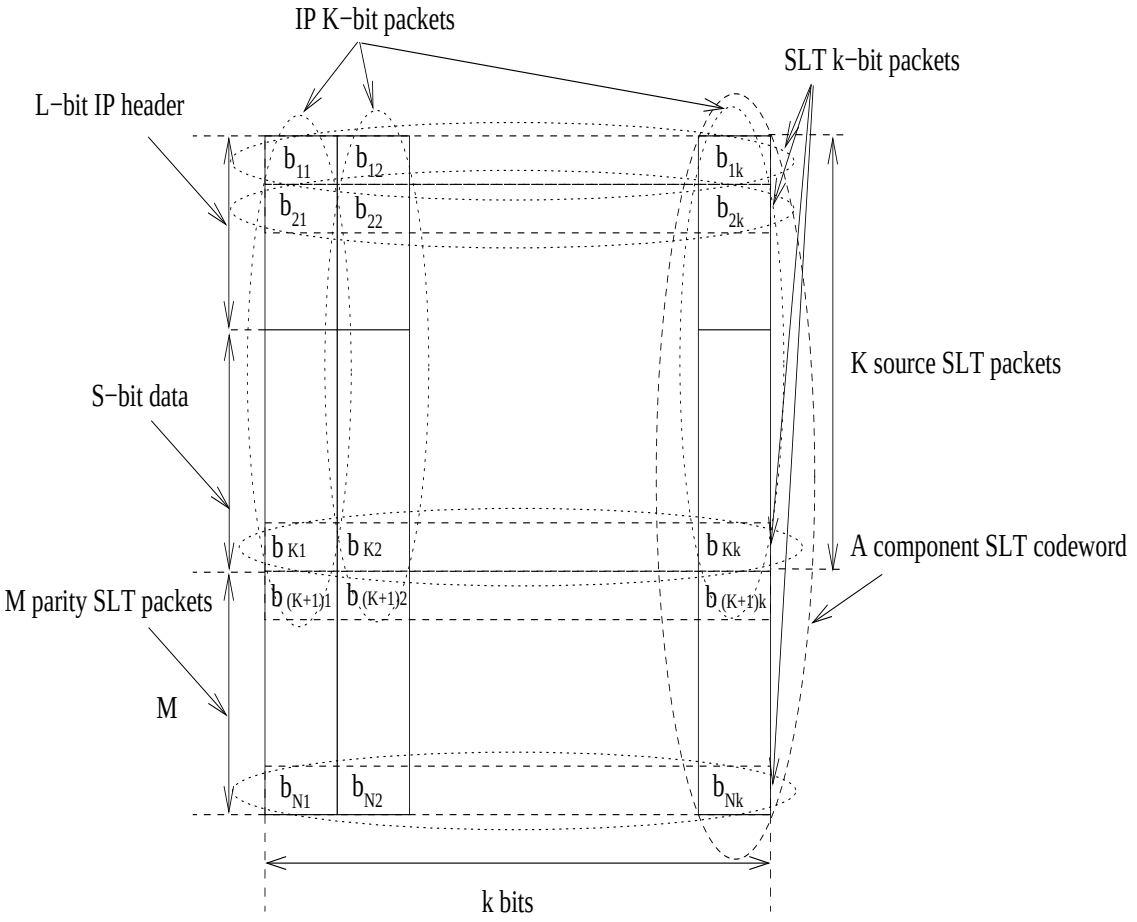


Figure 5.8: An Example of the relation between IP packets and SLT packets.

transmitter to request the retransmission of this codeword.

For each QAM symbol $x = f(b)$, where b is the m-bit sequence b= $(b_0,\ b_1, \cdots,\ b_{m-1})$, associated with the *a priori* LLRs spanning from LLR(b_0) to LLR(b_{m-1}) we have [30]:

$$LLR(b_j) = \log[P(b_j = 1)]/\log[P(b_j) = 0] \quad j = 0, \cdots, m\text{-}1. \tag{5.7}$$

The received signal y_k is represented as $y_k = a_k x_k + n_k$ [30], where n_k is the complex-valued zero-mean Gaussian noise process having a variance of $\sigma^2 = N_0/2$ per dimension. The notation a_k represents the complex-valued time-variant fading gain and x_k denotes the transmitted QAM symbol, when detection benefitting from coherent perfect channel knowledge is used. Then the conditional Probability Density Function (PDF) of the

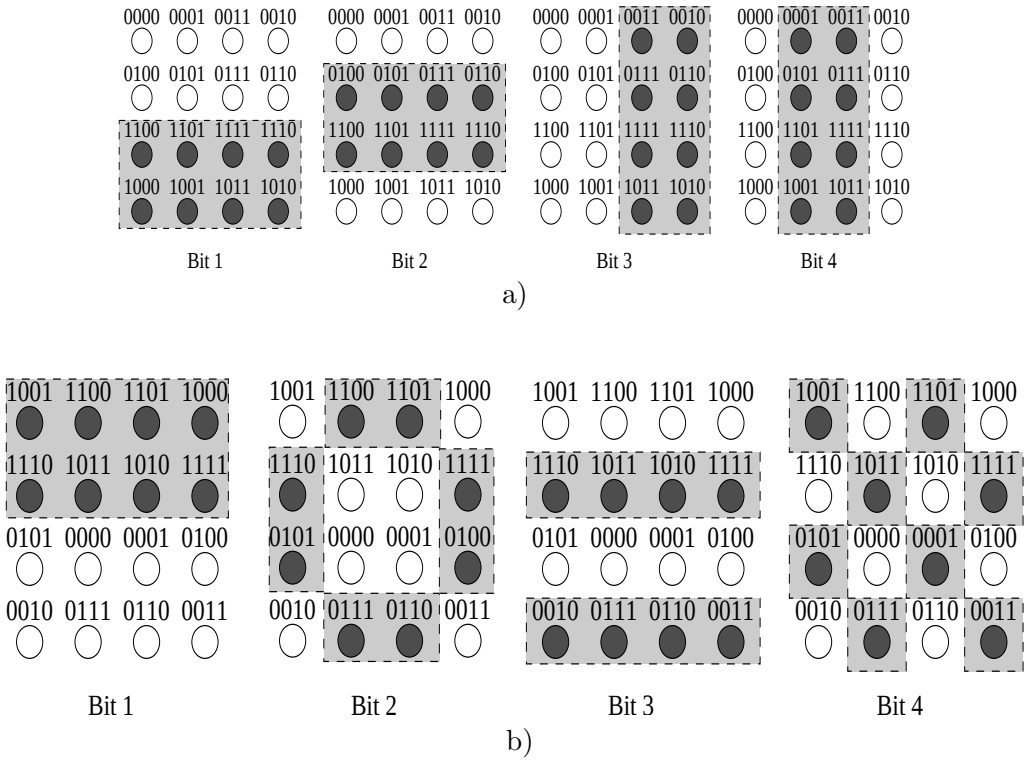


Figure 5.9: a) The constellation diagram of Gray mapping; b) The constellation diagram of Set-partitioning mapping.

received signal y may be calculated as follows [30]:

$$P(y|x,a) = \left(\frac{1}{\pi N_0})\exp(-(\frac{E_s}{N_0})|y-ax|^2\right). \tag{5.8}$$

Taking the logarithm of both two sides of (5.8), we arrive at the logarithmic-probability of [30]:

$$\log P(y|x,a) = \log\frac{1}{\pi N_0} - \frac{E_s}{N_0}|y|^2 - |a|^2\frac{E_s}{N_0}|x|^2 + 2|a|\frac{E_s}{N_0}(y_I x_I + y_Q x_q), \tag{5.9}$$

where the inphase and quadrature-phase transmitted signal x and received signal y can be represented as $x = x_I + jx_Q$ and $y = y_I + jy_Q$, respectively, and E_s/N_0 is the modulated symbol energy-to-noise ratio. If all bits b_j in the bit-sequence b are independent, we have:

$$\log P(x) = \log P(b) = \log P(b_0)P(b_1)\cdots P(b_{m-1}), \tag{5.10}$$

although in case of Gray-coding this is clearly not the case. From (5.8) and (5.10), we have the soft output of the demapper, expressed in terms of the LLRs as [30]:

$$\begin{aligned} \mathrm{LLR}(\hat{b}_j) &= \log\frac{P(b_j=1|y)}{P(b_j=0|y)} && (5.11)\\ &= \log\frac{\sum_{x:b_j=1}\exp\left[\log P(y|x,a)+\log P(x)\right]}{\sum_{x:b_j=0}\exp\left[\log P(y|x,a)+\log P(x)\right]} && (5.12)\\ &= \log\frac{\sum_{x:b_j=1}\exp\left[\gamma(y,x)\right]}{\sum_{x:b_j=0}\exp\left[\gamma(y,x)\right]}, && (5.13) \end{aligned}$$

where the receiver's estimate of x is denoted as $\hat{x}$ and we have $\hat{b} = (\hat{b_0}, \cdots, \hat{b}_{m-1})$, while the notation $\gamma(y,x)$ represents [30]:

$$\gamma(y,x) = -|a|^2\frac{E_s}{N_0}|x|^2 + 2|a|\frac{E_s}{N_0}(y_I x_I + y_Q x_Q) + \sum_{j=0}^{m-1} b_j \mathrm{LLR}(b_j). \quad (5.14)$$

The extrinsic information output is calculated as follows [30]:

$$\mathrm{LLR}_E(\hat{b}_j) = \mathrm{LLR}(\hat{b}_j) - \mathrm{LLR}(b_j). \quad (5.15)$$

The capacity of the SLT coded modulation scheme is denoted as C, and when transmitting over the AWGN channel and using 16-QAM, we have [78]:

$$C = \frac{1}{16}\sum_{x\in 16} I(x;y) = \frac{1}{16}\sum_{x\in 16}\int_{-\infty}^{\infty} P(y|x)\log_2\frac{P(y|x)}{P(y)}dy, \quad (5.16)$$

where $I(x;y)$ denotes the mutual information between the received signal y and the transmitted signal x, while $P(y|x)$ is given in (5.9) and a is a constant.

In the block diagram of the H-ARQ-SLT coded modulation scheme seen in Fig. 5.6, the demapper constitutes an inner decoder, while the SLT decoder acts as an outer decoder. Following deinterleaving the output extrinsic information LLRs gleaned from the demapper are fed to the input of the SLT decoder. These LLRs are processed by the inner SLT decoder. The extrinsix LLRs at the output of the SLT decoder are fed back to the demapper after interleaving, as seen in Fig. 5.6. If the syndrome checking block of the SLT decoder detects an illegitimate decoded codeword, it will send the control signal S_c to activate the ACK transmitter. An ACK will then be sent to the transmitter, requesting the transmission of extra parity. The receiver will continue the iterative decoding process. This retransmission process will then be repeated, until a legitimate SLT codeword is decoded or upon reaching the maximum number of retransmissions.

