EXIT CHART BASED ANALYSIS AND DESIGN OF RATELESS CODES FOR THE ERASURE AND GAUSSIAN CHANNELS

A Thesis

by

SABARESAN MOTHI VENKATESAN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2007

Major Subject: Electrical Engineering

EXIT CHART BASED ANALYSIS AND DESIGN OF RATELESS CODES FOR THE ERASURE AND GAUSSIAN CHANNELS

A Thesis

by

SABARESAN MOTHI VENKATESAN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Krishna R. Narayanan
Committee Members,	Henry D. Pfister
	Christi K. Madsen
	Prabir Daripa
Head of Department,	Costas N. Georghiades

August 2007

Major Subject: Electrical Engineering

ABSTRACT

EXIT Chart Based Analysis and Design of Rateless Codes for the Erasure and Gaussian Channels. (August 2007)

Sabaresan Mothi Venkatesan, B.E., Anna University,

PSG College of Technology, Coimbatore

Chair of Advisory Committee: Dr. Krishna R. Narayanan

Luby Transform Codes were the first class of universal erasure codes introduced to fully realize the concept of scalable and fault-tolerant distribution of data over computer networks, also called Digital Fountain. Later Raptor codes, a generalization of the LT codes were introduced to trade off complexity with performance. In this work, we show that an even broader class of codes exists that are near optimal for the erasure channel and that the Raptor codes form a special case. More precisely, Raptor-like codes can be designed based on an iterative (joint) decoding schedule wherein information is transferred between the LT decoder and an outer decoder in an iterative manner. The design of these codes can be formulated as a LP problem using EXIT Charts and density evolution. In our work, we show the existence of codes, other than the Raptor codes, that perform as good as the existing ones.

We extend this framework of joint decoding of the component codes to the additive white Gaussian noise channels and introduce the design of Rateless codes for these channels. Under this setting, for asymptotic lengths, it is possible to design codes that work for a class of channels defined by the signal-to-noise ratio. In our work, we show that good profiles can be designed using density evolution and Gaussian approximation. EXIT charts prove to be an intuitive tool and aid in formulating the code design problem as a LP problem. EXIT charts are not exact because of the inherent approximations. Therefore, we use density evolution to analyze the performance of these codes. In the Gaussian case, we show that for asymptotic lengths, a range of designs of Rateless codes exists to choose from based on the required complexity and the overhead.

Moreover, under this framework, we can design incrementally redundant schemes for already existing outer codes to make the communication system more robust to channel noise variations.

DEDICATION

To God, Mom, Dad & Harish

ACKNOWLEDGMENTS

I would like to express my heartfelt gratitude to The Almighty, for providing me the strength and endurance that kept me going during tough times and giving me the right ideas and insights whenever I needed them most. I am ever grateful to Him, for it was His Presence and Intelligence that guided me during the entire course of this work.

I am very grateful to my advisor, Dr. Narayanan, for giving me this opportunity to work under him. Without his constant guidance, suggestions and encouragement, this work would not have been possible. I would like to specifically thank him for the formal and informal discussions where he spent his precious time, and many weekends, sharing his invaluable ideas and experiences with me. He always made sure that I had my concepts clear and that I was heading in the right direction. In fact, this topic sprung from the discussions that we had after his Advanced Channel Coding class. I always enjoyed and loved working with him. Besides his research advising, I express my heartfelt gratitude to him, for his abundant help in the last few weeks before my final exam to meet multiple deadlines. He has been on my side through tough times and I believe I was fortunate to have him as my advisor.

I would also like to thank Dr. Pfister, for the opinions and ideas that he readily shared with me whenever I approached him. The discussions that I had with him besides my advisor were very fruitful and insightful. Also, his courses were enlightening and the knowledge I gained through his courses, especially, Channel Coding created the foundation upon which I built my thesis.

I would like to thank my committee members, Dr. Madsen and Dr. Daripa, for readily accepting to be on my committee when I first approached them. They were of great support to me during the last few days of my final exam preparation. Without their help and cooperation, this thesis could not have been completed on time. I would also express my gratitude to Ms. Tammy Carda, Ms. Jeanie Marshall, Ms. Paula Evans, Ms. Andrea Bragdon, Mr. John Neal, Ms. Christina Robertson, Ms. Shirley and other people in the Office of Graduate Studies, ECE Graduate Office and Thesis office for helping me out with all the required paper work. I would like to acknowledge our Departmental Graduate Advisor, Dr. S. Miller for approving my paper work whenever required.

I would like to acknowledge Karthik Nagasubramanian, for patiently working with me during all the combined course projects and sharing his experiences to solve issues I encountered in projects as well as my research. Further Karthik, Nirmal and Makesh Pravin helped proof read many versions of this thesis. I also extend my gratitude to Anantharaman, Jing Jiang, Janath Peris, Kapil Bhattad, Wei-Yu, Parimal Parag, Lingjia Lu, other friends and professors for their constant support.

Most importantly, I would like to thank my mother, father and brother for their love and for psychologically, financially and spiritually supporting me during tough times. My uncles, aunts and cousins have also extended their support to me in many ways during my stay in the United States.

ABBREVIATIONS

ARQ	Automatic Repeat Request
AWGN	Additive White Gaussian Noise
BIAWGN	Binary Input Additive White Gaussian Noise
BIMSC	Binary Input Memoryless Symmetric Channel
DDE	Discrete Density Evolution
DE	Density Evolution
EXIT	EXtrinsic Information Transfer
FEC	Forward Error Correcting Code
GA	Gaussian Approximation
LBC	Linear Block Code
LDPC	Low Density Parity Check Code
LLR	Log-Likelihood Ratio
LT	Luby Transform
MSE	Mean Squared Error
pdf	Probability density function
pmf	Probability mass function
RV	Random Variable
SNR	Signal-to-noise Ratio

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

1 INTRODUCTION

1.1 Overview

In computer networks, especially the internet, data is transmitted in the form of packets. These packets are routed from the transmitter to the receiver via many intermediate nodes. Based on the bandwidth, traffic and throughput of the channel and the router, packets are dropped at intermediate nodes. Many protocols have been developed to ensure reliable unicast (point-to-point) and multicast (transmitting same information to multiple users) communication even in the presence of such packet losses. In the point-to-point setting, there exist protocols based on Automatic Repeat Request (ARQ) schemes that have variable overheads and delays depending on the application. However, in the multicast scenario ARQ schemes can be expensive in terms of resources, because, if each user had to return an acknowledgment for the received packets, then for a large file (split into many packets), the overhead could be large. This gets even worse when at different receivers different sets of packets are lost because they have to request retransmission and the transmitter might end up transmitting many copies of the same packet.

Digital fountain codes were introduced as an efficient coding scheme for multicasting the same data over the network to multiple users experiencing different packet loss rates. The idea is to treat transmission of packets as transmission through an erasure channel and design appropriate coding schemes for the erasure channel. These are forward error correcting schemes i.e., they do not require an acknowledgement for each packet, as against ARQ schemes where the receiver has to request retransmission when packets are dropped or acknowledge when packets are received. Further, since the packet erasure rate to different receivers can vary widely

---

This thesis follows the style of the *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.*

and may not be known apriori, these codes have to be rateless i.e., they must produce an endless stream of coded packets from a finite number of data fragments and also achieve optimal performance for any erasure rate. Hence, design of such near optimal, rateless and low complexity coding schemes are of great importance to applications involving transmission of large files such as digital video broadcasts, etc., which cannot tolerate packet losses.

The concept of rateless and universal codes for the BIAWGN Channel has widespread applications in physical layer communications, especially, in the broadcast and multicast scenarios. These FEC rateless codes play a significant role when the feedback from the receiver is absent or expensive e.g. satellite communication.

1.2 Prior Work

Reed Solomon Codes are optimal for the packet erasure channel because they are maximum distance separable codes and hence, can recover the k original data packets from any subset of k packets of n coded packets. However, these codes have decoding and encoding complexities that are polynomial in n, thereby making them impractical for large block lengths. Further, they cannot be made rateless.

Recent developments in Low Density Parity Check (LDPC) Codes have shown that LDPC codes can be very efficient in terms of complexity and memory requirements, and there is a good amount of literature dealing with these code constructions [1], [2], [3], [4] and hence, are a viable choice for the multicast application. These codes are capacity achieving especially for the erasure channel and also have linear time encoding and decoding requirements. The only disadvantage of these codes is that, they are not rateless. In other words, it is not possible to design low overhead LDPC codes that produce an infinite stream of coded bits on the fly.

Luby Transform (LT) codes [5], [6] were the first full realization of the Digital Fountain concept. LT codes are rateless codes based on sparse graphs and require vanishing overheads with block length. They are universal for the erasure channel i.e.,k

data packets can be decoded on any erasure channel once any $k(1+\epsilon)$ of the encoded packets have been received for an arbitrarily small ϵ. Hence, they are well suited for the multicast application. These codes are designed by borrowing ideas from Random codes and LDPC codes and incorporate the best of both constructions. The only drawback is that the encoding and decoding complexities are $O(k \ln k)$ which is somewhat large for large k.

Raptor codes were later introduced in [7] to overcome this drawback of LT codes at the expense of a slight increase in the overhead. These codes are attractive because they have linear time encoding and decoding complexities like LDPC Codes. As a result of simplicity and low complexity requirements of these codes, they have already found applications in many current practical systems including 3GPP, [8]. A more detailed description of Raptor and LT codes will follow in section 2.

The universal code design problem has not been solved in full generality for Binary Input Memoryless Symmetric Channels (BIMSCs), other than the erasure channel. The success of Raptor codes for the erasure channel (belonging to the class of BIMSCs) suggests that similar results can be extended to other BIMSCs like the BIAWGN Channel too. Etesami and Shokrollahi addressed this issue of Raptor code design for the BIAWGN Channel and other BIMSCs in [9]. Their approach involved designing a near optimal (minimal reception overhead) Raptor code for a particular channel given by its channel parameter, e.g. Signal-to-Noise Ratio, SNR_{design} in BIAWGN Channels or crossover probability, p_{design} in BSC Channels. However, this procedure resulted in an overhead significantly larger than the design overhead for the receivers experiencing a different channel i.e., for $SNR \neq SNR_{design}$ in the case of the BIAWGN channel and for $p \neq p_{design}$ in the BSC case.

1.3 Contribution

We introduce the concept of iteratively decoding the inner code and the outer code and show that a larger class of codes exists for the BIMSCs and that we can indeed achieve the trade-offs even under this setting. In this framework, we are interested in the information transfer between the two component decoders. We analyze the existing codes under this more general decoding structure. Our approach involves the application of existing tools like EXIT Charts [10] and Density Evolution [2], to design codes that subtly trade overhead, space and complexity under the iterative decoder setting. At this point, it is worth mentioning; that codes obtained in [5], [6], [7], [9] can be viewed as special classes of our design, where each one is obtained by imposing a certain restriction on the decoding structure.

In [7], [9] the authors mention about these trade-offs, however, they have not proposed any technique which can efficiently address the issue of improving performance when space is not an issue. In fact, we show that there exist designs that require more space, but are less complex and perform better than the codes proposed in [7], [9].

The concept of incrementally adding redundancy using a randomly generated inner code given an outer code for the erasure and BIAWGN Channels have widespread application especially because most existing communication systems have efficient implementations of some inner code e.g., LDPC codes. Our framework, addresses this problem of designing inner codes that match existing outer code profiles with minimal overhead and complexity.

The problem of universal code design for the BIAWGN Channel is not as simple as the erasure channel and hence, we need to broaden the search space to include all potential candidates to design efficient codes. We extend the framework developed in the erasure case to the BIAWGN Channel to help us obtain near optimal codes. We have made an attempt to design codes that are almost optimal (require low overheads) and less complex than the codes proposed in [9] for a wide range of SNRs. Our simulation results coincide with the asymptotic analysis for designs based on the above techniques for both the erasure and the BIAWGN Channels.

2 BACKGROUND

2.1 Terminology

2.1.1 Rateless Codes

A code is said to be rateless if the number of encoding symbols that can be generated from the information bits is potentially limitless. Furthermore, encoding symbols can be generated on the fly, for as few or as many as needed. Also, for large lengths, with high probability the decoder can recover the original information bits from any set of the encoded symbols of size slightly more than the actual number of bits required as per the capacity of the channel. In other words, if the length of the information sequence is k and the capacity of channel is C, then the number of encoded symbols required at the receiver is $(1+\epsilon)k/C$.

2.1.2 Digital Fountain

A digital fountain is a concept that was originally proposed for the packet based erasure channel, where the servers can transmit an endless stream of distinct encoded packets to multiple users and the receivers can decode the data once they receive any subset of the transmitted packets equal in length to the total source data. These packets resemble droplets of a fountain and hence, the term digital fountain is used to refer to these rateless codes over the packet erasure channel. In an idealized digital fountain, the receivers can reconstruct an exact copy from the data burst independent of the losses incurred by the channel and independent of the time at which the receiver enters the session.

2.1.3 Universal Codes

A code is said to be universal over a class of channels defined by the channel parameter and capacities $C \in \mathcal{C}$, if the code requires arbitrarily low reception overhead ϵ beyond the capacity to decode the actual information bits for all the channels $\in \mathcal{C}$. In other words, any receiver that sees a channel with any capacity $C \in \mathcal{C}$, can recover the data from $(1+\epsilon)k/C \; \forall \; C \in \mathcal{C}$, where k is the number of information bits. For example, consider a set of BIAWGN channels with different SNRs and let $C_{BPSK}(snr)$ denote the capacity of the binary input AWGN channel with $SNR = snr$. Then, for any channel with $SNR = snr$, we should be able to decode k information bits from $(1+\epsilon)k/C_{BPSK}(snr)$.

2.2 Tanner Graph

Any linear block code can be described by a bipartite (tanner) graph as shown in Fig.1. The bipartite graph has two types of nodes – the check nodes and the bit nodes. Each check node can be thought of as a single parity code with the data bits corresponding to the bit nodes that are connected to it. Each check node therefore denotes a linear combination (XOR) of a few bit node messages. An edge in the graph between a bit node and a check node denotes that the bit contributes to the linear combination. All the codes that will be discussed hereinafter, LDPC, LT, Raptor, etc., can be viewed as a graph based code.

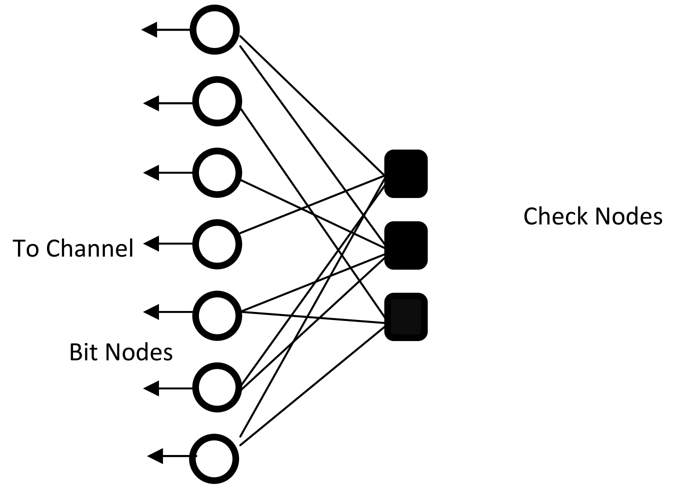

Fig.1 Bipartite graph of a linear block code

In general, the code is described by the degree profiles corresponding to bit and check nodes. The *bit edge* degree profile of a graph based code is given by,

$$\lambda(x) = \sum_i \lambda_i x^{i-1} \tag{2.1}$$

where, λ_i represents the fraction of edges that are connected to a bit node of degree i. The *bit node* degree profile is given by,

$$L(x) = \sum_i L_i x^i \tag{2.2}$$

where, L_i represents the fraction of bit nodes of degree i. Similarly, the *check edge* degree profile is given by,

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

where, ρ_i represents the fraction of edges that are connected to a check node of degree i. The *check node* degree profile is given by,

$$R(x) = \sum_i R_i x^i \tag{2.4}$$

where, R_i represents the fraction of check node of degree i. Also the relation between the edge degree profile, $e(x)$ and the node degree profile, $N(x)$ is given by

$$N(x) = \frac{\int_0^x e(z)dz}{\int_0^1 e(z)dz} \tag{2.5}$$

$$e(x) = \frac{N'(x)}{N'(1)} \tag{2.6}$$

The rate of the code in terms of its degree profiles is given by

$$r = 1 - \frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx} = 1 - \frac{R'(1)}{L'(1)} \tag{2.7}$$

2.3 Message Passing Decoder

The decoding of a LBC that can be represented by an underlying graph follows by passing reliability values from one node to the other. We will discuss this algorithm for the communications channels of interest.

2.3.1 General Message Passing Algorithm

For the classes of codes that we will consider in Sections 3 and 4, the messages that are passed along the edges are indicated in Fig.2.

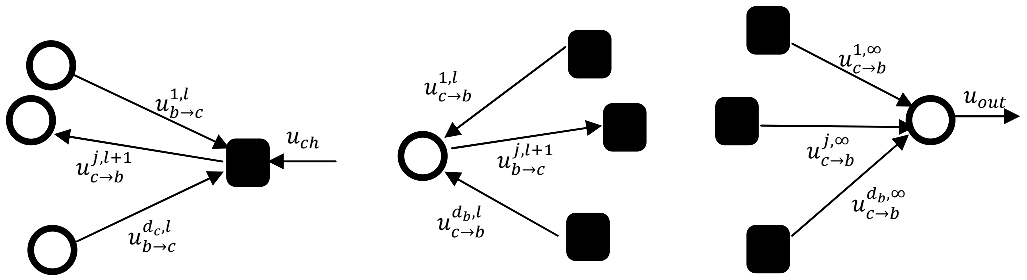

Fig.2 Bit node and check node MPA

The MPA is performed in 2 stages:

1. Messages sent along the edge from the bit nodes to the check nodes are

$$u_{b\to c}^{j,l+1} = g(u_{c\to b}^{1,l}, u_{c\to b}^{2,l}, \dots u_{c\to b}^{j-1,l}, u_{c\to b}^{j+1,l}, \dots u_{c\to b}^{d_b,l}, u_{ch}^{l}) \tag{2.8}$$

2. Messages sent along the edge from the check nodes to the bit nodes are

$$u_{c\to b}^{j,l+1} = h(u_{b\to c}^{1,l}, u_{b\to c}^{2,l}, \dots, u_{b\to c}^{j-1,l}, u_{b\to c}^{j+1,l}, \dots, u_{b\to c}^{d_c,l}) \tag{2.9}$$

where, u^l are messages passed in the l -th iteration. u_{ch}- channel input

h and g denote the operations performed at the check node and the bit node respectively, and are dependent on the communication channel.

After sufficiently large iterations, the final value of the bit are computed as

$$u_{out} = g(u_{c\to b}^{1,\infty}, u_{c\to b}^{2,\infty}, \dots, u_{c\to b}^{j-1,\infty}, u_{c\to b}^{j,\infty}, \dots, u_{c\to b}^{d_b,\infty}) \tag{2.10}$$

2.3.1.1 MPA for Erasure Channel

In the erasure channel, the message passing algorithm involves passing either an erasure, ? or the value i.e., 0 or 1. Let $\oplus$ denote the XOR operation. The message update rules are as follows:

1. $g(u_1, u_2, \dots) = \begin{cases} ?, if\ u_i = ?\ \forall i \\ u_j, if\ \exists\ j : u_j \neq ? \end{cases}$
2. $h(u_1, u_2, \dots) = \begin{cases} ?, if\ u_i = ?\ \text{for some } i \\ \oplus_i u_i\ if\ u_i \neq ?\ \forall i \end{cases}$

2.3.1.2 MPA for BIAWGN Channel

In the BIAWGN Channel, the message update rules are given as follows

1. $g(u_1, u_2, \dots) = \sum_i u_i$
2. $h(u_1, u_2, \dots) = 2\tanh^{-1}(\prod_i \tanh\left(\frac{u_i}{2}\right))$

2.4 Density Evolution

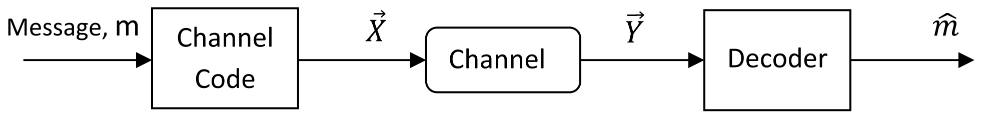

Fig.3 Communication system

Consider the communication systems shown in Fig.3. The message is encoded using a LBC that can be represented by a bipartite graph. The coded bits, $\vec{X}$ are transmitted across the channel and $\vec{Y}$ are received at the receiver. In BIMSCs, the log-likelihood ratio of each bit L_i is defined as,

$$L_i = \log \frac{\Pr[X_i = 0|\vec{Y}]}{\Pr[X_i = 1|\vec{Y}]} \tag{2.11}$$

Density Evolution is a technique used to analyze and design these bipartite graph based codes like LDPC, Raptor and LT codes. The channel induces a probability density function on these LLR values depending on the channel parameter. For the erasure channel, this parameter is the erasure probability, p_{eras} and for the BIAWGN channel it is the noise variance, σ^2. For our problem we are interested in characterizing the performance of the code by its overhead, ϵ. DE is used to obtain the minimum overhead, ϵ_{thresh} that ensures complete decoding of the message $(Pe(\epsilon) \to 0)$ i.e.,

$$\epsilon_{thresh} = \min_{P_e(\epsilon)\to 0} \epsilon \tag{2.12}$$

The density of the incoming channel $LLRs$ is symmetric if the communication channel is symmetric with respect to the input bits. Hence, for a LBC defined by its degree profiles, we track the densities along the edges between the nodes (bit and check) under the assumption that $the\ all\text{-}zero\ codeword$ was transmitted.

It is not possible to compute the closed form expression of the densities at the output of the check nodes because the operation at the check node is non-trivial. Hence, density evolution is typically performed numerically under different assumptions like Gaussian approximation, discretized density evolution [2], etc.,

2.4.1 Discretized Density Evolution

In discretized density evolution, we assume that the decoding is performed using an approximate iterative decoder. This decoder passes the quantized version of the messages along the edges. From [2] it is known that discretized density evolution gives an exact characterization of the quantized decoder.

In the BIAWGN channel, the density update equations corresponding to the messages used in section 2.3.1.2 are defined as follows:

$$f_{c\to b}^{l} = f_{ch} \boxplus \sum_i \rho_i(\boxplus_{i-1} f_{b\to c}^{l-1}) \tag{2.13}$$

$$f_{b\to c}^{l} = \sum_i \lambda_i(\otimes_{i-1} f_{c\to b}^{l}) \tag{2.14}$$

$$f_{out} = \sum_i L_i(\otimes_i f_{c\to b}^{\infty}) \tag{2.15}$$

where, $f_{c\to b}$ - denotes the density from the check node to the bit node,

$f_{b\to c}$ - denotes the density from the bit node to the check node,

f_{out} - denotes the density at the output of the bit node at the last iteration.

f_{ch} - denotes the density of channel LLR along the check node.

$\otimes_i$ - denotes the convolution operation performed with itself i times and corresponds to the sum of densities followed by appropriate truncation and

$\boxplus_{\mathrm{i}}$ - denotes the density obtained by $tanh$ operation performed i times followed by $atanh$ operation at the check node.

The reader is referred to [2] for more details of the density evolution equations and operations involved.

We fix the channel parameter, namely noise power in the Gaussian case, and initialize all the densities to an appropriate function. Usually, $f_{c\to b}$ and $f_{b\to c}$ are initialized to an impulse function around 0, and in the presence of channel observation, f_{ch} is initialized to the density of the channel LLR. We then begin computing the densities along the edges iteratively using the density update equations either until the density of LLR tends to a "point mass at infinity", $\delta(+\infty)$ or until the densities stop evolving. In the former case the probability of bit error $P_e \to 0$, however, in the latter the probability of error is given as the probability of f_{out} being non-positive and this quantity is bounded away from 0. The minimum threshold, ϵ_{thresh} obtained using DE is,

$$\epsilon_{thresh} = \inf\{\epsilon : f_{out}^{\infty}(\epsilon) \to \delta(+\infty)\} \tag{2.16}$$

2.4.2 Gaussian Approximation

In the BIAWGN channel, tracking densities exactly could be computationally complex and so, to simplify the analysis, we may approximate the densities by a Gaussian pdf. Since the underlying communication channel is a BIAWGN, the pdfs are symmetrical with respect to the input and can be described by one parameter i.e., the mean, μ of the Gaussian pdf since, the variance automatically follows as 2μ. We can compute the information rate between the LLR, L distributed according to a pdf $f(L)$ and the input message X under the $all\ zero\ codeword\ assumption$ using

$$I(L;X) = 1 - \int_{-\infty}^{+\infty} \log(1+e^{-L})\, f(L)dL \tag{2.17}$$

and then approximate the pdf $f(L)$ by a Gaussian pdf with $\mu = \mu_L$ such that, $J(\mu_L) = I(L;X)$. With slight abuse of notations, we will denote the Gaussian densities of LLR by $\mathcal{N}(\mu)$ to represent $\mathcal{N}(\mu, 2\mu)$ and the information corresponding to the pdf of LLR, $f(L)$ by $I(f)$ to represent $I(L;\ X)$. Then $J(\mu_L) = I(\mathcal{N}(\mu_L))$.

$$J(\mu) = 1 - \frac{1}{2\sqrt{\pi\mu}} \int_{-\infty}^{+\infty} \log\left(1+e^{-x}\right) e^{-\frac{(x-\mu)}{4\mu}} dx \tag{2.18}$$

Let the channel noise variance be σ^2, then we have

$$f_{ch} = \mathcal{N}\left(\frac{2}{\sigma^2}\right) = \mathcal{N}(\mu_{ch}) \tag{2.19}$$

as the density induced by the channel.

To obtain a Gaussian pdf corresponding to an information rate I, we define the J^{-1} function as a mapping from I to a Gaussian pdf of mean, μ such that, $I = J(\mu)$. Hence,

$$\mu = J^{-1}(I) \tag{2.20}$$

The means of the Gaussian densities along the edges are denoted by replacing f of the corresponding density by μ.

$$\mu^{l}_{b\to c} = J^{-1}\left(\sum_i \lambda_i J\left((i-1)\mu^{l-1}_{c\to b}\right)\right) \tag{2.21}$$

$$\mu^{l}_{c\to b} = J^{-1}\left(\sum_i \rho_i I\left(\mathcal{N}\left(\frac{2}{\sigma^2}\right) \boxplus \left(\boxplus_{\,i-1}\, \mathcal{N}\left(\mu^{l}_{b\to c}\right)\right)\right)\right) \tag{2.22}$$

$$\mu_{out} = J^{-1}\left(\sum_i L_i J(i\mu^{\infty}_{c\to b})\right) \tag{2.23}$$

2.5 Extrinsic Information and EXIT Charts

The extrinsic information is the amount of additional information that is obtained at the output of the decoder due to the underlying structure of the code when provided with an apriori information. The overall LLR, extrinsic LLR and the apriori LLR of the bit i at the output of the decoder are defined as follows

$$L_i = \frac{P\left(X_i = 0|\vec{Y}\right)}{P(X_i = 1|\vec{Y})} \tag{2.24}$$

$$L_{E,i} = \frac{P\left(X_i = 0|\overrightarrow{Y_{\sim\iota}}\right)}{P(X_i = 1|\overrightarrow{Y_{\sim\iota}})} \tag{2.25}$$

$$L_{A,i} = \frac{P(X_i = 0|Y_i)}{P(X_i = 1|Y_i)} \tag{2.26}$$

where $\vec{Y}$ denotes the received vector,

$\vec{Y_\iota}$ denotes the received vector except for the i-th bit , and

Y_i denotes the i-th bit.