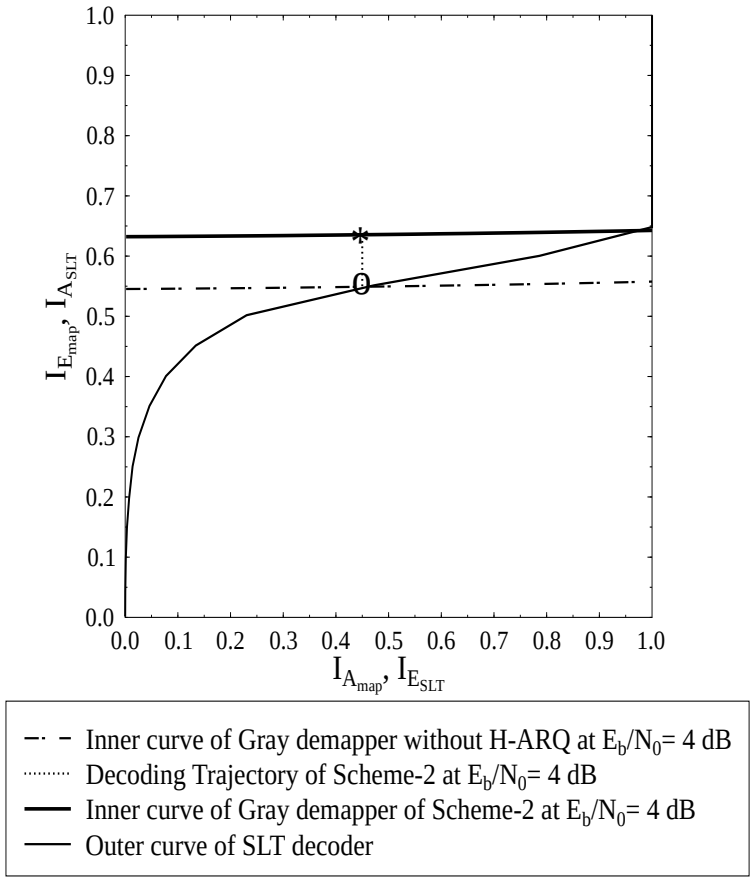


Figure 5.10: EXIT chart and the decoding trajectory of the H-ARQ-SLT coded modulation scheme using the classic Gray mapper (Scheme-2) of Fig. 5.9a.

5.1.3 EXIT Chart Analysis

In this section we use EXIT charts [70], [72] to analyse the convergence of the H-ARQ-SLT coded modulation scheme. The EXIT chart of the H-ARQ-SLT coded modulation scheme includes three curves in Fig. 5.10 and Fig. 5.11, namely the inner decoder's curve, the outer decoder's curve and the Monte-Carlo simulation based trajectory. The inner decoder's curve constitutes that of the demapper, the outer decoder's curve constitutes that of the SLT decoder and the decoding trajectory represents the extrinsic information exchange between the demapper and the SLT decoder.

The inner decoder's curve seen in Fig. 5.10 and Fig. 5.11 visualises the *apriori* input mutual information $I_{A_{map}}$ and output extrinsic information $I_{E_{map}}$ relationship of the demapper. By contrast, the outer decoder's EXIT curve seen in Fig. 5.10 and Fig. 5.11 represents

the relationship between the input mutual information $I_{A_{SLT}}$ and the output mutual information $I_{E_{LST}}$ of the SLT decoder of Fig. 5.6. If there is no retransmission between the transmitter and receiver of the H-ARQ-SLT coded modulation scheme, then the *a priori* LLRs of the SLT decoder are exactly the same as during the previous iteration, because the extrinsic LLRs of the demapper remain unchanged. Hence, the decoding trajectory of the H-ARQ-SLT coded modulation scheme represents the relationship between the *a priori* LLR input of the demapper and the extrinsic LLR output of the SLT decoder after l iterations. The output mutual information obeys the following function [27] [72]:

$$I_{E_{SLT}} = f\left(I_{E_{map}}, I_{A_m}\right), \tag{5.17}$$

where $I_{A_m} = I(X;R)$ is the average *a priori* information at the input of the SLT decoder's message node and R is the LLR message gleaned from the check nodes of the SLT decoder.

When there is a retransmission between the transmitter and the receiver of the H-ARQ-SLT coded modulation scheme, the decoding trajectory of the system is a function of both the input mutual information of the demapper as well as of the output mutual information of the SLT decoder and of the mutual information $I_{A_{retrans}}$ input to the SLT decoder. The retransmission-aided extra input mutual information $I_{A_{retrans}}$ of the SLT decoder is related to the number of retransmissions.

Again, Fig. 5.10 shows the corresponding EXIT curves and the decoding trajectory of the H-ARQ-SLT coded modulation scheme at $E_b/N_0 =$ 4 dB, when using 16-QAM and the classic Gray mapper. As seen in Fig. 5.10, the dashed-dotted line represents the inner EXIT curve of the original Gray mapper at E_b/N_0=4 dB, while the bold-continuous line marks the EXIT curve of the Gray mapper in the H-ARQ-SLT coded modulation scheme. Initially, the H-ARQ-SLT receiver attempts to iteratively decode the received codewords by exchanging extrinsic information between the SLT decoder and the demapper. However, the trajectory reaches the point of intersection between the two EXIT curves after a single iteration, which is marked by a circle, as seen in Fig 5.10. Hence, we cannot achieve an infinitesimally low BER. In the presence of SLT errors the H-ARQ-SLT receiver requires the H-ARQ-SLT transmitter to retransmit the parity part of the corrupted codewords. At this time, the EXIT curve of the inner mapper is moved upward, as indicated by the bold line seen in Fig. 5.10. The *a priori* LLRs at the input of the SLT decoder are gleaned from the extrinsic LLRs of the previous decoding iteration at the output of the SLT decoder. The receiver decodes again the received codewords and hence the decoding trajectory now reaches the point (0.64, 1), as seen in Fig. 5.10. The EXIT curve of the outer SLT code

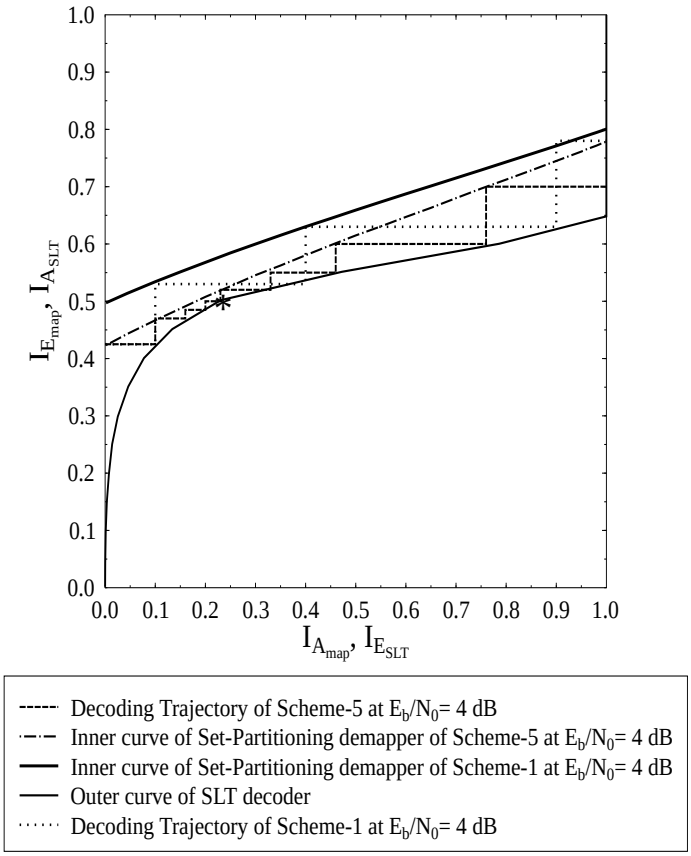


Figure 5.11: EXIT chart and the decoding trajectory of the H-ARQ-SLT coded modulation scheme using the set-partitioning mapper (Scheme-1) of Fig. 5.9b.

having a code rate of 0.5 [28] has a specific form, as seen in Fig. 5.10 and after four iterations the H-ARQ aided SLT decoder becomes capable of achieving BER$\leq 10^{-5}$ at E_b/N_0= 4 dB, as seen in Fig. 5.12 and Fig. 5.13.

Let us now embark on improving the achievable system performance by creating a more beneficial EXIT-curve shape for the inner demapper with the aid of set-partitioning. The EXIT chart of the H-ARQ-SLT coded modulation scheme using the set-partition demapper is portrayed in Fig. 5.11. The original EXIT curve of the inner set-partitioning mapper is plotted by the dashed-dotted line, while the EXIT curve after one retransmission is represented by the bold-thick line seen Fig. 5.11. If the receiver using the set-partitioning based demapper without the aid of retransmission invokes only four decoding iterations between the SLT decoder and the demapper, the decoded output codewords cannot achieve an infinitesimally low BER, because the trajectory's evolution is curtailed at the point

marked by the asterisk seen in Fig. 5.11. For the sake of achieving BER$\leq 10^{-5}$, the receiver has to allow for upto 8 iterations.

We can see by comparing Fig. 5.10 and Fig. 5.11 that at $E_b/N_0 = 4$ dB the H-ARQ-SLT coded modulation scheme using the Gray mapper requires a lower number of iterations for achieving BER$\leq 10^{-5}$ in comparison to the H-ARQ-SLT coded modulation scheme using the set-partitioning mapper. By contrast, the intercept point of the two EXIT curves is closer to the (1,1) point of perfect convergence, where an infinitesimally low BER can be achieved, if the associated higher number of decoding iterations and the ARQ-aided retransmission of the set-partitioning aided scheme are deemed affordable.

5.1.4 Simulation Results

SLT code rates r	1/2
Parity packet degree distribution	Truncated Degree Distribution proposed in [26]
The maximum number of inner iterations I_{max} of SLT codes	12 and 20
The number of outer iterations	2 and 4
Number of bits	10×1200×165
Modulation	16-QAM
Channel type	AWGN channel
Mapping	Gray mapping Set-partitioning mapping
H-ARQ	Modified Type II
The maximum number of retransmitting times	1
H-ARQ aided SLT coded modulation using set-partitioning mapping scheme	Scheme-1
H-ARQ aided SLT coded modulation using Gray mapping scheme	Scheme-2
H-ARQ aided SLT scheme	Scheme-3
H-ARQ scheme	Scheme-4
SLT coded modulation using set-partitioning mapping scheme	Scheme-5
SLT scheme	Scheme-6

Table 5.1: System parameters.