These quantities are related as follows

$$L_{E,i} = L_i - L_{A,i} \tag{2.27}$$

2.5.1 EXIT Chart

An EXtrinsic Information Transfer chart, commonly called an EXIT chart, is a visualization technique to analyze and design iteratively-decodable error-correcting codes. In the current framework, EXIT charts are used on concatenated codes encoded as shown in Fig.4.

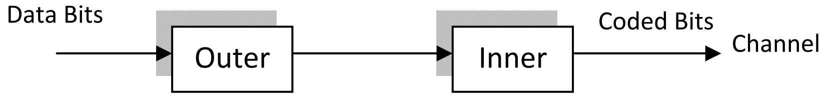

Fig.4 Concatenated encoder

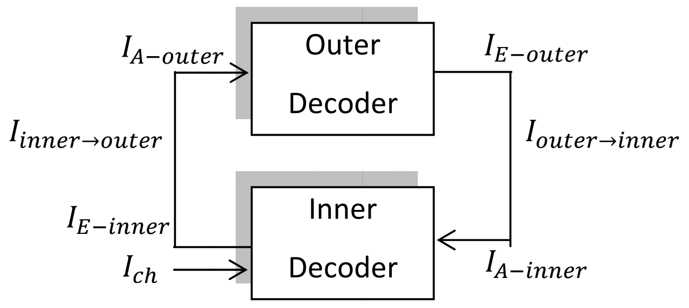

Fig.5 Decoder for a concatenated code

The iterative decoder in Fig.5 is used to decode a message encoded by the encoding scheme shown in Fig.4. In the iterative (joint) decoder, each component decoder passes extrinsic information denoted by I_E to the other decoder. This information in the case of BIMSCs are the extrinsic LLR values. EXIT charts can be used to design and analyze such concatenated coding schemes for asymptotic lengths.

2.5.2 Obtaining the EXIT Chart

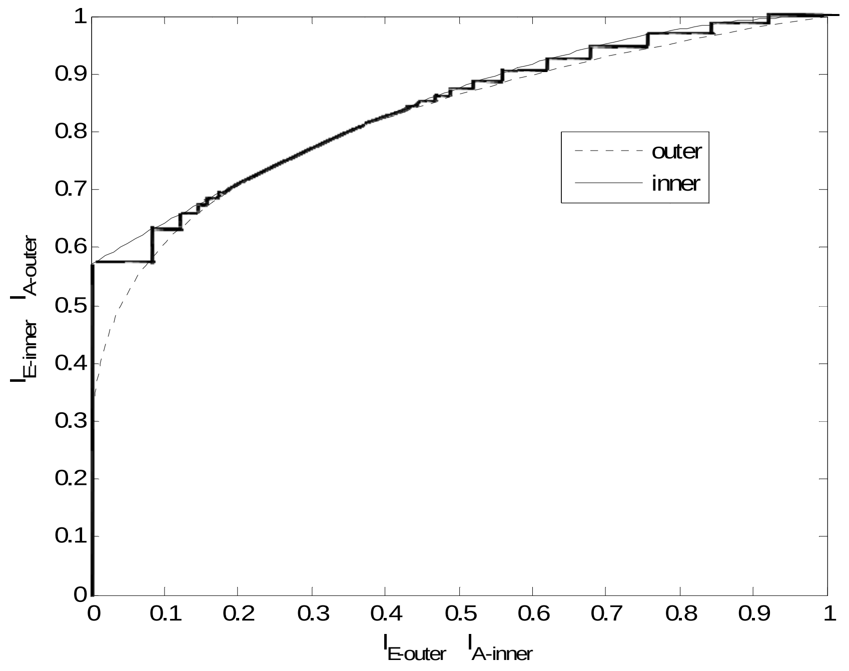

Fig.6 Information transfer path – convergence behavior

Let the mutual information at the output and input of the component code be

$$I_E = I(L_E; X) \tag{2.28}$$

$$I_E = I(L_A; X) \tag{2.29}$$

The EXIT chart is obtained by plotting the I_E vs I_A on a chart for both the inner code and the outer code on an appropriate axis as shown in Fig.6.

The convergence and the rate of the overall decoder can be graphically perceived using the EXIT Chart. The path indicates the typical information transfer characteristic between the component codes. One important property is that if the curve corresponding to the inner code lies below or crosses the outer code curve, then the convergence of decoding with arbitrarily low probability of error is not guaranteed.

2.5.3 Area Property

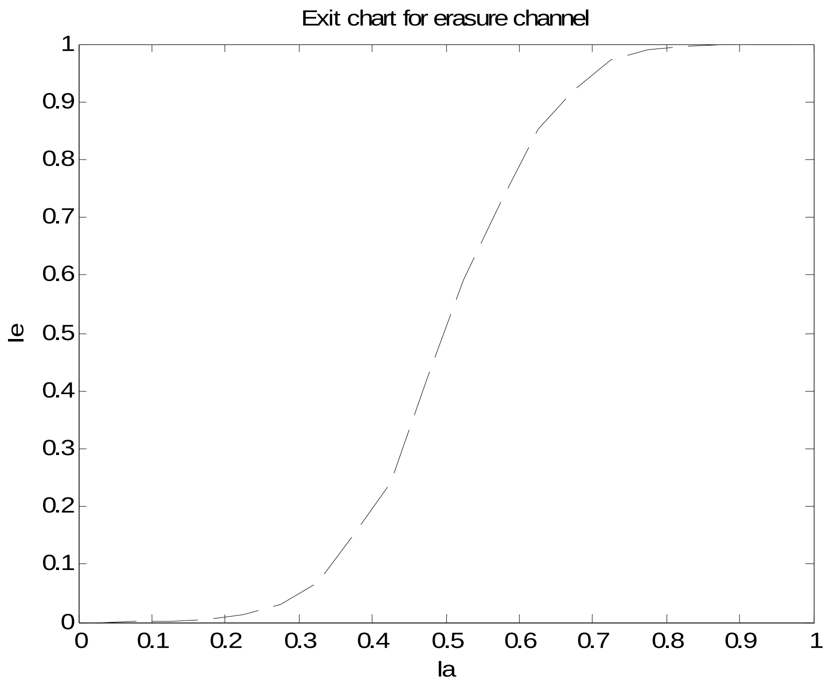

Fig.7 EXIT chart of a rate ½ - convolution code

The area theorem suggests that for each component code, the area A under the EXIT curve obtained by using an optimal decoder for the component code is equal to the $1 - Rate$ of the code. Fig.7 shows the EXIT chart for a rate 0.5 convolution code decoded using a BCJR-MAP Decoder. The area under the curve is precisely, 0.5 and hence, the convolution decoder under the BCJR Algorithm satisfies the Area theorem. Hence, the performance of any other decoder can be analyzed based on the area under the curve. The area theorem is exact in the case of an erasure channel but, is approximate in the Gaussian case, [10], [11].

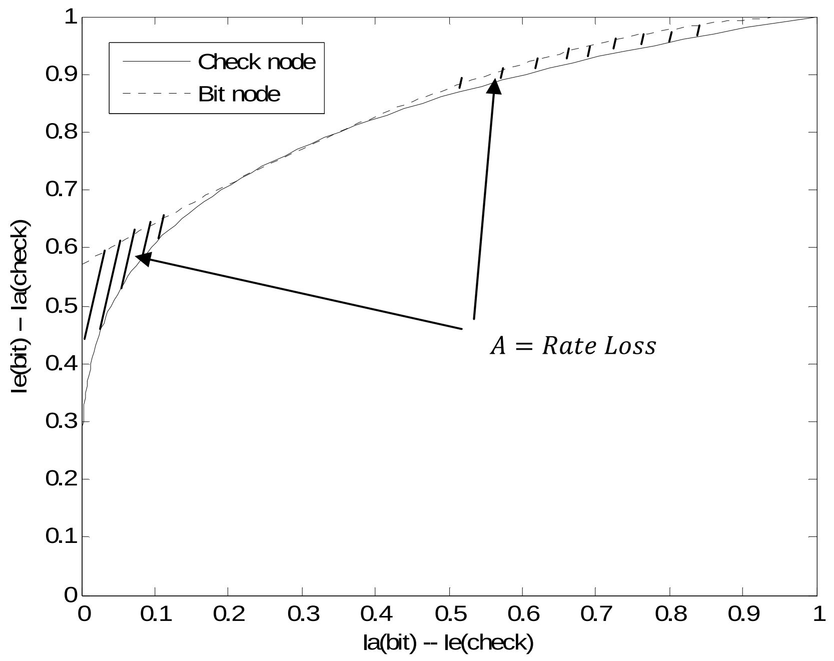

Fig.8 Area between check node and bit node of a LDPC code

More importantly, in the case of a joint decoder, the area between the two curves indicates the rate loss (with respect to capacity). Hence, this is an important measure of the efficiency of a particular design. We will be using this property, to analyze and design Raptor-like codes. This property is true even for codes like LDPC where the outer code can be thought of as the check node and the inner code as the bit node. The EXIT chart in Fig.8 illustrates the area property for a regular (3,6) LDPC code.

A code ensemble is said to be capacity achieving if the component codes satisfy the area theorem and the area between the component codes is arbitrarily small.

2.6 Luby Transform

These codes were introduced by Luby in [6] as the first class of rateless codes that are capacity achieving for all the erasure channels.

2.6.1 Encoder

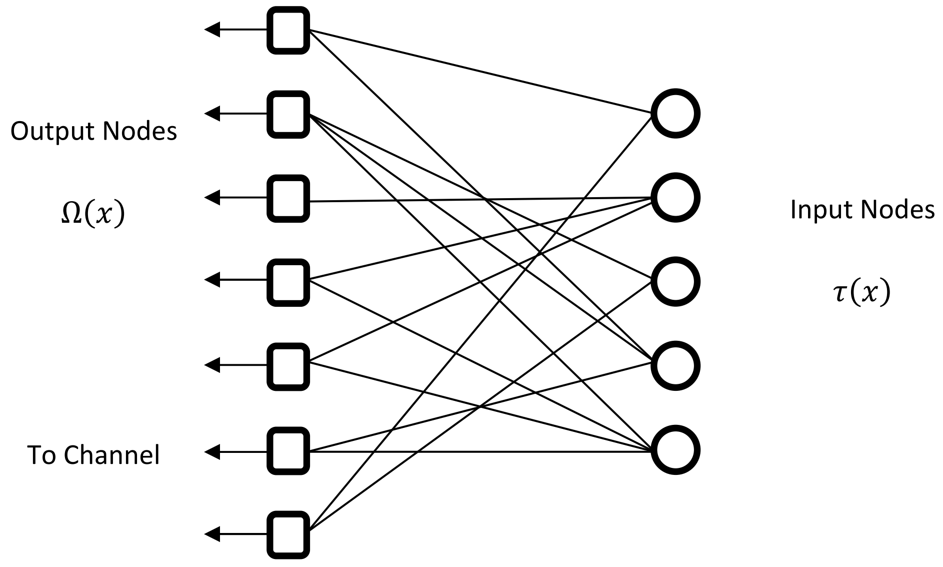

Fig.9 Bipartite graph of LT code

The encoder generates linear combinations of the original k data symbols also known as the input symbols to produce the encoded symbols also known as the output symbols according to a degree distribution, $\Omega(x) = \sum_d \Omega_d x^d$ where, Ω_d is the probability of the encoding symbol being degree d. These linear combinations are bitwise XOR operations that are performed on the data bits. Since the LT code is a rateless code, each encoded symbol is generated on the fly by choosing a degree d with a probability Ω_d of the encoding symbol, and then associating d randomly drawn input symbols denoted as edges. In this case, the input symbols, represent the bit nodes and the encoding symbols represent the check node as show in Fig.9.

The underlying distribution that ensures that all the information bits are covered in the process of encoding and keeps the decoding complexity low is the Ideal Soliton distribution given by,

$$\Omega(x) = \frac{1}{k} + \sum_{i=2}^{k} \frac{1}{i(i-1)} x^{i} \tag{2.30}$$

The output symbols can be thought of as the check node of the graph formed by placing edges between the output encoded symbols and the input symbols. Here $\Omega(x)$ is the check node degree distribution. The expected degree of each encoding symbol is $\Omega'(1) \approx \ln(k)$ for large k. Hence, the number of edges in the graph is $\approx k \ln k$.

2.6.2 Decoder

LT codes in the erasure channel can be viewed as special class of LDPC codes without transmitting the bit node values through the channel. The output symbols correspond to the check nodes of a LDPC code and the input symbols to the bit nodes. But the notion of rate is not the same as that of a LDPC code since the number of check nodes is greater than the bit nodes. LT decoding is equivalent to decoding a LDPC code on the bipartite graph and hence, the same message passing rules apply.

However, here we present an intuitive explanation of how the LT decoder works. The decoding process involves decoding each input symbol in an iterative manner. An input symbol can be recovered if it is connected to a neighboring output symbol of degree 1, and the value of the input symbol is equal to that of the output symbol. Whenever, an output symbol of degree 1 is encountered at any stage of the decoding process, the input symbol is decoded and then all the edges connected to the input symbol are removed after XORing the value of the input symbol with its neighbors. This process continues until all the input symbols are recovered. The complexity of this decoder is proportional to the number of edges i.e., $O(k \ln k)$.

The Ideal Soliton ensures that at every stage on an *average* at least one output symbol is of degree 1 to keep the decoding process continuing until all the edges are removed. A decoding failure occurs when the decoding process stops before all of the input symbols have been decoded. The required probability of decoding failure is $1/k^c$ for some arbitrary positive constant c.

However, the Ideal Soliton distribution works poorly in practice because the distribution was created with the average behavior in mind. Hence, to make the decoding process more robust to deviations from the expected behavior and to improve the probability of decoding success for finite length codes, Luby introduced the Robust Soliton distribution which is a slight modification of the Ideal Soliton to take into account the probability of random walk of length k deviating from its mean behavior. This improvement in the performance comes with a very slight increase in overhead; the required number of output symbols required is now $K = k + O(\ln^2\left(\frac{k}{\delta}\right)\sqrt{k})$ instead of k. However, the complexity of these codes still remains the same i.e., $O(k \ln k)$.

2.7 Raptor Codes

A Raptor code consists of two component codes - an outer code that encodes the information bits into intermediate bits and an inner code that converts the intermediate bits to an infinite stream of coded bits as shown in Fig.10. Typically, the outer code is a LDPC code [12] and the inner code is similar to a LT code. The outer code ensures that all the required data bits can be decoded once the inner decoder recovers a major portion of the intermediate bits (unlike the LT codes in [6], where all the input bits have to be decoded).

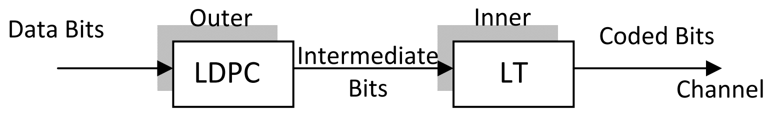

Fig.10 Raptor Codes

We will assume that k information bits are encoded by the LDPC code of rate $r_{LDPC}(0 < r_{LDPC} \leq 1)$ to $n = k/r_{LDPC}$ bits as shown in Fig.11. Similarly, let us define the rate of the inner LT code $(r_{LT} \geq 0)$ as the fraction of the output bits received at the receiver to the actual number of input bits. Hence, the LT process encodes the intermediate bits of length n into $N = n/r_{LT}$ coded bits. However, it is important to note that the actual number of bits produced by the LT process is an infinite stream of bits, but we will assume that the receiver stops receiving these bits once any N of them arrive. Hence, with minor abuse of notation we will continue to denote the rate of the LT code as r_{LT} though in fact it is rateless. The overall overhead of the Raptor code is $\epsilon = N/k \ - 1$ and is an important measure of performance.

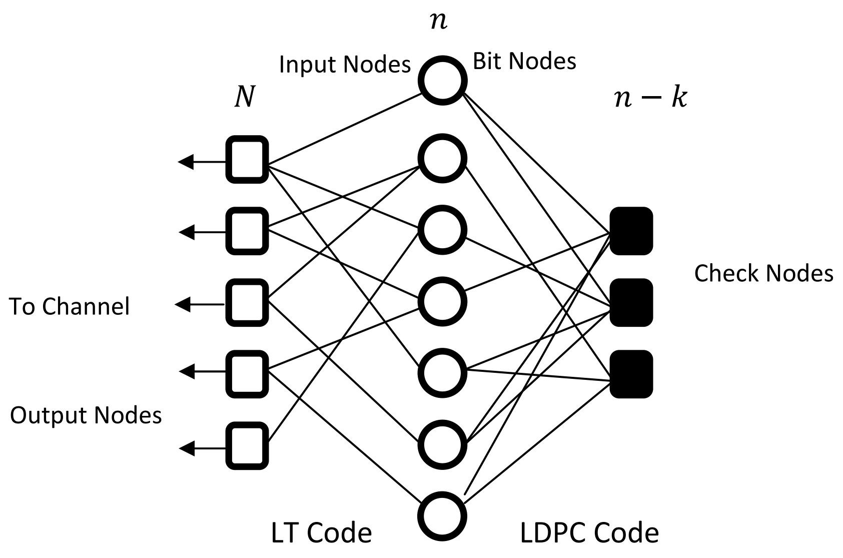

Fig.11 Tanner graph of Raptor codes

Let $\lambda(x)$ and $\rho(x)$ denote the degree profiles of the LDPC bit and check edges respectively and $L(x)$ and $R(x)$ denote the degree profiles of the LDPC bit and check nodes respectively. Similarly, let us define $\omega(x)$ and $\Omega(x)$ to represent the edge and node degree profiles of the LT output symbols respectively. The edge profile and the node profile of the input symbols of the LT code are essentially the same for sufficiently large N and are given by $\tau(x)$. The random construction of a LT code by assigning edges of the output bits based on $\Omega(x)$ to randomly chosen input symbols induces a Poisson distribution (for large block lengths) on the input edge and node degree profile $\tau(x)$ with parameter α. The average degree of the input node, α determines the complexity of the Raptor Code partially.

2.7.1 Raptor Codes for the Erasure Channel

The output symbol degree distribution of the inner LT code used here is a truncated version of the Ideal Soliton distribution of the LT process with a maximum degree of D. An appropriate weight is assigned to the output symbols of degree one to make it a valid probability distribution. Hence, based on the desired value of ϵ, we have a different maximum degree D (and also complexity) which is a function of ϵ and we get a class of codes by spanning a range of ϵ. The resulting distribution of the LT code's encoding symbols is given by,

$$\Omega(x) = \frac{1}{\mu+1}(\mu x + \sum_{i=2}^{D} \frac{x^i}{i(i-1)} + \frac{x^{D+1}}{D}) \tag{2.31}$$

where $\mu = \left(\frac{\epsilon}{2}\right) + \left(\frac{\epsilon}{2}\right)^2$ and $D = \lceil 4(1+\epsilon)/\epsilon \rceil$.

The rate of the LDPC code is given by,

$$r_{LDPC} = \frac{1+\frac{\epsilon}{2}}{1+\epsilon} \tag{2.32}$$

This random construction of the LT graph induces a Poisson distribution on the input node with parameter α,

$$\tau(x) = e^{\alpha(x-1)} \tag{2.33}$$

where α is the average degree of the input symbol.

$$\alpha = \Omega'(1) r_{LDPC}(1+\epsilon) \tag{2.34}$$

The overall overhead, ϵ of the Raptor code in terms of α is given by

$$\epsilon = \frac{\alpha}{\Omega'(1)\, r_{LDPC}} - 1 \tag{2.35}$$

and the overall rate of the code is

$$r_{Raptor} = \frac{\Omega'(1)\, r_{LDPC}}{\alpha} \tag{2.36}$$

The Raptor encoded bits that are sent across the channel are formed in two stages. The first stage of the encoding process is the LDPC code generation that outputs the intermediate symbols from the actual data bits using the generator matrix of a very high rate LDPC code, $r_{LDPC} = (1 + \epsilon/2)/(1 + \epsilon)$. This is followed by the inner LT encoder which, follows on the same lines as the one described in section 2.6.1 and the output symbols are directly sent across the channel. Shokrollahi used a low degree bit edge profile, $\lambda(x) = (2x + 3x^2)/5$ and then randomly assigned the edges of the bit nodes to check nodes. This construction is similar to that of the LT process but with the roles of the check and the bit node interchanged.

2.7.1.1 Decoder Structure

At the receiver, an *one-shot decoder* is employed as soon as any N output symbols have been received. The decoding follows logically from the encoder structure. First the inner LT decoder recovers at least $(1 - \epsilon/(4 + 4\epsilon\ ))n$ of the intermediate bits using the decoding sequence described in 2.6.2. The remaining bits are recovered using the inner LDPC decoder using the message passing decoder for the erasure channel (Section 2.3.1.1).

The decoding complexity of the LT code and the LDPC code is proportional to the average number of edges of these codes and hence, the overall complexity is the sum of the edges corresponding to the two component codes. Since the maximum degree and the average degree of the input symbols are fixed and do not vary with the length of the data bits, the decoding complexity is $O(1)$ making it practically implementable for large number of packets.

2.7.2 Raptor Codes for the BIAWGN Channel

In [9], Etesami and Shokrollahi extend the idea of Raptor codes to the BIMSCs and more specifically to the BIAWGN Channel. They introduce the design of the LT code for the one-shot decoder and the design objective is to decode most of the intermediate bits and allow a very high rate LDPC code decoder to recover the bits that were not decoded (covered) by the LT process, similar to the one proposed in the erasure case.

At this point it is worth mentioning, that in [9], the authors design the inner and outer codes independently. Further, in [13] designs for a particular channel parameter are proposed (SNR in the Gaussian case and p in the case of the BSC) and the hope is that it will perform well for a range of channels with parameters of interest.

The asymptotic length design is based on a refined Gaussian approximation where the authors use the empirical mean (obtained by sampling the distribution many times) to approximate the mean of the messages from the output symbol nodes to the input symbol nodes. The objective of the LT code design is to obtain the minimal overhead for a particular channel with capacity given by C, and a given average degree of the input symbol, α. It turns out that this problem, is a linear programming optimization which can be formulated as follows

$$\min_{\omega} \; C\alpha \sum_{d=1}^{D} \frac{\omega_d}{d} \tag{2.37}$$

Subject to

1. $\forall\, \mu \in (0, \mu_0)\,, \quad \forall\, i = 0, \dots, N-1 \;\; : \;\; \alpha \sum_{d=1}^{D} \omega_d f_d(\mu_i) > \mu_i$
2. $\sum_{d=1}^{D} \omega_d = 1$ and $\forall\, d = 1, \dots, D : \; \omega_d \geq 0$

The first constraint implies that the mean of the density should increase at every iteration until it reaches a very high value close to ∞ but $\neq \infty$. This is based on the assumption that the LDPC decoder can evolve this density to a point mass at

∞. While the second constraint implies that the resulting $\omega(x)$ should be a valid probability distribution.

Once the degree profiles of the LT codes are decided, the encoding procedure follows exactly from section 2.7.1. However, the decoder is similar to the one used in section 2.7.1 except for the messages that are passed and the message update rules are different and are as described in 2.3.1.2. A message passing decoder suited for the underlying BIMSC is employed independently on the LT code and the LDPC code. MPA is performed on the LT code until the BER stops decreasing. The LDPC decoder recovers the remaining bits. However, the authors do not mention any particular design or technique to design these inner codes to suit a particular application in the Gaussian case. Further, they do not provide any performance and complexity trade-off based code design.

3 EXIT CHART BASED DESIGN AND ANALYSIS FOR THE ERASURE CHANNEL

3.1 Introduction

In this section, we consider the analysis and design of Raptor-like codes under a different decoding algorithm than the one-shot decoder considered in [7]. The proposed decoder is a joint decoder that iteratively passes messages between the LT decoder and the LDPC decoder. Our objective is to develop a design procedure for optimizing the degree profiles described by $\lambda(x), \rho(x), \tau(x)$ and $\omega(x)$ under the iterative decoder framework instead of the one-shot decoding setup. The design of such codes can be intuitively understood using the idea of EXIT Charts [10]. Using this approach, we show that the design of degree profiles can be formulated as a linear programming optimization problem. Moreover, the erasure channel results help us to build the necessary intuition for the more complex Gaussian case.

3.2 Joint Decoder

The design of the component codes in [7] was based on an one-shot decoder, where the inner code is decoded followed by the decoding of the outer code. Here, we introduce the concept of iteratively decoding the inner code and the outer code. In this joint decoder, the information at the output of the inner decoder I_{E-LT} is fed-back as input to the outer decoder as I_{A-LDPC}, and the output of the outer decoder I_{E-LDPC} is

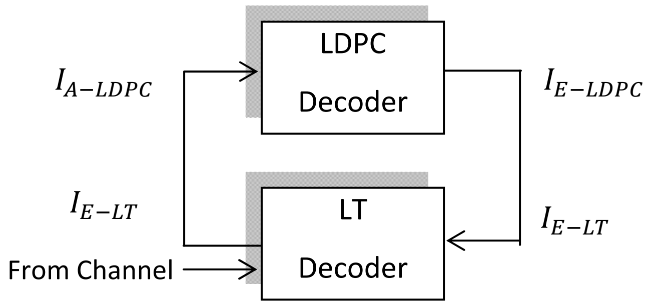

Fig.12 Iterative decoding of Raptor codes

fed back to the inner decoder as I_{A-LT} (Fig.12). This forms one iteration in the decoder and this process is repeated until all the information bits are successfully recovered. The messages that are passed on the underlying graph of the inner and outer codes are shown in Fig.13.

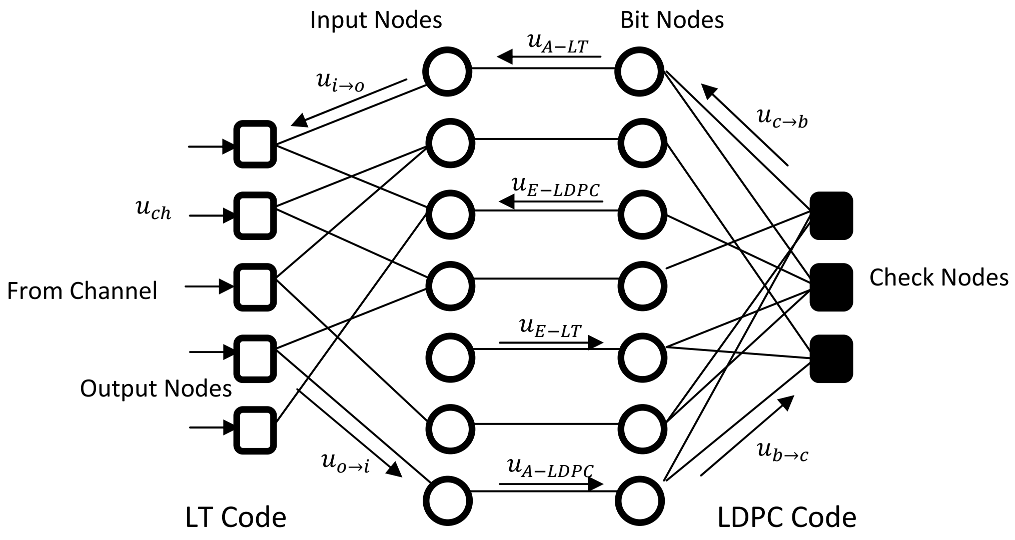

Fig.13 Bipartite graph showing messages along the edges

3.2.1 Decoding Schedule

The two component decoders used in the joint decoder are in turn iterative decoders. Hence, we have two levels of iterations, one within the component decoders and the other between them. This gives rise to several scheduling options for message passing, but the two most practical schedules are to:

- Iterate once within the first component code followed by once at the second component code but pass extrinsic information between them after each iteration.
- Iterate within each component code until a fixed point is reached and then pass extrinsic information to the other code and vice versa.

In the erasure channel, the scheduling does not affect the resulting information content, and hence, without loss of generality we will use the second scheduling option to design and analyze the performance of these Raptor codes.