Our simulation parameters used in Scheme-1, Scheme-2, Scheme-3, Scheme-4, Scheme-5 and Scheme-6 are seen in Table 5.1. Fig. 5.12 and Fig. 5.13 represent the BER performance of the H-ARQ-SLT coded modulation scheme using the parameters of Table 5.1. The left vertical axis represents the BER values, while the right vertical axis represents normalized throughput values and the horizontal axis the E_b/N_0 values. The normalized throughput values are calculated by normalizing the effective throughput by the throughput[1] of the H-ARQ-SLT coded modulation scheme, recorded at high E_b/N_0 values, i.e. when the data transmission is error-free. The SLT code rate of the scheme is 0.5, hence the maximum throughput of the 16-QAM H-ARQ-SLT coded modulation scheme is 2 bits/symbol, which represents our throughput normalization factor. As seen in Fig. 5.12 and Fig. 5.13, the continuous line marked by the hollow diamond represents the BER versus E_b/N_0 performance of the H-ARQ-SLT coded modulation scheme, which attains BER $< 10^{-5}$ for E_b/N_0 values in excess of 3.8 dB. An SLT coded modulation scheme dispensing both with ARQ assistance and with the required feedback link is used as one of our benchmarkers. The BER performance of this scheme is represented by the continuous line marked by the hollow square in Fig. 5.12, which requires an E_b/N_0 in excess of 4.5 dB to achieve BER $< 10^{-5}$. Another benchmark scheme we employed in Fig. 5.12 was based on separate SLT coding and 16-QAM[2], which was marked by the hollow circles. When we briefly compare the performance of the three schemes in Fig. 5.12 in terms of their achievable throughput, a near-unity throughput is achieved by the amalgamated H-ARQ-aided SLT coded modulation scheme for E_b/N_0 values in excess of about 4 dB. The other two schemes exhibit both a higher BER and a lower normalized throughput, indicating the superiority of the proposed scheme. Let us now focus our attention on Fig. 5.13, where we repeated the H-ARQ-aided SLT coded modulation scheme's performance curve from Fig. 5.12. Furthermore, the curves marked by the triangle represent the scenario, where the joint SLT and 16-QAM coded modulation scheme was replaced by separate SLT and 16-QAM schemes. Finally, the curves denoted by the circled-plus-sign represent a H-ARQ-aided but uncoded 16-QAM benchmarker, which hence has a 4 bits/symbol effective throughput. In the high-SNR region the throughput of this H-ARQ-aided but uncoded 16-QAM benchmarker is twice as high as that of the 0.5-rate SLT-coded system. By contrast, in the low-SNR region the normalized throughput of the uncoded H-ARQ-

[1]The effective throughput is defined as the ratio of the number of correct bits per the total number of transmitted bits.

[2]Separate SLT coding and 16-QAM is implemented by dispensing with the iterative decoding process exchanging extrinsic information between the SLT decoder and the demapper.

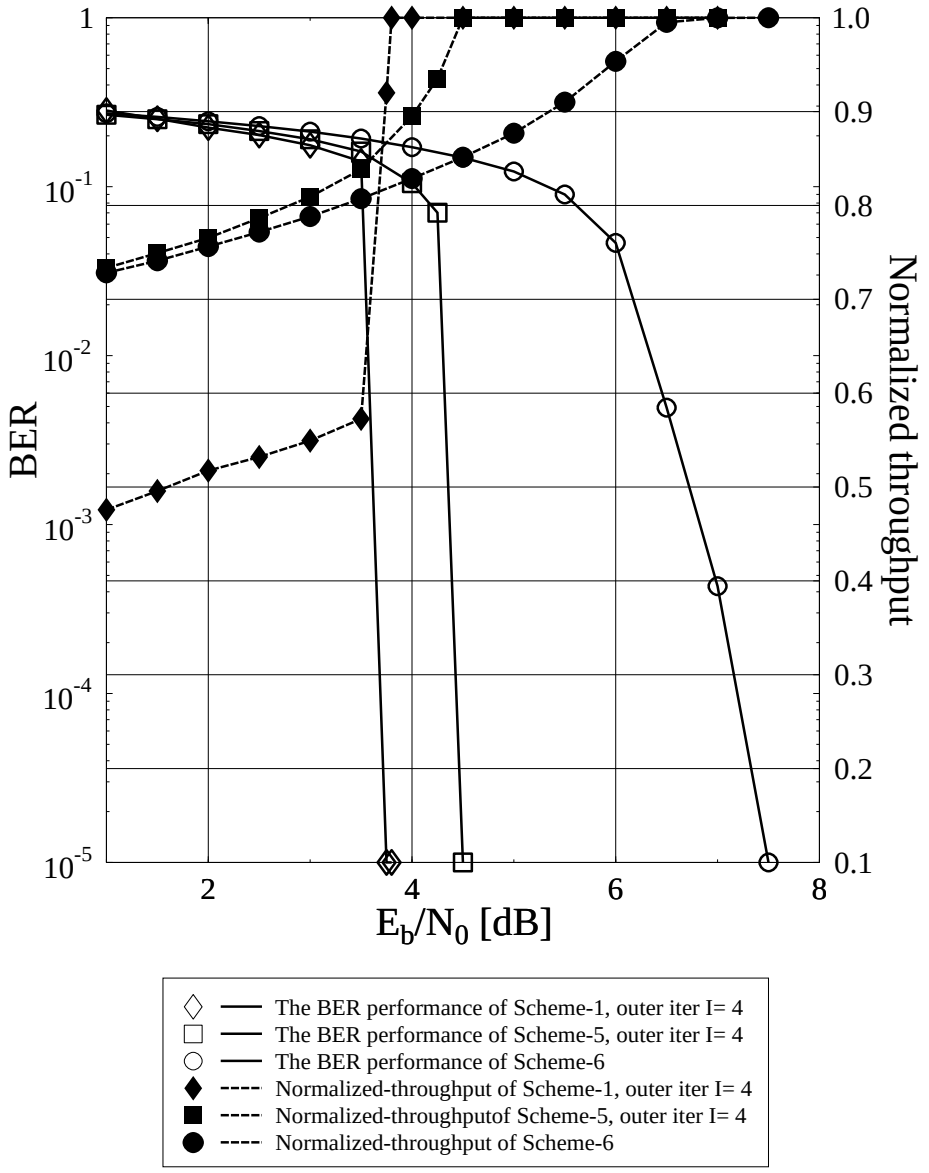


Figure 5.12: BER performances and normalized throughputs of Scheme-1, Scheme-5 and Scheme-6 seen in Table 5.1, having a maximum of I= 20 SLT decoder iterations, when transmitting data over an AWGN channel using 16-QAM.

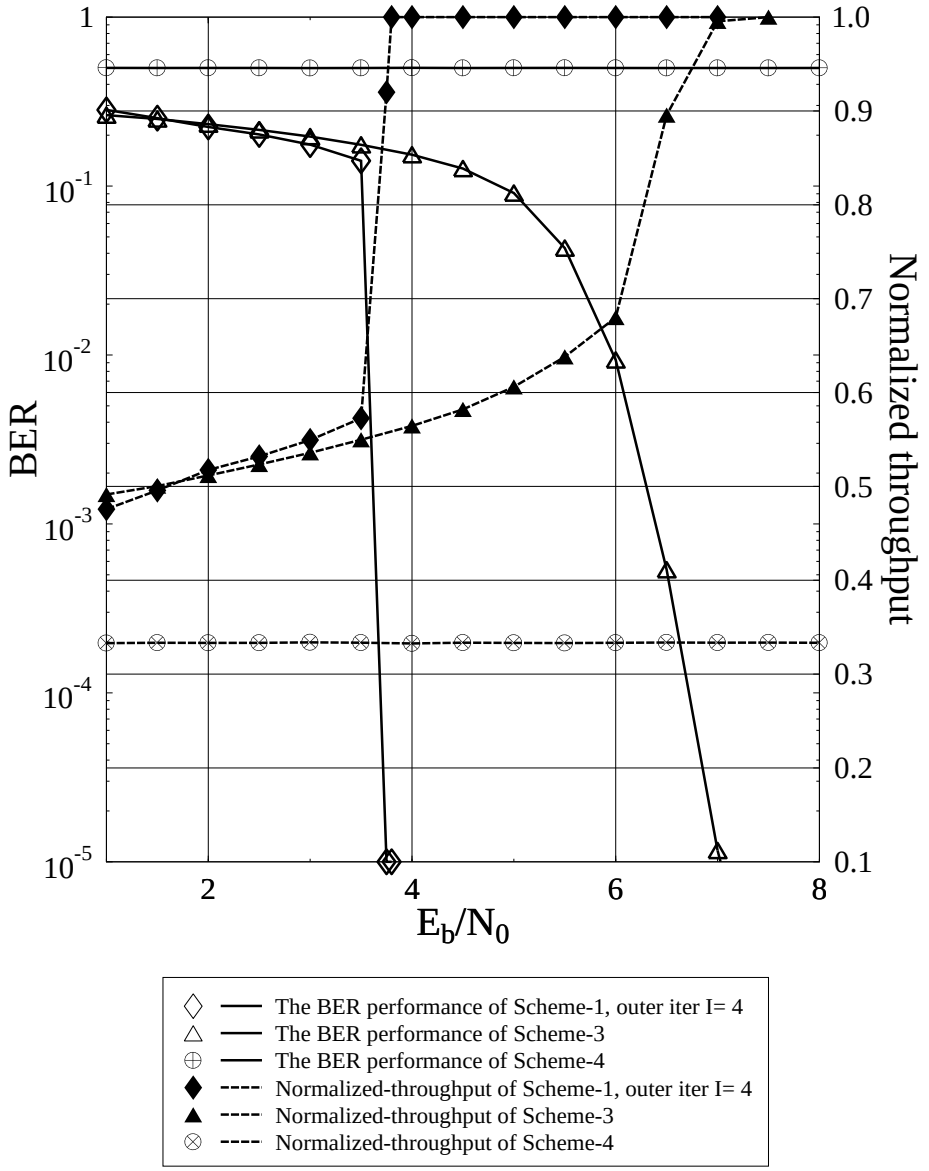


Figure 5.13: BER performances and normalized throughputs of Scheme-1, Scheme-3 having a maximum of I= 20 SLT decode iterations and Scheme-4 seen in Table 5.1, when transmitting data over an AWGN channel using 16-QAM. Note that while the joint SLT coded 16-QAM scheme exchanges extrinsic information, the separate scheme does not, hence iterations are only carried out within the SLT decoder. The anonym SP represents set-partitioning in the context of the schemes represented by the diamond and square legends, while Gray-mapping was used by the separate SLT and 16-QAM-aided scheme.

aided scheme becomes ~ 0.33, because in this region the receiver is unable to attain an infinitesimally low BER at the output, hence it requires the maximum affordable number of retransmissions. The H-ARQ-SLT scheme using separate SLT and 16-QAM arrangements has an approximately 3 dB higher E_b/N_0 requirement than our proposed scheme, as indicated by the continuous line marked by the triangle in Fig. 5.13. The ARQ scheme operating without the assistance of the SLT code requires an E_b/N_0 value in excess of 18 dB to attain BER $< 10^{-5}$, although this BER value is not reached within the E_b/N_0 range shown in Fig. 5.13, hence indicating BER≈ 0.5.