3.2.2 Complexity

We will consider a particular implementation of the aforementioned decoder called the peeling decoder [14]. As the name implies, the decoding process can be conceptually understood as removing (peeling) the edges of the underlying graph. A peeling decoder decodes a bit node message whenever it encounters a check node of degree one connected to that bit node. The edges that are connected to the recovered bit are removed after recovering a bit node value, and the corresponding check node values are updated by XORing the bit node value that was removed. This is continued until no more edges are present in the graph or until the graph cannot be reduced any further.

In the erasure case, a peeling decoder is equivalent to a message passing decoder. The complexity of the peeling decoder is proportional to the number of edges, η in the graph on which the decoder works. Hence, the overall complexity, Δ of a Raptor code is given by,

$$\Delta \propto \left(\frac{L'(1)}{r_{LDPC}}\right) + \left(\Omega'(1) \times (1+\epsilon)\right) \tag{3.1}$$

In [7], the LT code is decoded until almost all the bits are recovered, then the LDPC decoder recovers the rest of the bits. Hence, the decoder eventually removes almost all the edges in both the LT and LDPC graphs. Hence, the complexity of the proposed decoder is comparable to the existing implementation in [7] on the same graph.

3.2.3 Reliability

The reliability of the Raptor code in [7] is analyzed using the probability of decoding failure. A decoding failure occurs when the receiver is not able to decode the frame. For the codes considered in [7], based on stopping set analysis [15], have a decoding failure given by $1/k^c$. Since the iterative decoder, used in our case also uses similar component codes, by the same arguments we can show that the reliability is $1/k^{c'}$, but for some $c' \neq c$.

3.3 Density Evolution

A unique property of the erasure channel, that helps the universal code design is that the erasure channel has a capacity of $1 - p_{eras}$. Hence, without loss of generality, we can say that, any random rateless code design that works for the case $p_{eras} = 0$ will work equally well for the $p_{eras} > 0$ too. Hence, for the erasure channel, the EXIT chart and the density evolution equations are obtained by assuming that the channel erasure probability is 0 and that the required number of $(1 + \epsilon)k$ encoded symbols have been received perfectly.

3.3.1 Fixed Point Characterization

In a peeling decoder, edges are removed iteratively until it reaches a point after which no check nodes with degree one are available to sustain the process. At this point, the decoder is said to have reached a fixed point and the decoder can no longer provide any more information to the other component decoder.

For a given $\lambda(x)$, $\rho(x)$,$\tau(x)$ and $\omega(x)$, it is possible to obtain the fixed point analytically. Let p_{A-LT} (p_{A-LDPC}) denote the apriori erasure probability of the LT (LDPC) decoder from the LDPC (LT) decoder and p_{E-LT} (P_{E-LDPC}) denote the corresponding extrinsic erasure probability at the end of the decoding process within one component code. In the erasure channel, tracking the densities is equivalent to just tracking the erasure probability of the messages along the edges.

Let $p^{l}_{c\to b}$ denote the erasure probability of a check to bit message, $u^{l}_{c\to b}$ at the l-th iteration of the LDPC decoder, which is defined over the set {0,1,?}. Under the all zero-codeword assumption, the density evolution equations of the LDPC code are as follows:

$$p_{c\to b}^{l+1} = 1 - \rho\left(1 - p_{A-LDPC}\lambda\left(p_{c\to b}^{l}\right)\right) \tag{3.2}$$

$$p_{c\to b}^{l+1} < p_{c\to b}^{l} \tag{3.3}$$

$$p_{E-LDPC} = L(p_{c\to b}^{\infty}) \tag{3.4}$$

where, $p_{c\to b}^{\infty}$ denotes the probability of erasure on reaching a fixed point that ends the iterations within the LDPC decoder.

Let $p_{o\to i}^{l}$ denote the erasure probability of the output node to input node message $u_{o\to i}^{l}$ at the l-th iteration of the LT decoder. The density evolution equations for the LT decoder are as follows:

$$p_{o\to i}^{l+1} = 1 - \omega\left(1 - p_{A-LT}\tau\left(p_{o\to i}^{l}\right)\right) \tag{3.5}$$

$$p_{o\to i}^{l+1} < p_{o\to i}^{l} \tag{3.6}$$

$$p_{E-LT} = \tau(p_{o\to i}^{\infty}) \tag{3.7}$$

These above equations provide us the exact evolution of the erasure probability with iteration and also the relationship between the apriori information and the extrinsic information given the degree profiles of the component codes.

3.4 Analysis Using EXIT Charts

The EXIT chart of any given Raptor code is obtained by plotting the information rates, $(1 - p_{eras})$ i.e., I_{A-LT} vs. I_{E-LT} and I_{E-LDPC} vs. I_{A-LDPC} on the same plot as shown in Fig.14 and Fig.15. In the erasure case, the charts can also plotted with the erasure probability on each axis i.e., p_{A-LT} vs. p_{E-LT} and p_{E-LDPC} vs. p_{A-LDPC}. The later EXIT Chart is just a flipped version of the former along both the axis. When the

EXIT chart is obtained for the component codes, if the two curves overlap each other or exhibit crossovers then the decoding procedure will not converge to an erasure probability of 0. Hence, the probability of decoding error is bounded away from 0 and the given code is inefficient under the iterative decoding scheme.

On the other hand if the plots do not touch each other and they lie one above the other, it indicates that asymptotically in length the erasure probability of the code ensemble converges to 0. However, if they are too far apart as shown in Fig.14, the codes suffer a rate loss given by the Area Property, 2.5.3. In our framework, it implies an excess overhead proportional to the area between the curves.

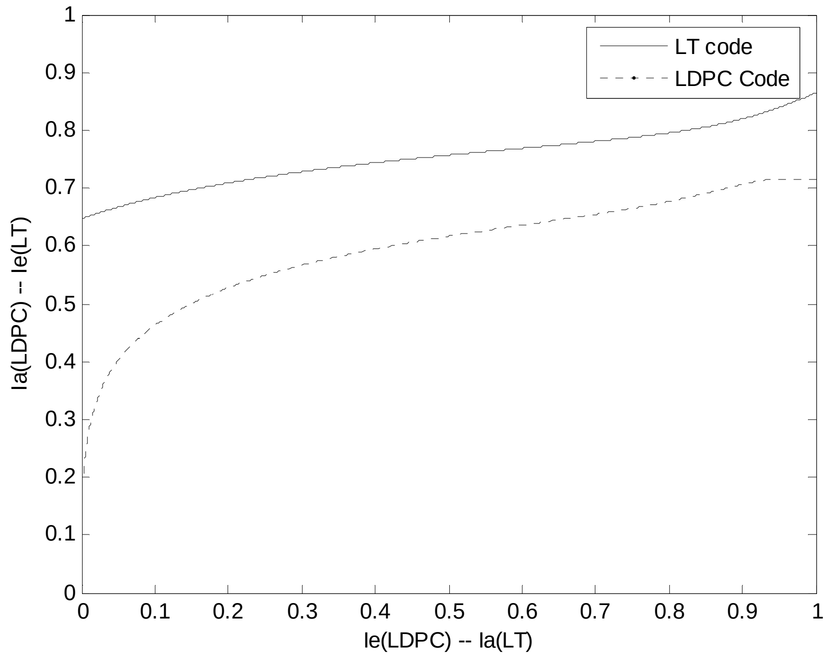

Fig.14 EXIT chart - Rate loss

3.4.1 Overhead

A code is near optimal, if the area between the inner code and outer code EXIT curves is negligible and if the curves do not exhibit crossovers. This is the criterion that we use to construct asymptotically good codes. In other words, we search for profiles that can be near optimal, based on the properties of the EXIT charts. The performance of the Raptor code is determined by the overhead and hence, it is important to understand this term.

There are 2 reasons for the non-zero overhead of the given degree profiles,

- Due to a non-zero area between the two curves obtained
- Due of the rate loss of the LDPC code as a result of the sub-optimality of the iterative decoder.

When we want codes which perform significantly close to capacity, we are interested in finding profiles for both the LT and LDPC codes such that the overall overhead is minimal. However, when we want to design LT processes to suit existing implementation of the LDPC codes we will use the overhead increase due to the non-zero area between the two EXIT curves as our measure of performance.

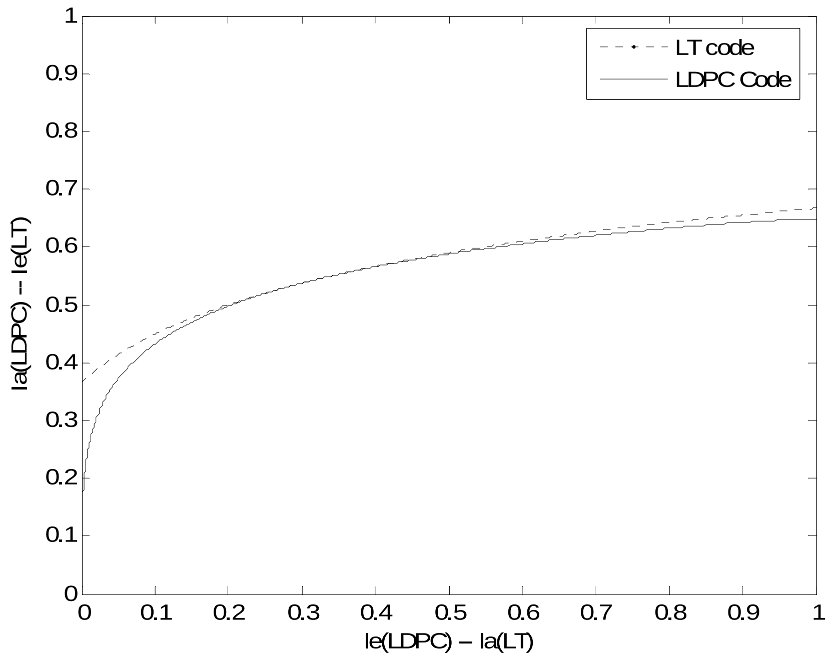

Fig.15 EXIT chart for the erasure channel

3.5 Design Using EXIT Charts

From previous sections, it is clear that to design codes close to the capacity of the erasure channel, we need to optimize the LDPC code and LT code degree profiles such that we have the two curves barely touching each other for the entire range of I_{A-LT} and I_{A-LDPC}. This can be formulated as a linear programming problem based on the EXIT Charts under certain valid assumptions. The design of almost universal codes is based on the iterative optimization of the component code degree profiles until their EXIT curves are sufficiently close to one another.

The decoding schedule that we shall consider to make our design problem tractable is as follows:

- Perform MPA within the first component decoder with the apriori information $I_{A-C1}^{l} = I_{E-C2}^{l-1}$ from the other component decoder and starting with all the messages along the edges erased until a fixed point is reached.
- Pass the extrinsic information I_{E-C1}^{l} corresponding to the fixed point to the second component decoder.
- Perform MPA within the second component decoder with the apriori information $I_{A-C2}^{l} = I_{E-C1}^{l}$ from the first component decoder starting with all the messages along the edges erased until a fixed point is reached.
- Pass the extrinsic information I_{E-C2}^{l} corresponding to the fixed point to the first component decoder.

This is iteratively performed until we decode all the bits. Thus, to ensure that all the bits are successfully decoded, the design problem reduces to finding LT and LDPC profiles such that at the end of one whole iteration, $I_{E-C2}^{l} > I_{E-C2}^{l-1}$ and $I_{E-C1}^{l} > I_{E-C1}^{l-1}$ until $I_{E-C1} = 1$ or $I_{E-C2} = 1$. Suppose one of these conditions are violated i.e., $I_{E-C2}^{l} = I_{E-C2}^{l-1}$ or $I_{E-C1}^{l} = I_{E-C1}^{l-1}$, then either decoder cannot produce any information, because it has already reached a fixed point for the given apriori.

3.5.1 Optimization of the LT Code for a Given LDPC Code

Let us say that we are given a $\lambda(x)$ and $\rho(x)$, then we can obtain the bit node profile, $L(x)$ from the bit edge profile $\lambda(x)$ as follows,

$$L(x) = \frac{\int_0^x \lambda(z)dz}{\int_0^1 \lambda(z)dz} \tag{3.8}$$

Our objective is to find the degree profile $\omega(x)$ of the LT code for a fixed complexity α that matches closely with the LDPC code i.e., design a profile such that the EXIT curves of the LDPC and LT codes are as close to each other as possible.

When the LDPC profiles are known, the LDPC EXIT curve is available. For the rateless code to be successfully decoded, we need to ensure that at every point along the LDPC curve defined by (I_{A-LDPC}, I_{E-LDPC}), a desired LT profile should be able to reduce the extrinsic erasure or equivalently provide more extrinsic information, i.e., $I_{E-LT}(I_{A-LDPC}) > I_{A-LDPC}$.

3.5.1.1 $\boldsymbol{\tau^{-1}}$ *Function*

Given $\rho(x)$, $\lambda(x)$ we can perform actual density evolution for an input, I_{A-LDPC} and obtain the extrinsic information, $I_{E-LDPC}(I_{A-LDPC}) = I_{A-LT}(I_{A-LDPC})$ and thus obtain the EXIT curve. Each p_{A-LDPC} corresponds to an output node to input node message fixed point erasure probability in the previous iteration of the LT decoder, $p_{o\to i}^{\infty-}$. The relation between $p_{o\to i}^{\infty-}$ and p_{A-LDPC} is given by $p_{A-LDPC} = \tau(p_{o\to i}^{\infty-})$.

From the density evolution equations for the LT,

$$p_{o\to i}^{l+1} = 1 - \omega\left(1 - p_{A-LT}\tau\left(p_{o\to i}^{l}\right)\right) \tag{3.9}$$

$$p_{E-LT} = \tau(p_{o\to i}^{\infty+}) \tag{3.10}$$

it is clear that the density evolution equations are linear in the coefficient of $\omega(x)$ for a given $p_{o\to i}$ and p_{A-LT}. However, they cannot be expressed as a linear function of ω_i's for a given p_{A-LDPC} and $p_{A-LT} = p_{E-LDPC}$. So in practice, we need the erasure probability of the output node to input node message at the fixed point, $p_{o\to i}^{\infty-}$ corresponding to p_{A-LDPC} to formulate the design problem as a LP. Since, we fix α for the optimization problem, and $\tau(x)$ is invertible we can obtain $\tau^{-1}(y)$ as

$$\tau^{-1}(y) = \frac{\ln(y)}{\alpha} + 1 \tag{3.11}$$

Hence, for a given p_{A-LDPC} we have,

$$p_{o\to i}^{\infty-} = \tau^{-1}(p_{A-LDPC}) \tag{3.12}$$

3.5.1.2 LDPC EXIT Function

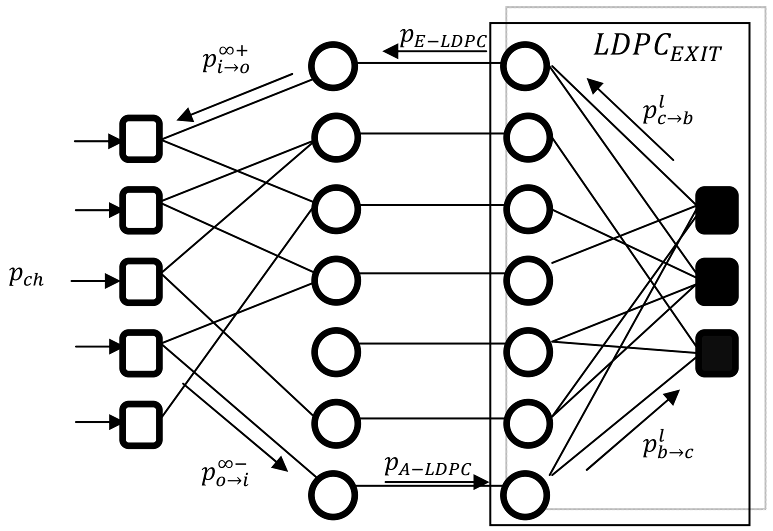

Fig.16 LDPC EXIT function (Erasure)

Another efficient way of expressing the design problem using the LP framework is to express the density evolution equations for the LDPC code in terms of the fixed point of LT, $p_{o\to i}^{\infty-}$ instead of p_{A-LDPC}. This way, we need not implement the inverse function $\tau^{-1}(x)$. The apriori for the LDPC in terms of the fixed point is given by

$$p_{A-LDPC} = \tau(p_{o\to i}^{\infty-}) \tag{3.13}$$

The output of the LDPC is p_{E-LDPC} which is obtained by starting with $p_{c\to b}^{0} = 1$ and iteratively applying update equations for LDPC function for a fixed $p_{o\to i}^{\infty-}$:

$$p_{c\to b}^{l+1} = 1 - \rho\left(1 - \tau(p_{o\to i}^{\infty-})\lambda\left(p_{c\to b}^{l}\right)\right) \tag{3.14}$$

until we reach, $p_{c\to b}^{\infty}$. Then the extrinsic at the output of the LDPC function is

$$p_{E-LDPC} = L(p_{c\to b}^{\infty}) = p_{A-LT} \tag{3.15}$$

Hence, the LDPC function shown in Fig.16 defines a mapping $p_{o\to i}^{\infty-}$ to p_{E-LDPC}/p_{A-LT}

$$p_{A-LT} = LDPC_{EXIT}(p_{o\to i}^{\infty-}) \tag{3.16}$$

3.5.1.3 LP Formulation

The best profile $\omega_{opt}(x)$ for a given complexity, α and maximum degree of $\omega(x)$, $\omega_{deg-max}$ can be obtained as the solution to the linear programming problem

$$\min_{\omega} \sum_{i=0}^{\omega_{deg-max}} \frac{\omega_i}{(i+1)} \tag{3.17}$$

subject to:

1. Using the τ^{-1} function

$$x > \sum_i \omega_i\left(1 - (1 - I_{A-LT})\tau(1-x)\right)^{i-1}$$
$$\forall x \in [0, 1 - \tau^{-1}(1 - I_{E-LT}(I_{A-LT}))]$$
$$\forall I_{A-LT} \in [0,1)$$

or using the LDPC EXIT function

$$x > \sum_i \omega_i\left(1 - (LDPC_{EXIT}(p_{o\to i}^{\infty-}))\tau(1-x)\right)^{i-1}$$
$$\forall x \in [0, 1 - p_{o\to i}^{\infty-}]$$
$$\forall p_{o\to i}^{\infty-} \in (0,1]$$

2. $\sum_i \omega_i = 1$

3. $0 \leq \omega_i \leq 1 \qquad \forall\, i \in 0, 1, 2, \ldots\ \omega_{max-deg}$

This objective function ensures that the resulting $\omega(x)$ has the least overhead requirements i.e., minimizes r_{LT}. The first constraint implies that the resulting LT process with $\omega(x)$, does not reach a fixed point below the LDPC curve for all values of the apriori to the LT code. This constraint also implies that when the decoding process is started with the new apriori after initializing the messages along the edges to erasures, it does not have a fixed point anywhere below the LDPC curve and hence, produces a larger extrinsic information. This constraint is usually enforced on a finely sampled region below the LDPC curve. The last two constraints ensure that the coefficients of $\omega(x)$ form a valid probability distribution.

3.5.1.4 Choice of α

An appropriate value for α has to be chosen to ensure that we can get started with the optimization. If α is chosen to be too low then the constraint may not yield any feasible solution. Usually, α is chosen large enough to ensure that we obtain a feasible solution and then appropriately adjusted to suit the complexity requirements. As a rule of thumb, for a given LDPC, if p_{A-LDPC}^{thresh} denotes the maximum p_{A-LDPC} (erasure probability) corresponding to an extrinsic information of 1, then

$$\alpha \geq -\ln\left(p_{A-LDPC}^{thresh}\right) \tag{3.18}$$

For values of α very close to the equality, LP may not yield solutions even for large values of $\omega_{max-deg}$.

3.5.2 Optimization of the LDPC Code for a Given LT Code

Let us say that we are given a $\tau(x)$ and $\omega(x)$, our objective is to find the degree profile $\rho(x)$ of the LDPC code for a fixed bit edge degree profile $\lambda(x)$ that matches closely with the LT code i.e., design a profile such that the EXIT curves of the LDPC and

LT codes are as close as possible. The average degree of the bit node and r_{LDPC} determine the complexity.

When the LT profiles are known, the LT EXIT curve is available. For the rateless code to be successfully decoded, we need to ensure that at every point along the LT curve defined by (I_{A-LT}, I_{E-LT}), a desired LDPC profile should be able to provide more extrinsic information or reduce the extrinsic erasure, i.e., $I_{E-LDPC}(I_{A-LT}) > I_{A-LT}$.

3.5.2.1 $\boldsymbol{L^{-1}}$ *Function*

Given $\tau(x)$, $\omega(x)$ we can perform density evolution for an input, I_{A-LT} and obtain the extrinsic information, $I_{E-LT}(I_{A-LT}) = I_{A-LDPC}(I_{A-LT})$ and thus obtain the EXIT curve. Each p_{A-LT} corresponds to a check node to bit node message fixed point erasure probability of the LDPC decoder, $p_{c\to b}^{\infty-}$ in the previous iteration. The relation between $p_{c\to b}^{\infty-}$ and p_{A-LT} is given by $p_{A-LT} = L(p_{c\to b}^{\infty-})$.

From the density evolution equations for the LDPC,

$$p_{c\to b}^{l+1} = 1 - \rho\left(1 - p_{A-LDPC}\lambda\left(p_{c\to b}^{l}\right)\right) \tag{3.19}$$

$$p_{E-LDPC} = L(p_{c\to b}^{\infty+}) \tag{3.20}$$

it is clear that the density evolution equations are linear in the coefficient of $\rho(x)$ for a given $p_{c\to b}$ and p_{A-LDPC}. However, they cannot be expressed as a linear combination of in $p_{A-LDPC} = p_{E-LT}$ and p_{A-LT}. So in practice, we need the erasure probability of the check to bit message at the fixed point, $p_{c\to b}^{\infty-}$ corresponding to p_{A-LT} to formulate the design problem as a LP.

$\lambda(x)$ is fixed for the optimization problem and hence, $L(x)$ is known and we need to obtain $L^{-1}(y)$. Since $L(x)$ is a polynomial, the desired mapping $L^{-1}(y)$ can be obtained as the roots of the polynomial $L(x) = y$. However, this could result in multiple roots, but we are interested only in roots $\in [0,1]$. Since $L(x)$ has only non-negative powers and non-negative coefficients, it is a non-decreasing function in

$x \; \forall \, x \in (0,1)$. Hence, we can claim there exists only one solution to $L^{-1}(y)$ between $(0,1)$. Hence, for a given p_{A-LT} we have,

$$p_{c\to b}^{\infty-} = L^{-1}(p_{A-LT}) \tag{3.21}$$

3.5.2.2 LT EXIT Function

Another efficient way of expressing the design problem using the LP framework is to express the density evolution equations for the LT code in terms of the fixed point of LDPC, $p_{c\to b}^{\infty-}$ instead of p_{A-LT}. This way, we need not implement the inverse function $L^{-1}(x)$. The apriori for the LT in terms of the fixed point is given by

$$p_{A-LT} = L(p_{c\to b}^{\infty-}) \tag{3.22}$$

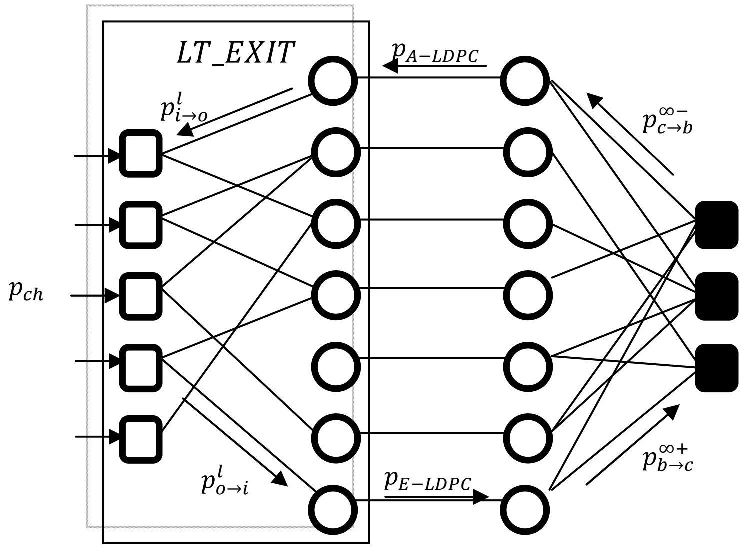

Fig.17 LT EXIT function (Erasure)

The output of the LT is p_{E-LT} which is obtained by starting with $p_{o\to i}^{0} = 1$ and iteratively applying update equations for the LT function for a fixed $p_{c\to b}^{\infty-}$:

$$p_{o\to i}^{l+1} = 1 - \omega\left(1 - L(p_{c\to b}^{\infty-})\tau\left(p_{o\to i}^{l}\right)\right) \tag{3.23}$$

until we reach, $p_{o\to i}^{\infty}$. Then the extrinsic at the output of the LT function is

$$p_{E-LT} = \tau(p_{o\to i}^{\infty}) = p_{A-LDPC} \tag{3.24}$$

Hence, the LT function shown in Fig.17 defines a mapping $p_{c\to b}^{\infty-}$ to p_{E-LT}/p_{A-LDPC}

$$p_{A-LDPC} = LT_{EXIT}(p_{c\to b}^{\infty-}) \tag{3.25}$$

3.5.2.3 LP Formulation

The best profile $\rho_{opt}(x)$ for a given bit edge degree profile $\lambda(x)$ and maximum degree of $\rho(x)$, $\rho_{deg-max}$ can be obtained as the solution to the linear programming problem

$$\min_{\rho} \sum_{i=0}^{\rho_{deg-max}} \frac{\rho_i}{(i+1)} \tag{3.26}$$

subject to:

1. Using L^{-1} function

$$x > \sum_i \rho_i (1-(1-I_{A-LDPC})\lambda(1-x))^{i-1}$$
$$\forall x \ \in [0,1-L^{-1}(1-I_{E-LDPC}(I_{A-LDPC}))]$$
$$\forall I_{A-LDPC} \in [0,1)$$

   or Using LT EXIT function

$$x > \sum_i \rho_i \left(1-(LT_{EXIT}(p_{c\to b}^{\infty-}))\lambda(1-x)\right)^{i-1}$$
$$\forall x \ \in [0, 1-p_{c\to b}^{\infty-}]$$
$$\forall p_{c\to b}^{\infty-} \in (0,1]$$

2. $\sum_i \rho_i = 1$
3. $0 \le \rho_i \le 1 \qquad \forall\, i \in 0,1,2,\dots\, \lambda_{max-deg}$

This objective function ensures that the resulting $\rho(x)$ maximizes the rate of LDPC code i.e., r_{LDPC}. The first constraint implies that the resulting LDPC code with $\rho(x)$, does not reach a fixed point below the LT curve for all values of the apriori to the

LDPC code. This constraint also implies that when the decoding process is started with the new apriori after initializing the messages along the edges to erasures, it does not have a fixed point anywhere below the LT curve and hence, produces a larger extrinsic. This constraint is usually enforced on a finely sampled region below the LT curve. The last two constraints ensure that the coefficients of $\rho(x)$ form a valid probability distribution. Fig.18 indicates the regions over which the constraints should be imposed for the optimization problem.

3.5.2.4 Choice of $\boldsymbol{\lambda(x)}$

Depending upon the complexity requirements, the $\lambda(x)$ can be chosen. If a bit edge degree profile with high average bit degree is chosen, then the resulting code might be complex. However, the degree 2 bit nodes should be carefully selected to satisfy the stability criterion of the LDPC code.