Finally, in Fig. 5.14 we compare the set-partitioned and Gray-coded H-ARQ-aided SLT-16-QAM schemes. Observe that the Gray-coded scheme performs better both in terms of its BER and normalized throughput, when using two iterations between the demapper and the SLT decoder.

5.2 Chapter Conclusions

The proposed H-ARQ-SLT aided scheme is capable of achieving an E_b/N_0 gain ranging from 0.5 dB upto 3.5 dB and a higher throughput in comparison to the benchmark schemes. The H-ARQ-SLT scheme advocated is also capable of attaining an infinitesimally low BER at a low E_b/N_0 value, while maintaining a high normalized throughput. By exploiting the syndrome checking capability of the SLT decoder we can reduce the complexity of the scheme, while simultaneously eliminating the redundant bits used for checking the parity of the IP packets in classic CRC schemes. Furthermore, it was shown that the H-ARQ-SLT coded modulation scheme using the Gray mapper performs better in terms of its BER, while imposing a lower complexity and achieving a higher throughput in comparison to the set-partitioning mapper, which was proposed in [28].

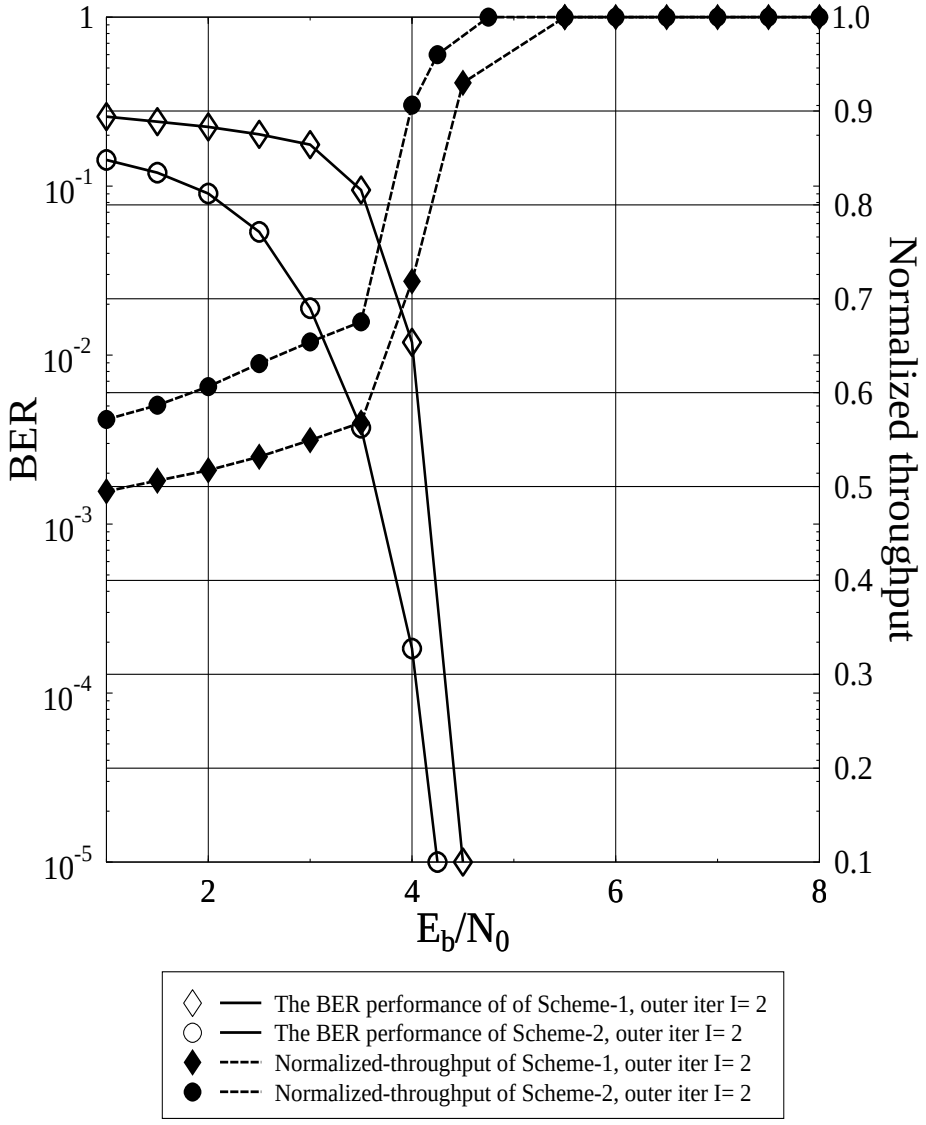


Figure 5.14: BER performances and normalized throughputs of the H-ARQ-SLT system having a maximum of $I=$ 12 SLT decoder iterations, when transmitting data over an AWGN channel using 16-QAM and employing either SP (Scheme-1) or Gray-coding (Scheme-2).

Chapter 6

Conclusions and Future Work

In this concluding chapter, a summary of the thesis and the main findings of our investigations will be presented. This will be followed by a range of future research ideas.

6.1 Summary and Conclusions

This thesis investigated the application of Fountain codes especially for the BEC channel and wireless Internet channels. Five type of erasure codes are continuously mentioned in the Chapter 2 which are Tornado codes, random linear codes, Luby transform codes, Raptor codes and systematic Raptor codes. A general concept of Tornado codes is introduced in Section 2.1.1. The concept of erasure decoding is rooted from the decoding process of the Tornado code. This erasure decoding process is implemented by finding, XOR-adding and erasing the edge connecting between encoded packets. The performance of Tornado codes is not good when the error happen in the information part of the code words because of the encoding structure based on the assumption that erasure events only happen in the parity part of the tornado code words. Even Tornado codes are not good in implementing, but analysing their encoding and decoding processes help us having a basic to analyse the Fountain code later in Section 2.2. Section 2.2 presents four types of fountain codes which are random linear codes, Luby transform codes, raptor codes and systematic codes. The random linear code is analysed in Section 2.2.1 to expose the encoding and decoding methods used in fountain codes. A history and some basic concepts of Luby transform codes are given in Section 2.2.2. In this subsection, the generator matrix creating process as well as the decoding process are dissected more clearly. Section 2.2.3 shows a new ver-

sion of fountain codes which was proposed by Shokrollahi et al in 2002. The architecture of raptor codes was produced and some concepts such as Tanner graphs, pre-codes, e.t.c are given in this subsection. The advantages and disadvantages of raptor codes are also analysed in this subsection. Section 2.2.4 outlines the systematic raptor codes, a special version of the raptor codes, and its applications in communication systems. The degree distribution designed for both Luby transform codes and raptor codes is very improtant and it will be the main key of Section 2.2.5. In other words, the performance as well as the complexity of both Luby transform codes and raptor codes are dependant on the degree distribution design. The robust soliton degree distribution is analysed in Section 2.2.5.1 and the Poisson degree distribution is introduced in Section 2.2.5.2 let we know about the condition and meaning of each parameters of each degree distribution.

Chapter 3 presents the novel contributions as well as applications of Luby transform codes using the hard bit decoding algorithm in communication models. Section 3.1 proposed a novel improvement of the degree distribution and introduces an improved degree distribution known as the Improved Robust Soliton Degree Distribution. The improved robust soliton degree distribution supplements the shortcoming of the original robust soliton distribution. As seen in this section, we showed out the unfeasability of the probability of events having an appearing times smaller than 1. By supplementing a new factor termed as $\nu(d)$ in the degree distribution, it makes the packet error rate performance of Luby transform codes becomes better than the Luby transform codes using the original robust soliton degree distribution as seen in Figure 3.4 and Figure 3.5. A concatenated scenarios of Luby transform codes and BICM-ID codes is proposed in Section 3.1.2 and its performances in the BEC contaminated AWGN and uncorrelated fading channels are given in Section 3.1.3. For the sake of improving the performance of communication systems using Luby Transform codes over Wireless Internet Channels(WICs) with the aid of exchanging the feedback information between Luby Transform codes and other block codes, a new scenario of Luby Transform and Generalize-Low Density Parity Check(G-LDPC) codes is presented in Section 3.2. The purpose of this scenario is that the packet error rate acknowledgement is very important for decoding process of Luby transform codes. Knowing the error rate of each received packets helps the Luby transform decoder reject contaminated packets from the decoding process to avoid propagating errors when recovering source information packets from contaminated packets. The contribution of this Section is that the author gave out a new reliable Amalgamated Luby Transform codes and G-LDPC codes scheme and using a clever G-LDPC code word mapping into LT packets. This scheme

improves further PER and BER performances of Luby transform codes in noisy channels. The simulation results are given in Section 3.2.2. A novel random integer index generator for the degree distribution of source information packets of Luby transform codes. A bit by bit decoding method and hard bit decoding method based on the LLRs estimation of BICM-ID codes applied for Luby transform codes, are presented in Section 3.3. The degree distribution design is not only important for the Luby Transform encoded packets, but also for the input packets. A novel random integer generator termed as the Swapping Bit Random Integer Generator (SBRIG) as well as hard bit by bit decoding based on the LLRs estimation of constituent BICM-ID codes are proposed for the amalgamated Luby transform and BICM-ID code scheme to achieve further improvement BER performance of the Luby transform codes in noisy channel. The performances of this scheme are given in Section 3.3.3. Section 3.4 proposes the Luby Transform Coding Aided Iterative Detection for Downlink SDMA Systems, which invokes a low-complexity near-Maximum-Likelihood (ML) Sphere Decoder (SD). The Ethernet-based Internet section of the transmission chain inflicts random packet erasures, which is modelled by the Binary Erasure Channel (BEC), while the wireless downlink imposes both fading and noise. A novel Log-Likelihood Ratio based packet reliability metric is used for identifying the channel-decoded packets, which are likely to be error-infested. Packets having residual errors must not be passed on to the LT decoder for the sake of avoiding LT-decoding-induced error propagation. The proposed scheme is capable of maintaining an infinitesimally low packet error ratio in the downlink of the wireless Internet for E_b/N_0 values in excess of about 3dB. The PER and BER performances of this scheme are given in Section 3.4.4.

Chapter 4 proposes the Soft bit decoding algorithm for systematic LT codes and their performance in the wireless Internet and MIMO channels. For the sake of improving further the performance and making LT codes more flexible in the communication systems we introduced a novel soft bit decoding algorithm in Section 4.1. From the analyse in Section 4.1 we modified the systematic Luby transform codes for applying the soft bit decoding algorithm in Section 4.2. Analysing the performance of SLT codes base on EXIT charts is implemented in Section 4.2.3. The BER performances of SLT code using the soft bit decoding algorithm over AWGN-contaminated BEC channels are given in Section 4.2.4. A conclusion is given in Section 4.2.5. Section 4.3 presents a novel improved degree distribution termed as Truncated Robust Soliton Degree Distribution applied for the degree of the encoded SLT packet and a new conditional random integer generator applied for the degree distribution of the SLT input packets. Again, the performance of

the new version of SLT codes is analysed by EXIT charts in Section 4.3.4 and its BER performances are given in Section 4.3.5. Section 4.3.6 concludes that SLT codes using both the proposed truncated degree distribution and the conditional random integer generator achieved a high performance for transmission over various combinations of the BEC, AWGN and Rayleigh fading channels. The system using the SLT(1200,3600) code requires low E_b/N_0 values for attaining output BERs lower than 10^{-5}, when communicating over the different channels. Quantitatively, this system is capable of achieving BER $\leq 10^{-5}$ for an E_b/N_0 value around 1.5 dB over the AWGN channel and at 3.5 dB over the Rayleigh fading channel. By contrast, the corresponding benchmark system using the Low Density Parity Check code LDPC(1200,3600) requires an E_b/N_0 value upto 2.5 dB to achieve BER $\leq 10^{-5}$ over the AWGN channel. Section 4.4 investigates the BER performance of SLT codes using different degree distribution and different random integer index generating algorithm and the section conclusion is given in Section 4.4.3. Section 4.5 proposes the SLT coded V-BLAST system and BER performance of the SLT coded V-BLAST, when transmission over MIMO channels. A short conclusion of this section is given in Section 4.5.5. The chapter conclusions are detailed in Section 4.6.

Chapter 5 presents the H-ARQ SLT coded modulation and H-ARQ irregular SLT coded V-BLAST schemes. An introduction of conventional H-ARQ schemes is provoked in Section 5.1. For the sake of achieving high throughput and BER performance of SLT codes in communication systems we proposed the H-ARQ SLT coded modulation in Section 5.1.2. The proposed H-ARQ SLT coded modulation scheme in Section 5.1.2 achieves the best BER performance as well as the highest throughput in comparison to other schemes such as the SLT coded modulation without H-ARQ, the H-ARQ aided SLT and the conventional H-ARQ schemes. Section 5.1.3 shows the EXIT chart analysis of the H-ARQ SLT coded modulation scheme using the Gray mapping and the Set-Partitioning mapping. Section 5.1.4 presents the BER performance and throughput of the proposed H-ARQ SLT coded modulation scheme. Finally, the chapter conclusion is given in Section 5.2. The parameters used in the sections are listed as follows:

The Parameters of Simulation Results in Sections	
Parameters of simulation results in Section 3.1.1	
The number of source packets K	10,000
The IRSDD parameters in (3.6)	
Fig. 3.4	δ= 0.5, c= 0.1.
Fig. 3.5	δ= 0.5, c= 0.03.
Parameters of simulation results in Section 3.1.2	
RS codes	C(7,3,2)
LT codes	C(10,000, 13,000)
LT-BICM-ID codes	C(30,000, 52,000)
BEC erasure probability	0.1
Modulation	16QAM
Number of source packets K	10,000
Number of transmitted packets K'	13,000
Number of bits per source packets $P.n$	165

Table 6.1: System parameters used in Section 3.1.1 and Section 3.1.2.

Parameters of simulation results in Section 3.2	
Erasure probability	P_e= 0.1
LT code parameters	δ= 0.5 c= 0.1
Number of bits per LT packet	168
Number of source packets K	10,000
Number of encoded LT packets N	13,000
G-LDPC component codes	BCH(15,11,1)
Component code rate of G-LDPC	$r = \frac{11}{15}$
Code rate of G-LDPC	$\frac{7}{15}$
Modulation	BPSK
Parameters of simulation results in Section 3.3	
BICM-ID code rate	r=3/4
Modulation	16QAM
Number of source packets	10,000
Number of transmitted packets	13,000
Number of bits per source packet	165
BEC erasure probability	P_e=0.1
LT degree of distribution	IRSDD

Table 6.2: System parameters used in Section 3.2 and Section 3.3.

Parameters of simulation results in Section 3.4	
LT Packet size	120 bits
LT Distribution	IRSDD [20]
Number of source packets K	10,000
Number of redundancy packets $N-K$	3,000
Erasure probability of BEC P_e	0.1
Recursive Systematic Code	RSC(5,7)
Interleaver length	10^5 bits
Modulation	4 QAM
Number of users K'	3
Number of transmit antennas M'	6 for $L_s = 1$, 8 for $L_s = 1.333$
Number of receive antennas N_k	2
Normalized Doppler frequency	$f_d = f_D \cdot T = 0.001$
Parameters of simulation results in Section 4.2.2	
Erasure probability P_e=	0.0; 0.1
Parameters of IRSDD in Eq.(4.24)	δ= 0.5; c= 0.1
The number of source SLT packets K	500
SLT packet size	1000 bits
SLT code rates r	1/3
Modulation	QPSK
Noise channel	AWGN
Number of the SLT code's iteration	0, 2, 4, 6

Table 6.3: System parameters used in Section 3.4 and Section 4.2.2.

Parameters of simulation results in Section 4.3	
The TDD's parameters in (4.24)	δ= 0.5 c= 0.1
The total number of source information packets	1200 × 100
SLT packet size	165 bits
SLT(1200,2400), SLT(1200,3600) SLT(1200,1800)	$\gamma = 2$ $\gamma = 3$
LDPC(1200,$\frac{1200}{r}$) with code rate r	1/3, 1/2, 2/3
Maximum number of iterations	20
The erasure probability P_e	0.0 and 0.1
Modulation	QPSK
Channel types	BEC, AWGN and uncorrelated non-dispersive Rayleigh channels
Parameters of simulation results in Section 4.5	
Parameters of TDD in (4.24)	δ= 0.5 c= 0.1
SLT code rates r	1/2
γ	2
The maximum number of inner iterations $Imax_{inner}$ of SLT codes	30
Modulation	QPSK-set partition mapping
Parameters V-BLAST	
Number of transmit antennas T_x	4
Number of receive antennas R_x	4
Iterative lengths	2,400 or 24,000

Table 6.4: System parameters used in Section 4.3 and Section 4.5.

Parameters of simulation results in Section 5.1	
SLT code rates r	1/2
Parity packet degree distribution	Truncated Degree Distribution proposed in [26]
The maximum number of inner iterations I_{max} of SLT codes	12 and 20
The number of outer iterations	2 and 4
Number of bits	10×1200×165
Modulation	16-QAM
Channel type	AWGN channel
Mapping	Gray mapping Set-partitioning mapping
H-ARQ	Modified Type II
The maximum number of retransmitting times	1
H-ARQ aided SLT coded modulation using set-partitioning mapping scheme	Scheme-1
H-ARQ aided SLT coded modulation using Gray mapping scheme	Scheme-2
H-ARQ aided SLT scheme	Scheme-3
H-ARQ scheme	Scheme-4
SLT coded modulation using set-partitioning mapping scheme	Scheme-5
SLT scheme	Scheme-6

Table 6.5: System parameters used in Section 5.1.

6.2 Suggestions for Future Work

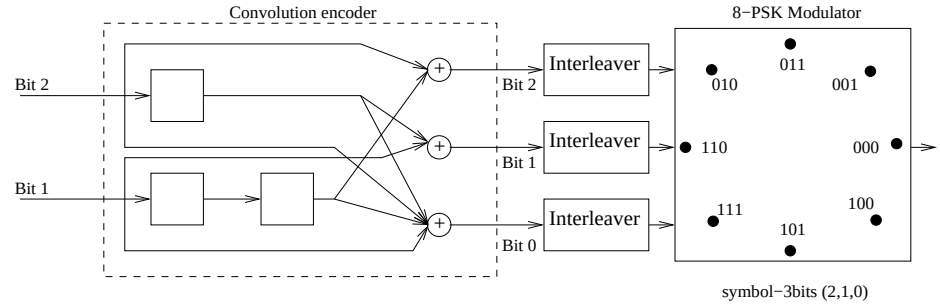
- Designing an Irregular Systematic Luby Transform (IrSLT) code and optimise its degree distribution using EXIT charts. The IrSLT is constituted of two parallel constituent SLT codes which exchange extrinsic information using iterative detection, like classic turbo codes [100].
- Combining Hybrid Automatic Repeat reQuest (H-ARQ) with the proposed IrSLT, while employing V-BLAST for the transmission of data.
- The SLT code's performance is promising as demonstrated in Chapter 4 and Chapter 5. Therefore, the application of SLT codes in communication systems may be beneficial.
- The SLT code with an adaptive degree distribution applied for different communication channel models also promises a better BER performance of SLT codes.
- The implementation of LT and SLT codes in the mobile ad-hoc network is also needed to be investigated. The mobile ad-hoc network is a kind of wireless ad-hoc network, it is a self- configuring network containing mobile routers connected by wireless links and its topology may change rapidly and unpredictably causing a very complicated communication environment. The transmitted data is affected from either the erasure events and the noise of this wireless Internet channel. Hence, an adaptive SLT codes are needed to be designed for applying in the mobile ad-hoc network.

Appendix A

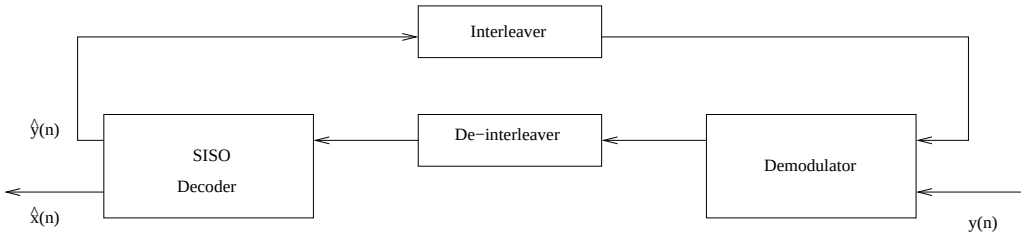
BICM-ID stands for Bit Interleaved Coded Modulation - Iterative decoding. The basic of BICM encoder is the concatenated scheme between an binary encoder having a codeword $\mathcal{C}$ and an $\mathcal{N}$-dimensional memoryless modulator creating a set $\mathcal{X}$ of symbols having a size of m bits, through a bit interleaver. The output bit sequence of the encoder is interleaved bit by bit and mapped to the symbol $\mathcal{X}$. Finally, the resulting signal sequence x at the output of the modulator is transmitted over channels. At the receiver side, the output LLRs of the soft bit demodulator decoder are de-interleavered and passed to the binary decoder. The output LLRs of the binary decoder are interleaver and fed back to the demodulator input. The scenario of a BICM system using convolution codes, is portrayed as in Figure A.1.