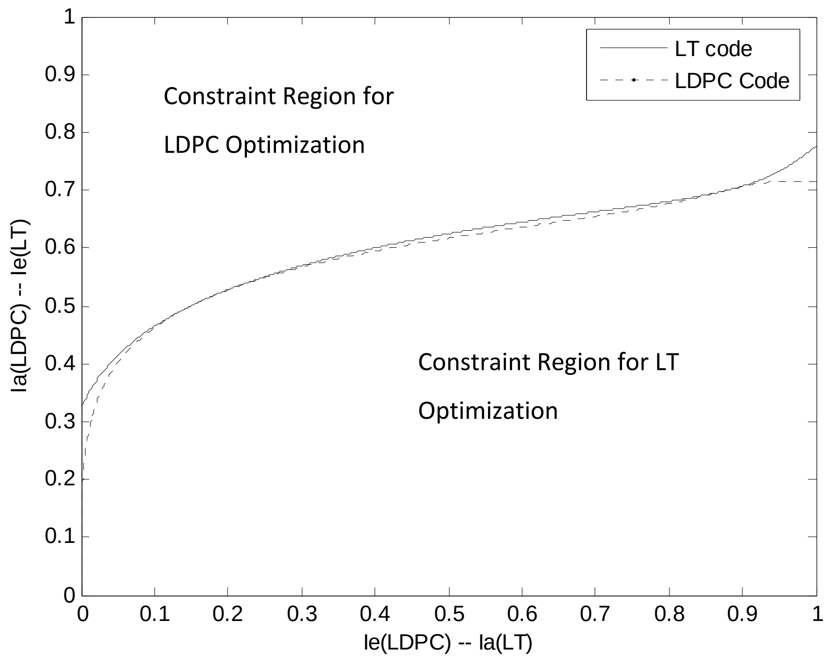

Fig.18 Regions of optimization

3.5.3 Recursive Optimization

Good profiles are designed by iteratively optimizing the LDPC and LT code profiles until they stop evolving under the given constraints. It appears as though optimizing just one of the component codes to suit the other should be sufficient. However, from our simulations we observed that it is always not true that we will end up finding suitable degree profiles $\omega(x)$ or $\rho(x)$ in just one iteration of the above optimization. One of the main reasons that we cannot achieve this in one step is that, we have a complexity constraint as well and a maximum degree constraint on both the LDPC and the LT code that impose some restrictions on the optimization problem. Sometimes, for an initial choice of α, we cannot find desirable degree profiles. Hence, we might have to change the value of α and repeat the optimization process.

3.6 Incremental Redundancy

Most practical communication systems have an existing implementation of a code that is usually memory efficient and computationally fast. More precisely, the amount of storage required at the decoder may impose a high cost on certain receivers and hence, LDPC code profiles for which memory efficient decoders [16] can be constructed are preferred. Also, in order to make the system more robust to channel variations, a LT code may be required that will use the existing outer code, without much modification.

This problem of designing incrementally redundant schemes given an inner code can be by and large solved in the erasure case with minimal overhead in the current framework. However, when we refer to overhead in this setting, it considers only the rate loss incurred by the outer LT code and not the rate loss due to the sub-optimality of the decoding algorithm of the existing inner code as mentioned in section 3.4.1. Hence, the overall overhead, ϵ_{incr} is given by,

$$\epsilon_{incr} = \frac{\alpha}{\Omega'(1)R_{LDPC-eff}} - 1 \tag{3.27}$$

where $R_{LDPC-eff}$ is the effective rate of the outer code. Moreover, from the area property, discussed in section 2.5.3

$$\begin{gathered} R_{LDPC-eff} \geq R_{LDPC} \\ \Rightarrow \ \epsilon_{incr} \leq \epsilon_{overall} \end{gathered} \tag{3.28}$$

For a given LDPC code we may not always be able to find a close match by using the procedure mentioned in section 3.5.1. However, if we are ready to trade off complexity, α with performance, we may obtain profiles for the LT codes that have very low overheads.

3.6.1 Average Overhead

For finite lengths, the overheads obtained during design may be insufficient to ensure a high probability of decoding success. We may have to increase the overhead incrementally until we are able to decode the message. Hence, we measure the performance of the rateless code based on the average overhead required to decode a code of particular length.

In practice, we can wait until the number of encoded output symbols received at the receiver equals the asymptotic requirements. Then the LT decoder and the LDPC decoder are iteratively run until they are unable to decode any further. At this point, more encoded bits are input and this is continued until all the bits are decoded. This is particularly useful for users with a bad channel, as they experience a significant delay in receiving packets.

3.7 Results

3.7.1 Design

We designed two profiles by using EXIT Charts and iteratively LP optimization on the inner and the outer codes. These profiles (A,B) are given below along with the profile (C) from [7] :

Profile A: Low complexity

$$\lambda(x) = x^2$$

$$\rho(x) = 0.4219x^4 + 0.2647\,x^5 + 0.0440x^{23} + 0.2694\,x^{50}$$

$$\omega(x) = 0.0001 + 0.8044x + 0.1102x^{40} + 0.0853x^{100}$$

$$\tau(x) = e^{1.5(x-1)}$$

$$Complexity, \Delta \propto 7.7230 \quad Overhead, \epsilon = 0.02636 \quad r_{LDPC} = 0.5932$$

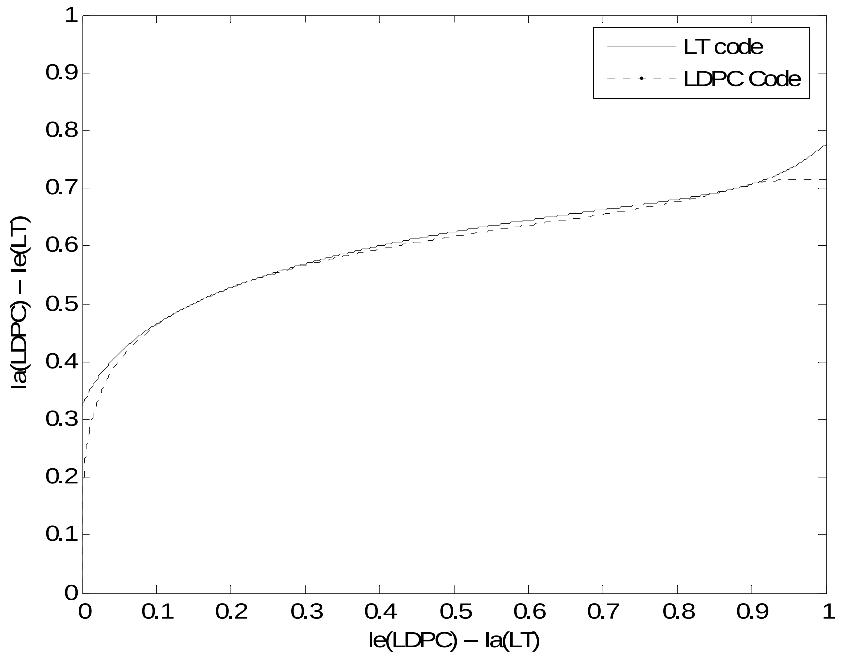

Fig.19 EXIT Chart for profile A

From the Fig.19 , we can see that the two curves are closely matched. The area between the two curves contribute to the overhead. Also, the LDPC code that we designed has an inherent rate loss due to iterative decoding.

Profile B: Low overhead

$$\lambda(x) = 0.4x^2 + 0.6x^3$$

$$\rho(x) = 0.1495x^3 + 0.2892x^4 + 0.0365x^7 + 0.2733x^{15} + 0.1581x^{100} + 0.0934x^{300}$$

$$\omega(x) = 0.001 + 0.8519x + 0.0735x^2 + 0.0234x^{10} + 0.0501x^{68}$$

$$\tau(x) = e^{1.29(x-1)}$$

$$Complexity, \Delta \propto 8.2953 \qquad Overhead, \epsilon = 0.0088 \qquad r_{LDPC} = 0.5810$$

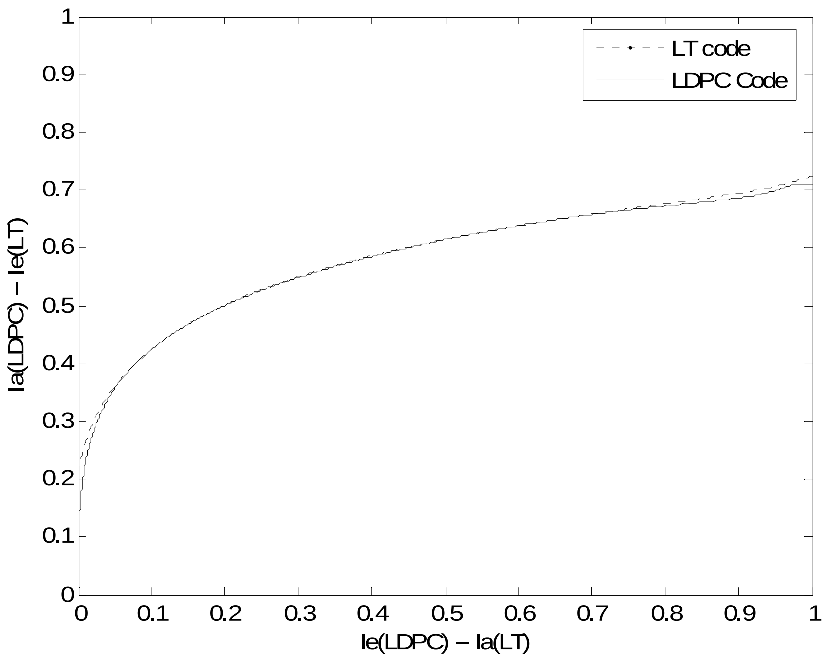

Fig.20 EXIT Chart for profile B

From Fig.20, we can see that the two EXIT curves are almost overlapping and hence, the overhead is negligible. However, this design is more complex than Profile A

Profile C: From [7]

$$\lambda(x) = 0.4x + 0.6x^2$$

$$\rho(x) = e^{136.612(x-1)}$$

$$\Omega(x) = 0.0080x + 0.4936x^2 + 0.1662x^3 + 0.0726x^4 + 0.0826x^5 + 0.0561x^8 + 0.0372x^9 + 0.0556x^{19} + 0.0250x^{65} + 0.0031x^{66}$$

$$\tau(x) = \mathrm{e}^{6.2067(\mathrm{x}-1)}$$

$$Complexity, \Delta \propto 9 \quad Overhead, \epsilon = 0.038 \quad r_{LDPC} = 0.9817$$

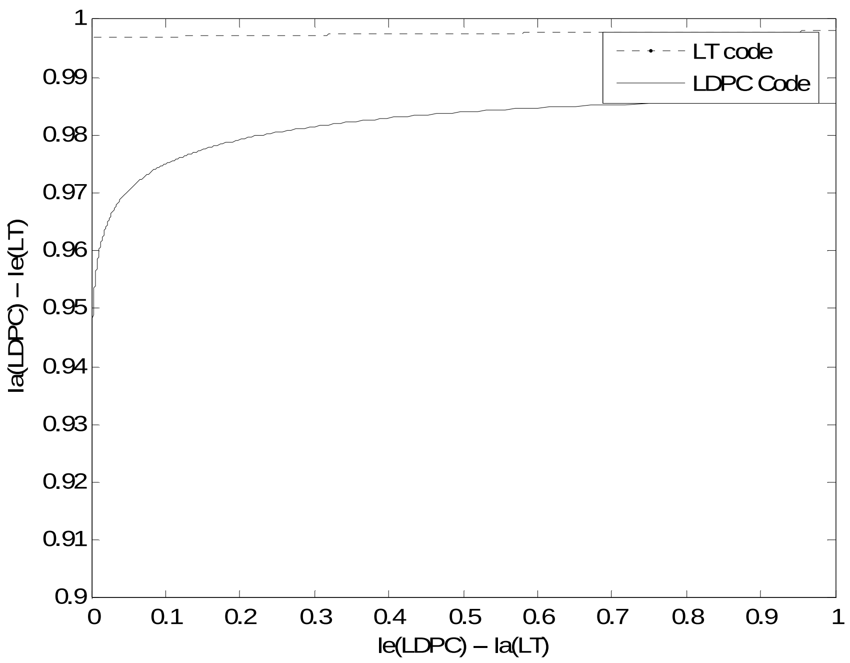

Fig.21 EXIT Chart for profile C

The EXIT Chart of Fig.21 clearly shows that the design does not take into account iterations because the curves are really flat and we gain not much by iterating.

A quick comparison shows that our profiles have low complexity and require low design overheads compared to the Profile C. However, it is important to note that we have not included the space requirements at the encoder in all of these cases. We have used a constant rate, ($r_{LDPC} = 0.6$) LDPC code in our designs which takes 40% more space.

3.7.2 Simulation

We simulated the Profile A and Profile C to see if they match with our theoretical results. We ran the encoding and decoding algorithm for 1000 frames of size 65536 using these degree distributions and we obtained the probability of success at various values of overhead. From Fig.22 and Fig.23 it is clear that Profile C has a significant improvement over Profile A in terms of performance and reliability.