For the sake of increasing the Euclidean distance between any two points in the constellation map, instead of using the Gray mapping we use the set partition mapping. The Gray mapping and set-partition mapping are portrayed in Figure A.2.

Clearly, the set partition mapping creates the minimum Euclidean distance bigger than that of the gray mapping for both Bit2 and Bit1 as seen in figure A.2.



a) BICM encoder using convolution code and Gray mapping



b) BICM−ID decoder

Figure A.1: An example of a BICM system

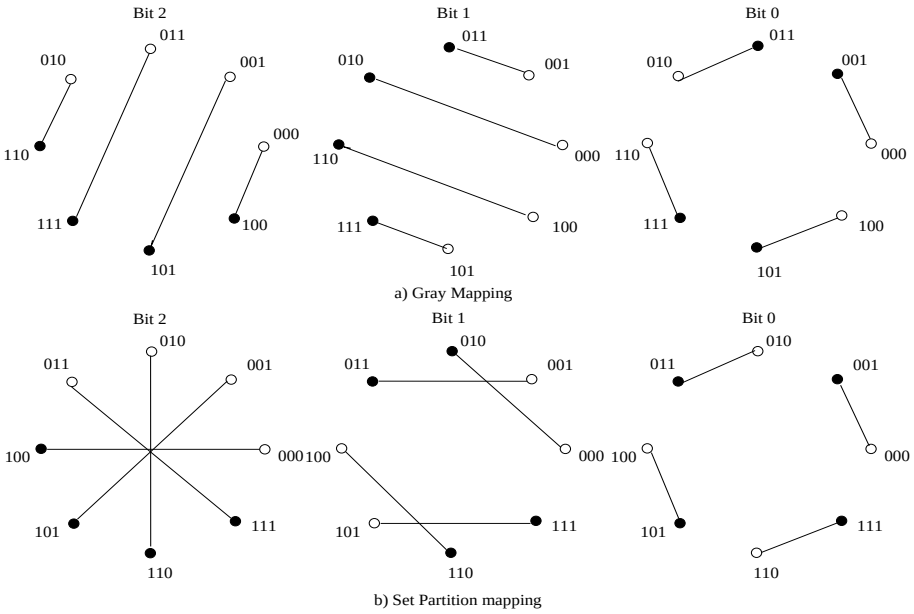


Figure A.2: The Gray and Set-partition mappings

Appendix B

A Parity Check Matrix (PCM) of G-LDPC codes portrayed in Figure B.1 is created from constituent Single Parity Check (SPC) codes having a size SPC(4,3). The PCM of GLDPC-codes in Figure B.1 has three submatrices. The first matrix H^1 is a block diagonal matrix having the matrix elements SPC(4,3) constituted along its main diagonal. The second submatrix H^2 is created by interleaving the first submatrix and the third submatrix H^3 is created by interleaving the second submatrix H^2. The parameters of the G-LDPC code are K, N, J, where $K = 3$, $N = 12$, $J = 3$. The code rate of the G-LDPC code equals to $K/N = 1/4$.

An example of structure of G-LDPC codes is shown in Fig. B.1 as follows.

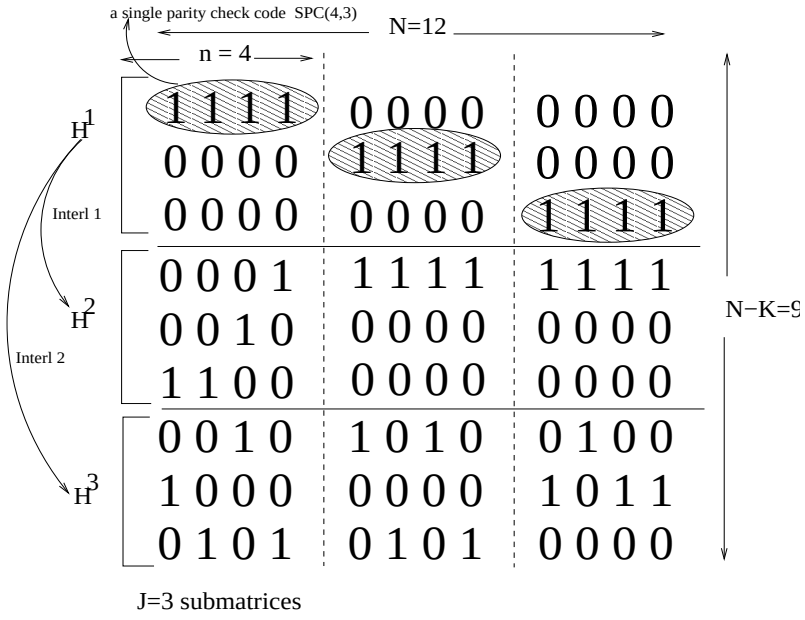


Figure B.1: A parity check matrix of G-LDPC codes

List of Symbols

General notation

- The superscript * is used to indicate complex conjugation. Therefore, a^* represents the complex conjugate of the variable a.
- The superscript ^{T} is used to indicate matrix transpose operation and the superscript $^{-1}$ is used to indicate matrix iverted operation. Therefore, $\mathbf{H}^T$ represents the transpose of the matrix $\mathbf{H}$.
- The notation $*$ denotes the convolutional process. Therefore, $a * b$ represents the convolution between variables a and b.
- The notation $\hat{x}$ represents the estimate of x.

Special symbols

A: The Non-singular matrix existing in a generator Matrix.

B: The component part of generator Matrix.

d: The degree of one packet.

$\rho(d)$: The Ideal Soliton degree distribution of degree d

$\tau(d)$: A part degree distribution of degree (d) of the robust soliton degree distribution.

K: The number of input LT packets.

N: The number of output LT packets.

Z: The sum of all $\rho(d)$ and $\tau(d)$ in function of the robust soliton degree distribution.

D, Ω, Λ: The degree distribution functions.

S: The number of LT encoded packets having a degree-one.

δ: The probability at which the LT decoder fails to recover all source packets from a certain number of output encoded packets just after encoding process.

c: The positive factor is used in calculating the number of LT encoded packets having a degree-one.

N_0: Noise energy.

E_b: Bit energy.

E_b/N_0: Ratio of bit energy to noise power spectral density.

P_e: Erasure Probability.

$\nu(d)$: The supplement function of the improved robust soliton degree distribution.

C: The code word.

I_E: Mutual Information of output Extrinsic LLRs at the output.

I_A: Mutual Information of output a priori LLRs at the input.

I_n: The integer number nth.

rot: The rotation function.

mod: The modulo function.

Det: indicates the detector.

apr: A priori.

ex: Extrinsic.

apt: A posteriori.

K': The number of users in the SDMA system.

M': The number of transmit antennas.

N': The number of receive antennas.

L_s: The system load.

tanh: The hyperbolic tangent.

d_c: The degree of check nodes.

d_m: The degree of message nodes.

γ: A positive integer factor used in the truncated degree distribution for designing the generator matrix of systematic Luby transform codes.

Q: Log-likelihood ratio message passed from message nodes to parity nodes.

R: Log-likelihood ratio message passed from parity nodes to message nodes.

ν: An extra probability is defined in IRSDD.

G,**G'**: Generator matrices.

$\mathbf{G}_m$: A component generator matrix.

H,**H'**: Parity check matrices.

$\mathbf{H}_m$: A component parity check matrix.

B,**A**,**C**, **R**,**S**,**0**: Matrices.

ξ: A set of coding graphs.

ξ_m: A component of ξ.

ϵ: An error percentage of a LT packet.

Glossary

16QAM	16-level Quadrature Amplitude Modulation
3G	Third Generation
AAODD	All-AT-One Degree Distribution
ACK	ACKnowledgement
ARQ	Automatic Request, Automatic request for retransmission of corrupted data
AWGN	Additive White Gaussian Noise
BCH	Bose-Chaudhuri-Hocquenghem. A class of forward error correcting codes (FEC)
BEC	Binary Erasure Channel
BER	Bit error ratio, the number of the bits received incorrectly
BICM-ID	Bit-Interleaved Coded Modulation with Iterative decoding
BP	Belief Propagation
BPSK	Binary Phase Shift Keying
BS	A common abbreviation for Base Station
CIR	Channel Impulse Response
CRC	Cyclic Redundancy Check
CRIG	Conditional Random Integer Generator

DVB-H	Digital TeleVision Broadcasting-Handheld
DVB-S	Digital TeleVision Broadcasting-Satellite
DVB-T	Digital TeleVision Broadcasting-Terestrial
EXIT	EXtrinsic Information Transfer
FEC	Forward Error Correction
G-LDPC	Generalized Low Density Parity Check
GM	Generator Matrix
H-ARQ	Hybrid Automatic Repeat reQuest
IP	Internet Protocol
IPTV	Internet Protocol TeleVision
IRSDD	Improved Robust Soliton Degree Distribution
LCRIG	Linear Congruential Random Integer Generator
LDGM	Low Density Generator Matrix
LDPC	Low Density Parity Check
LLR	Log Likelihood Ratio
LT	Luby Transform
LT-SDMA	Luby Transform coded Spatial Division Multiple Access
MAC	Medium Access Control
MAP	Maximum A Posteriori
MIMO	Multi-Input Multi-Output
ML	Maximum Likelihood
MMSE	Minimum Mean Square Error
MS	A common abbreviation for Mobile Station

MUI	Multi-User Interference
NACK	Negative ACKnowledgement
OHRSA	Optimized Hierarchy Reduced Search Algorithm
PCM	Parity Check Matrix
PDD	Poisson Degree Distribution
PER	Packet Error Ratio
PRI	Packet Reliability Information
QAM	Quadrature Amplitude Modulation
ReqN	Request Number
RIG	Random Integer Generator
RN	Receive Number
RS	Reed Solomon Codes
RSC	Recursive Systematic Convolutional
RSDD	Robust Soliton Degree Distribution
SBRIG	Swapping Bit Random Integer Generator
SD	Soft Decision
SDMA	Spartial Devision Multiple Access
SIC	Successive Interference Cancellation
SLT	Systematic Luby Transform
SN	Sequence Number
SR	Shift Register
SVD	Singular Value Decomposition
TDD	Truncated Degree Distribution

TDVD	Time DiVision Duplex
TPDD	Truncated Poisson Degree Distribution
V-BLAST	Vertical Bell Labs Layered Space-Time
WIC	Wireless Internet Channel
XOR	Exclusive OR
ZF	Zero Forcing

Bibliography

[1] M. G. Luby, M. Mitzenmacher, M. A. Schokrollahi and D. A. Spielman, "Efficient Erasure Correcting Codes," *IEEE Transactions on Information Theory*, vol. 47, pp. 569–584, Feb 2001.

[2] P. Elias, "Coding for Noisy Channels," *IRE International Convention Record*, vol. Part IV, pp. 37–46, 1955.

[3] I. S. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields," *SIAM Journal of Applied Math*, vol. 8, pp. 300–304, 1960.

[4] A. Hocquenghem, "Codes Correcteurs Derreurs," *Chiffres*, vol. 2, pp. 147–156, September 1959.

[5] R. C. Bose and C. R. Ray-Chaudhuri, "On a Class of Error correcting Binary Group Codes," *Information and Control*, vol. 3, pp. 68–79, 1960.

[6] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes," *Proceedings of the International Conference on Communications*, pp. 1064–1070s, May 1993.

[7] R. Gallager, "Low Density Parity Check Codes," *IRE Transactions on Information Theory*, pp. 21–28, Jan 1962.

[8] Q. Luo and P. Sweeney, "Hybrid-ARQ Protocols Based on Multilevel Coded Modulation," *IEE Electronics Letters*, vol. 39, pp. 1063–1065, July 2003.

[9] R. Fantacci, "Queuing Analysis of the Selective Repeat Automatic Repeat Request Protocol Wireless Packet Networks," *IEEE Transactions, Vehicular Technology*, vol. 45, pp. 258–264, May 1996.

[10] M. Luby, "LT Codes," in *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science, Vancouver, BC, Canada*, pp. 271–282, November 2002.

[11] A. Shokrollahi, "Raptor Codes," *IEEE Transactions on Information Theory*, vol. 52, pp. 2551–2567, June 2006.

[12] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, Oct 1948.

[13] A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory*, vol. IT-13, pp. 260–269, April 1967.

[14] L. R. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal Decoding of Linear Codes for Minimising Symbol Error Rate," *IEEE Transactions on Information Theory*, vol. 20, pp. 284–287, March 1974.

[15] C. Berrou, A. Glavieux and P. Thitimajshirna, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes," *In ICC'93*, pp. 1064 – 1070, March 1993.

[16] "Turbo Codes & Related Topics," *Proceedings of the International Symposium, ISTC'97, Brest, France*, 3–5 September 1997.

[17] D. J. Costello, A. Banerjee, T. E. Fuja and P. C. Massey, "Some Reflections on the Design of Bandwidth Efficient Turbo Codes," in *Proceedings of 4th ITG Conference on Source and Channel Coding*, no. 170 in ITG Fachbericht, (Berlin), pp. 357–363, VDE–Verlag, 28–30 January 2002.

[18] "Third Generation Partnership (3GPP) ." Technical Specification Group Radio Access Network, 1999. Downloadable at ftp://ftp.3gpp.org/Specs/March.

[19] J. W. Byers, M. Luby and M. Mitzenmacher, "Accessing Multiple Mirror Sites in Parallel: Using Tornado Codes to Speed Up Downloads," *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 1, pp. 275–283, March 1999.

[20] R. Tee, T. D. Nguyen, L-L. Yang and L. Hanzo, "Serially Concatenated Luby Transform Coding and Bit-Interleaved Coded Modulation Using Iterative Decoding for the Wireless Internet," *Proceedings of VTC 2006 Spring, Melbourne, CD ROM*, vol. 138, pp. 177–182, May 2006.

[21] T. D. Nguyen, F. C. Kuo, L-L. Yang and L. Hanzo, "Amalgamated Generalized Low Density Parity Check and Luby Transform Codes for the Wireless Internet," *On IEEE 65th VTC2007-Spring Vehicular Technology Conference, Dublin*, pp. 2440–2444, April 2007.

[22] R. Tee, T. D. Nguyen, L-L. Yang and L. Hanzo, "Luby Transform Coding Aided Bit-Interleaved Coded Modulation for the Wireless Internet," *in Proceedings of VTC Fall 2007, Baltimore*, pp. 2025–2029, September 2007.

[23] C-Y. Wei, T. D. Nguyen, N. Wu, J. Akhtman, L-L. Yang and L. Hanzo, "Luby Transform Coding Aided Iterative Detection for Downlink SDMA Systems," *In Proceedings of IEEE Workshop on Signal Processing Systems in Shanghai, China*, pp. 105–110, October 2007.

[24] N. Wu, T. D. Nguyen, Chun-Yi Wei, L. L. Yang and L. Hanzo, "Integrated Luby Transform Coding, Bit Interleaved Differential Space Time Coding and Sphere Packing Modulation for the Wireless Internet," *In Proceedings of IEEE 67th VTC2008-Spring Vehicular Technology Conference, Singapore*, pp. 344–348, May 2008.

[25] T. D. Nguyen, L-L Yang and L. Hanzo, "Systematic Luby Transform Codes and Their Soft Decoding," *Proceedings of IEEE Workshop on Signal Processing Systems in Shanghai*, pp. 67–72, October 2007.

[26] T. D. Nguyen, L. L. Yang, S. X. Ng and L. Hanzo, "An Optimal Degree Distribution Design and A Conditional Random Integer Generator for the Systematic Luby Transform Coded Wireless Internet," *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 243–248, April 2008.

[27] T. D. Nguyen, L. L. Yang, S. X. Ng and L. Hanzo, "Systematic Luby Transform Codes and Their Degree Distribution Designed for Transmission over Rayleigh Fading Contaminated Binary Erasure Channels," *Submitted to the IEEE Transactions on Vehicular Technology*, 2008.

[28] T. D. Nguyen, M. El-Hajjar, L. L. Yang and L. Hanzo, "A Systematic Luby Transform Coded V-BLAST System," *Proceedings of the 2008 IEEE International Conference on Communications, Beijing, China*, pp. 775–779, 19-23 May 2008.

[29] T. D. Nguyen, R. Tee, L-L. Yang and L. Hanzo, "Hybrid ARQ Aided Systematic Luby Transform Coded Modulation," *Submitted to Transactions Vehicular Technology, 2008, at http://eprints.ecs.soton.ac.uk/view/groups/group-comms/.*

[30] L. Hanzo, T. H. Liew and B. L. Yeap, *Turbo Coding, Turbo Equalisation and Space-Time Coding for Transmission over Fading Channels*. Wiley & IEEE Press, 2002.

[31] S. ten Brink, "Convergence Behavior of Iteratively Decoded Parallel Concatenated Codes," *IEEE Transactions on Communications*, vol. 49, pp. 1727–1737, Oct 2001.

[32] M. C. Davey, "Error-correction using low density parity check codes," *Ph.D thesis, University of Cambridge, UK, at http://www.inference.phy.cam.ac.uk/mcdavey*, 1999.

[33] J. Garcia-Frias and W. Zhong, "Approaching Shannon Performance by Iterative Decoding of Linear Codes with Low-Density Generator Matrix," *Communications Letters, IEEE*, vol. 7, pp. 266–268, June 2003.

[34] H. Lou and J. Garcia-Frias, "Low-Density Generator Matrix Codes for Indoor and Markov Channels," *IEEE Transactions on Wireless Communications*, vol. 6, pp. 1436–1445, April 2007.

[35] K. Nybom, J. Kempe and J. Bjorkqvist, "Improving Object Delivery Using Application Layer Tornado Codes in DVB-H Systems," *Personal, Indoor and Mobile Radio Communications, 2006 IEEE 17th International Symposium*, pp. 1–5, September 2006.

[36] D. J. C. Mackay, "Fountain Codes," in *Cavendish Laboratory, University of Cambridge, at http://www.inference.phy.cam.ac.uk/mackay*, October 2004.

[37] R. J. McEliece, " A Public-Key Cryptosystem Based on Algebraic Coding Theory," *Tech. report, Jet Propulsion Lab. DSN Progress Report*, 1978.

[38] J. W. Byers, M. Luby and M. Mitzenmacher, "A Digital Fountain Approach to Asynchronous Reliable Multicast," *IEEE Journal on Selected Areas in Communications*, vol. 20, pp. 1528–1540, October 2002.

[39] D. J. C. MacKay, *Information Theory, Interference and Learning Algorithms*. Cambridge University Press, 2003. http://www.inference.phy.cam.ac.uk/mackay/itila/.

[40] A. Viswanathan, K. J. Zhang, "Stopping Set Distribution of LDPC Code Ensembles," *IEEE Transactions on Information Theory*, vol. 51, pp. 929–953, March 2005.

[41] R. Karp, M. Luby and A. Shokrollahi, "Finite Length Analysis of LT Codes," *ISIT 2004*, p. 37, June 27-July 2 2004.

[42] M. G. Luby, M. Shokrollahi, M. Watson and T. Stockhammer, "Raptor Forward Error Correction Scheme," *Reliable Multicast Transport Internet-Draft, at http://www.ietf.org/mail-archive/web/ietf-announce/current/msg04204.html*, June 2005.

[43] J. Buyers, M. Luby, M. Mitzenmacher and A. Rege, "A Digital Fountain Approach to Reliable Distribution of Bulk Data," *In Proceedings of ACM SIGCOMM'98*, pp. 56–67, September 1998.

[44] E. Zehavi, "8-PSK Trellis Codes For a Rayleigh Fading Channel," *IEEE Transactions on Communications*, vol. 40, pp. 873–883, May 1992.

[45] X. Li and J. A. Ritcey, "Bit-Interleaved Coded Modulation with Iterative Decoding," *IEEE Communications Letters*, vol. 1, pp. 169–171, November 1997.

[46] L. Hanzo, L-L. Yang, E-L. Kuan and K. Yen, *Single- and Multi-Carrier DS-CDMA*. New York, USA: John Wiley IEEE Press, 2003.

[47] M. Lentmaier and K. S. Zigangirov, "On Generalized Low-Density Parity-Check Codes Based on Hamming Component Codes," *IEEE Communications Letters*, pp. 248–250, Aug 1999.

[48] F. C. Kuo, L. Hanzo., "Symbol-Flipping Based Decoding of Generalized Low-Density Parity-Check Codes over GF(q)," *Proceedings and CDROM of IEEE WCNC 2006, 3-6 April,2006, Las Vegas*, vol. 138, pp. 1207–1211, May 2006.

[49] S. Hirst and B. Honary, "Decoding of Generalised Low-Density Parity-Check Codes Using Weighted Bit-Flip Voting," *IEE Proceedings Communications*, vol. 149, pp. 1–5, Feb 2002.

[50] S. Lin, D. J. Costello and M. J. Miller, "Automatic-Repeat-Request Error-Control Schemes," *Communications Magazine, IEEE*, vol. 22, pp. 5–17, December 1984.

[51] R. Fantacci, "Queuing Analysis of the Selective Repeat Automatic Repeat Request Protocol Wireless Packet Networks," *On Vehicular Technology, IEEE Transactions*, vol. 45, pp. 258–264, May 1996.

[52] M. Matsumoto and T. Nishimura, "Mersenne Twister: A623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator," *ACM Trans, Modeling and Computer Simulation*, vol. 8, no. 1, pp. 31–42, 1998.

[53] T. Clevorn, S. Godtmann and P. Vary, "BER Prediction Using EXIT Charts For BICM With Iterative Decoding," *IEEE Communications Letters*, vol. 10, pp. 49–51, January 2006.

[54] A. Fog, *Chaotic Random Number Generators with Random Cycle Lengths*. December 2004. http://www.agner.org/random/theory.

[55] C. Kuo, T. W. Cadman and J. R. Arsenault, "Sequential Random Integer Generator," *Computer Physics Communications*, vol. 12, pp. 163–171, May 1996.

[56] L. Hanzo, M. Münster, B. J. Choi and T. Keller, *OFDM and MC-CDMA for Broadband Multi-User Communications, WLANs and Broadcasting*. Chichester, UK: Wiley, 2003.

[57] P. Elias, "Coding for Two Noisy Channels," in *Information Theory Proceedings of the 3rd London Symposium*, pp. 61–76, 1955.

[58] D. Divsalar, S. Dolinar and F. Pollara, "Serial Turbo Trellis Coded Modulation with Rate-1 Inner Code," *IEEE Global Telecommunications Conference, 2000. GLOBECOM '00.*, vol. 2, pp. 777 – 782, 2000.

[59] L-U Choi and R. D. Murch, "A Transmit Preprocessing Technique for Multiuser MIMO Systems Using A Decomposition Approach," *IEEE Transactions on Wireless Communications*, vol. 3, no. 1, pp. 20 – 24, 2004.

[60] J. Akhtman, C-Y. Wei and L. Hanzo, "Reduced-Complexity Maximum-Likelihood Detection in Downlink SDMA Systems," *Vehicular Technology Conference, Sep 25-28, Montreal, Canada, VTC2006-Fall. 2006 IEEE 64th, eprint at http://eprints.ecs.soton.ac.uk/view/groups/group-comms/2006.html*, 2006.

[61] L. Hanzo and M. Münster and B-J. Choi and T. Keller, *OFDM and MC-CDMA for Broadband Multi-User Communications, WLANs and Broadcasting*. John Wiley and IEEE press, 2003.

[62] L. Hanzo and T. Keller, *OFDM and MC-CDMA: A Primer*. John Wiley and IEEE press, 2006.

[63] J. E. Gentle, *Numerical Linear Algebra for Applications in Statistics*. Berlin: Springer-Verlag, 1998.

[64] R. Karp, M. Luby and A. Shokrollahi, "Verification Decoding of Raptor Codes," *Information Theory, 2005, ISIT 2005, Proceedings, International Symposium*, pp. 1310 – 1314, Sebt 2005.

[65] T. Richardson, R. Urbanke, "The Capacity of Low-Density Parity Check Codes Under Message-Passing Decoding," *IEEE Transaction on Information Theory*, vol. 47, pp. 599–618, Feb 2001.

[66] F. Guo, "Low Density Parity Check Coding," *Faculty of Engineering and Applied Science, School of Electronics and Computer Science, Southampton University*, January 2005.

[67] R. Palanki and J. S. Yedidia, "Rateless Codes on Noisy Channels," *ISIT 2004. Proceedings*, p. 37, June 2004.

[68] S. Y. Chung, *On the Construction of Some Capacity-Approaching Coding Schemes*. 2000. at http://portal.acm.org/citation.cfm?id=934066.

[69] B. M. Kurkoski, P.H. Siegel and J. K. Wolf, "Joint Message-Passing Decoding of LDPC Codes and Partial-Response Channels," *On IEEE Transactions on Information Theory*, vol. 48, pp. 1410–1422, June 2002.

[70] A. Ashikhmin, G. Kramer and S. ten Brink, "Extrinsic Information Transfer Functions: Model and Erasure Channel Properties," *IEEE Transactions on Information Theory*, vol. 50, pp. 2657–2673, Nov 2004.

[71] E. Sharon, "Analysis of Belief-Propagation Decoding of LDPC Codes over the bi-AWGN Channel using Improved Gaussian Approximation Based on the Mutual Information Measure," *Electrical and Electronics Engineers in Israel. The 22nd Convention of*, pp. 262–264, Dec 2002.

[72] E. Sharon, A. Ashikhmin and S. Litsyn, "Analysis of Low-Density Parity-Check Codes Based on EXIT Functions," *IEEE Transactions on Communications*, vol. 54, pp. 1407–1414, Aug 2006.

[73] V. Tarokh, H. Jafarkhani and A. R. Calderbank, "Space-time Block Codes from Orthogonal Designs," *IEEE Transactions on Information Theory*, vol. 45, pp. 1456–1467, May 1999.

[74] P. W. Wolniansky, G. J. Foschini, G. D. Golden and R. A. Valenzuela, "V-BLAST: An Architecture for Realizing Very High Data Rates over The Rich-Scattering Wireless Channel," *International Symposium on Signals, Systems, and Electronics*, pp. 295–300, 1998.

[75] L. Hanzo, S. X. Ng, T. Keller and W. Webb, *Quadrature Amplitude Modulation: From Basics to Adaptive Trellis-Coded, Turbo Equalised and Space-Time Coded OFDM, CDMA and MC-CDMA Systems, 2nd Edition*. Chichester, England: John Wiley and Sons Ltd and IEEE Press, 2004.

[76] G. Foschini and M. Gans, "On Limits of Wireless Communication in a Fading Environment when using Multiple Antennas," *Wireless Personal Comunications*, vol. 10, pp. 311–335, March 1998.

[77] E. Telatar, "Capacity of Multi-Antenna Gaussian Channels," *European Transactions on Telecommunications*, vol. 10, pp. 585–595, Nov./Dec. 1999.

[78] N. H. Tran and H. H. Nguyen, "Signal Mappings of 8-ary Constellations for Bit Interleaved Coded Modulation with Iterative Decoding," *IEEE Transactions on Broadcasting*, vol. 52, pp. 92–99, March 2006.

[79] F. Shreckenbach, P. Henkel, N. Görtz and G. Bauch, "Analysis and Design of Mappings for Iterative Decoding of BICM," *XI National Symposium of Radio Sciences, Poznan*, pp. 82–87, April 2005.

[80] G. D. Golden, G. J. Foschini, R. A. Valenzuela and P. W. Wolniansky, "Detection Algorithms and Initial Laboratory Results using V-BLAST Space-Time Communication Architecture," *IEE Electronics Letters*, vol. 35, pp. 14–16, January 1999.

[81] G. Caire, G. Taricco and E. Biglieri, "Bit-Interleaved Coded Modulation," *IEEE Transactions on Information theory*, vol. 44, pp. 927–946, May 1998.

[82] M. Sabbaghian, D. Falconer, "Comparison between Convolutional and LDPC Code-based Turbo Frequency Domain Equalization," *IEEE International Conference on Communications*, vol. 12, pp. 5432–5437, June 2006.

[83] P. S. Sindhu, "Retransmission Error Control with Memory," *IEEE Transactions on Comunications*, vol. COM-25, pp. 473–479, May 1977.

[84] G. Benelli, "Throughput Optimisation of a Stop-And-Wait ARQ Protocol," *Electronics Letters*, vol. 24, pp. 735–736, June 1988.

[85] C. Pimentel and R. L. Siqueira, "Analysis of the Go-Back-N Protocol on Finite State Markov Fading Channels," *Vehicular Technology Conference, 2006. VTC 2006-Spring. IEEE 63rd*, vol. 4, pp. 2042–2046, May 2006.

[86] M. Miller and S. Lin, "The Analysis of Some Selective-Repeat ARQ Schemes with Finite Receiver Buffer," *IEEE Transactions on Communications*, vol. 29, pp. 1307–1315, September 1981.

[87] D. Towsley, "A Statistical Analysis of ARQ Protocols Operating in a Nonindependent Error Environment," *IEEE Transactions on Communications*, vol. 29, pp. 971–981, July 1981.

[88] S. Kallel, "Analysis of a Type II Hybrid ARQ Scheme with Code Combining," *IEEE Transactions on Communications*, vol. 38, pp. 1133–1137, August 1990.

[89] A. Shiozaki, K. Okuno, K. Suzuki and T. Segawa, "A Hybrid ARQ Scheme with Adaptive forward Error Correction for Satellite Communications," *IEEE Transactions on Communications*, vol. 39, pp. 482–484, April 1991.

[90] F. Babich, "Performance of Hybrid ARQ Schemes for the Fading Channel," *IEEE Transactions on Communications*, vol. 50, pp. 1882–1885, December 2002.

[91] R. H. Deng, "Hybrid ARQ Schemes Employing Coded Modulation and Sequence Combining," *IEEE Transactions on Communications*, vol. 42, pp. 2239–2245, June 1994.

[92] L. F. Wei, "Coded Modulation with Unequal Error Protection," *IEEE Transactions on Communications*, vol. 41, pp. 1439–1450, October 1993.

[93] F. Guo, S. X. Ng and L. Hanzo, "LDPC assisted Block Coded Modulation for Transmission over Rayleigh Fading Channels," in *IEEE Vehicular Technology Conference*, vol. 3, (Florida, USA), pp. 1867–1871, April Spring 2003.

[94] A. Avudainayagam, J. M. Shea and A. Roongta, "Improving the Efficiency of Reliability-based Hybrid-ARQ with Convolutional Codes," *Military Communications Conference - MILCOM. IEEE, Atlantic City*, vol. 1, pp. 448–454, October 2005.

[95] J. Hamorsky and L. Hanzo, "Performance of the Turbo Hybrid Automatic Repeat Request System Type II," *ITW 1999, Metsovo, Greece, 27 June - 1 July*, p. 51, 1999.

[96] W. Chen, Y. B. Lin and A. C. Pang, "An IPv4-IPv6 Translation Mechanism for SIP Overlay Network in UMTS all-IP Environment," *IEEE Journal of Selected Areas in Communications*, vol. 23, pp. 2152–2160, November 2005.

[97] A. Chindapol, J. A. Ricey, "Design, Analysis, and Performance Evaluation for BICM-ID with Square QAM Constellations in Rayleigh Fading Channels," *IEEE Journal on Selected Areas in Communications*, vol. 19, pp. 944–957, May 2001.

[98] G. Lechner, J. Sayir and I. Land, "Optimization of LDPC Codes for Receiver Frontends," *International Symposium on Information Theory*, pp. 2388–2392, July 2006.

[99] S. Y. L. Goff, "Signal Constellations for Bit-Interleaved Coded Modulation," *IEEE Transactions on Information Theory*, vol. 49, pp. 307–313, January 2003.

[100] C. Berrou and A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo-Codes," *IEEE Transactions on Communications*, vol. 44, pp. 1261–1271, October 1996.

Index

Author Index