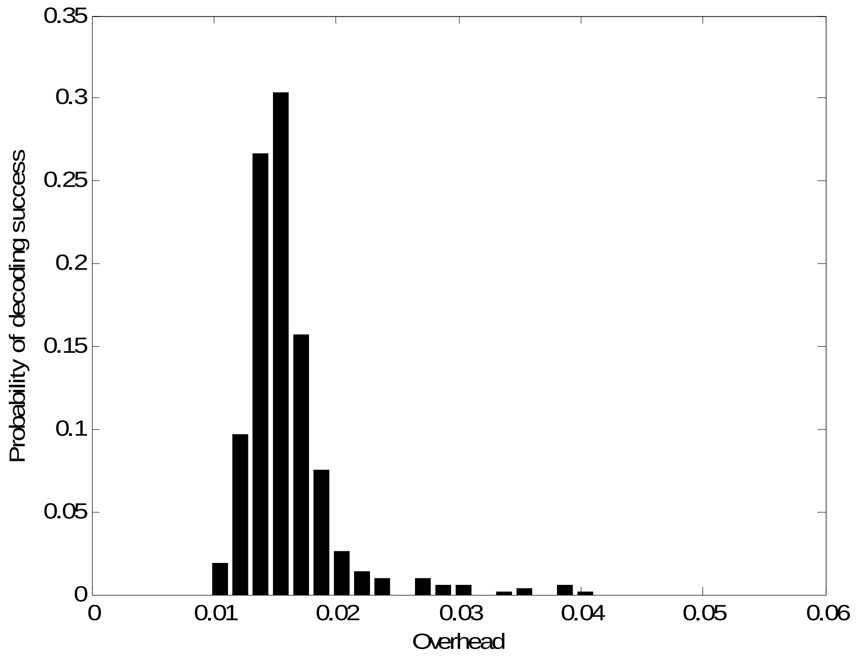

Fig.22 Probability density of decoding success for profile A

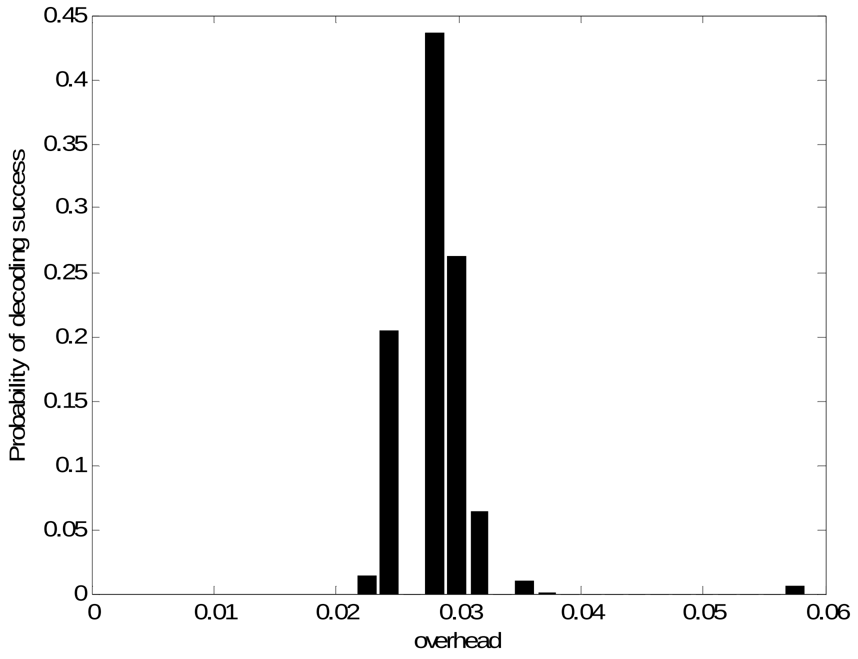

Fig.23 Probability density of decoding success for profile C

For the 1000 frames of length 65536 we observed that we could decode all the packets within a maximum overhead of 0.04 using a random realization of the code in Profile C. However, when we simulated a random realization of the Profile A, we observe that 4 out of the 1000 packets were not decoded even with overheads around 0.0575. We notice a slight deviation from the design overhead as we have not accounted for the finite length effects.

From these results, it is now clear that there are more codes than the ones provided in [7] that span these tradeoffs in the erasure case.

4 EXIT CHART BASED DESIGN AND ANALYSIS FOR THE GAUSSIAN CHANNEL

4.1 Introduction

In this section, we consider the analysis and design of Raptor-like codes under the iterative decoding scheme for a class of BIAWGN channels defined by the channel noise variance, σ^2 (or, equivalently, by the SNR). Our objective is to develop a design procedure for optimizing the degree profiles described by $\lambda(x), \rho(x)$ and $\omega(x)$ under the iterative decoder framework and design codes that are near optimal for a wide range of channels defined by the SNRs. We use Density Evolution, Gaussian Approximation and EXIT Charts to help us understand the existing codes and design profiles that can better trade-off complexity with overhead.

4.1.1 Notion of Universality

A code is said to be universal for the BIAWGN Channel if the overhead required to decode the code over a broad range of channels is small and approximately a constant irrespective of the SNR as explained in section 2.1.3. In this problem, we consider the design of codes that trade-off performance equally for a particular class of channels based on the noise level (SNR). This ensures that different users experiencing different noise levels but within the design range can enjoy the same performance. In other words, we are interested in designing rateless codes for a class of channels instead of designing them for a particular channel (SNR) as in [9], [13].

4.1.2 Decoding Structure

The decoder used here is the joint decoder in which messages are iteratively passed between the outer decoder and the inner decoder. However, in this case instead of passing bits and erasures between the component codes, we will be passing extrinsic LLRs. The decoding schedule does not affect the performance of the rateless code in the Gaussian case if the underlying graphs of the LT and LDPC codes are trees. The tree assumption is valid for large lengths and hence, we shall assume decoding schedules that simplifies our analysis and design procedures.

4.1.3 Complexity

The complexity of a rateless code is proportional to the number of edges of the underlying bipartite graph (Section 3.2.2) and the number of iterations performed. We can obtain the number of iterations to reach a desired probability of error using DE for a particular noise variance. However, to keep things simple we will assume that the complexity to be dominated by the number of edges. Though this may not be exact, we will still use this as a measure of complexity to design codes.

In practical communication systems, the memory required to store the intermediate messages (messages along the edges) is also an important parameter that determines the cost of decoding. In our case the number of edges, determine the number of intermediate messages and hence the memory required.

Since we are designing codes for a class of channels, the number of output symbols required to decode the information bits varies depending on the capacity of the channel. Hence, the complexity per information bit of the overall code varies with the noise variance and is given by,

$$\Delta(\sigma) \propto \left(\frac{L'(1)}{r_{LDPC}}\right) + \frac{\left(\Omega'(1) \times (1+\epsilon)\right)}{J\left(\frac{2}{\sigma^2}\right)} \tag{4.1}$$

4.2 Discretized Density Evolution

Given a $\rho(x)$, $\lambda(x)$, $\omega(x)$ and $\tau(x)$, we can analyze the component code decoders using DE as described in section 3.3. DDE follows directly from the decoding schedule. We shall assume that the $all\ zero\text{-}codeword$ was transmitted. The messages that are passed from one node to the other are clearly indicated in Fig.24 and the pdf of the corresponding messages are represented by replacing u by f.

To begin with, the channel input induces a distribution on the LLR of the output node to the input node message. This pdf propagates through the edges with iteration. The pdf along the edges are computed for the approximate (quantized) decoder by numerically performing the operations based on the type of the node. If the decoding schedule involves one iteration of MPA at the LT decoder followed by one at LDPC decoder, from section 2.4 the pdf update equations are

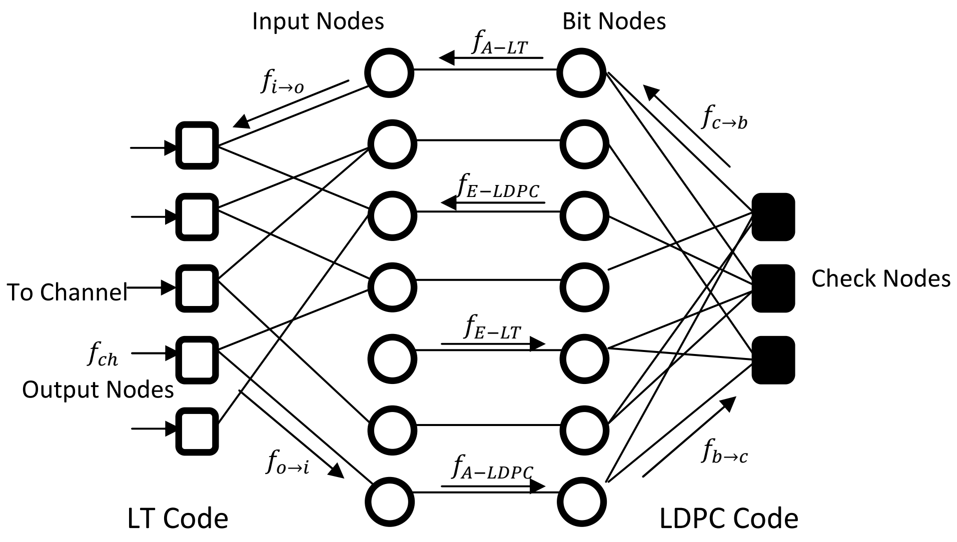

Fig.24 Bipartite graph showing densities

$$f_{i\to o}^{l} = f_{A-LT}^{l} \otimes \sum_{i} \tau_i (\otimes_{i-1} f_{o\to i}^{l-1}) \tag{4.2}$$

$$f_{o\to i}^{l} = \sum_{i} \omega_i \left(f_{ch} \boxplus (\boxplus_{i-1} f_{i\to o}^{l}) \right) \tag{4.3}$$

$$f_{E-LT}^{l} = \sum_{i} \tau_i (\otimes_i f_{o\to i}^{l}) \tag{4.4}$$

$$f_{A-LDPC}^{l} = f_{E-LT}^{l} \tag{4.5}$$

$$f^{l}_{b\to c} = f^{l}_{A-LDPC} \otimes \sum_{i} \lambda_i(\otimes_{i-1} f^{l-1}_{c\to b}) \tag{4.6}$$

$$f^{l}_{c\to b} = \sum_{i} \rho_i(\boxplus_{i-1} f^{l}_{b\to c}) \tag{4.7}$$

$$f^{l}_{E-LDPC} = \sum_{i} L_i(\otimes_{i} f^{l}_{c\to b}) \tag{4.8}$$

$$f^{l}_{A-LT} = f^{l}_{E-LDPC} \tag{4.9}$$

In the case of DDE these pdfs are tracked and we can conceptually think of DDE as passing the pdfs along the edges instead of the actual messages. Initially all the pdfs are initialized to an impulse around 0, while the output symbols receive channel LLRs. DDE proceeds in iterations and at each iteration, the pdfs along the edges are computed using the above update equations.

The number of bits used for quantization of the pdf determines the accuracy of the analysis of the unquantized DE. It has been shown in [2] that 10-bit quantized decoder produces results that are very close to that of an unquantized decoder.

4.2.1 Fixed Point Characterization

4.2.1.1 Fixed Point of the Component Code

DDE can be performed for an individual component code as explained in section 2.4 until the pdf stops evolving. Then the pdf is said to have reached a fixed point and this corresponds to the fixed point of the component code.

4.2.1.2 Fixed Point of the Overall Code

When DE is performed on existing profiles of LDPC and LT, the density of the LLRs evolves with iteration. Extrinsic information is iteratively passed between the component codes,

- Until we obtain complete information at the output of one of the component decoders which corresponds to zero probability of error.
- Or, until we reach a point where further iterations between the component codes, result in no additional information being generated. At this point, the decoding process is said to have reached the fixed point of the overall code. This also corresponds to the point where both the component coders have reached their individual fixed points.

4.3 Analysis Using Density Evolution

Analysis of a given LDPC and LT code profile is based on DDE. Since DDE very closely approximates actual DE, it provides a very good characterization of the performance of the given profiles for the range of channels (SNRs) under consideration. We do not make any assumptions or approximations on the distribution of the messages along the edges. Since the decoding schedule does not affect our analysis, we shall assume the following schedule to make our analysis simple:

- Perform one round of MPA within the LT decoder
- Pass the extrinsic LLRs from LT to the LDPC
- Perform one round of MPA within the LDPC decoder
- Pass the extrinsic LLRs from LDPC to the LT

These steps are repeated until the pdf stops evolving or until the pdf reaches a point mass at $+\infty$.

4.3.1 Overhead

After each bit (input) node and check (output) node update inside the LDPC (LT) decoder; the extrinsic pdf is computed and passed to the LT (LDPC) as the apriori pdf. This process is iterated until the pdfs stop evolving or until the probability of error goes to zero. Let the probability of error for a given $SNR = snr$ be denoted as $Pb(snr)$. The channel overhead required (refer to section 2.1.3) given a $SNR = snr$ is

$$\epsilon(snr) = \min_{\mathrm{Pb(snr)}\to 0} \epsilon \tag{4.10}$$

This is computed for all the channels of interest, $\mathcal{C}$, and the overhead of the code for the channels $\in \mathcal{C}$ is

$$\epsilon_{\mathcal{C}} = \max_{\mathrm{snr}\in\mathcal{C}} \epsilon(snr) \tag{4.11}$$

We say that a code is near universal for the class of channels $\mathcal{C}$, if the required overhead is minimal and almost equal for all SNRs $\in \mathcal{C}$.

4.4 Gaussian Approximation

Under this approximation as discussed in section 2.4.2, the density of the messages passed from one node to the other indicated in Fig.24 are approximated by a Gaussian pdf [1] and the mean of these pdfs are represented by replacing u by μ. It is important to note that $\tau(x)$ represents a Poisson distribution and can be expressed using the Taylor's series with all the coefficients being positive.

GA will be used in design and hence, from this perspective, we use a different decoding schedule that simplifies our design, where we perform MPA within a component code until the densities stop evolving.

4.5 Design Using EXIT Charts

Our objective is to design LT and LDPC profiles such that we can be simultaneously near optimal for a range of SNR. In this section, we will assume a decoding schedule different from the one discussed in section 4.3 and is given as follows:

- Perform DE within LT code until fixed point of the LT code is reached
- Pass extrinsic information to the LDPC Code
- Perform DE within LDPC code until the fixed point of the LDPC code is reached
- Pass extrinsic information to the LT Code

Under this decoding schedule, we cannot decode all the information bits even if one of the component decoders does not produce enough extrinsic information. Since, the MPA is performed on the component code until a fixed point is reached, for the same component code to produce more extrinsic information at the output of the next iteration, the apriori information should have increased. In other words, the other component decoder should be able to produce sufficient extrinsic information to sustain the decoding process. To be more precise, $I_{E-LDPC}^{l+1} > I_{E-LDPC}^{l}$ and $I_{E-LT}^{l+1} > I_{E-LT}^{l}$ until $I_{E-LDPC} = 1$ or $I_{E-LT} = 1$. These constraints on an EXIT chart indicate that the LDPC curve should lie strictly about LT curve and vice versa. The EXIT curves are obtained using the decoding schedule discussed here i.e., DDE is performed at each component decoder for a given apriori, I_A until we reach the fixed point and then the resulting extrinsic information, I_E is computed. We plot the I_{A-LT} vs. $I_{E-LT}(snr)$ $\forall\ snr \in \mathcal{C}$ and I_{E-LDPC} vs. I_{A-LDPC} as shown in Fig.25.

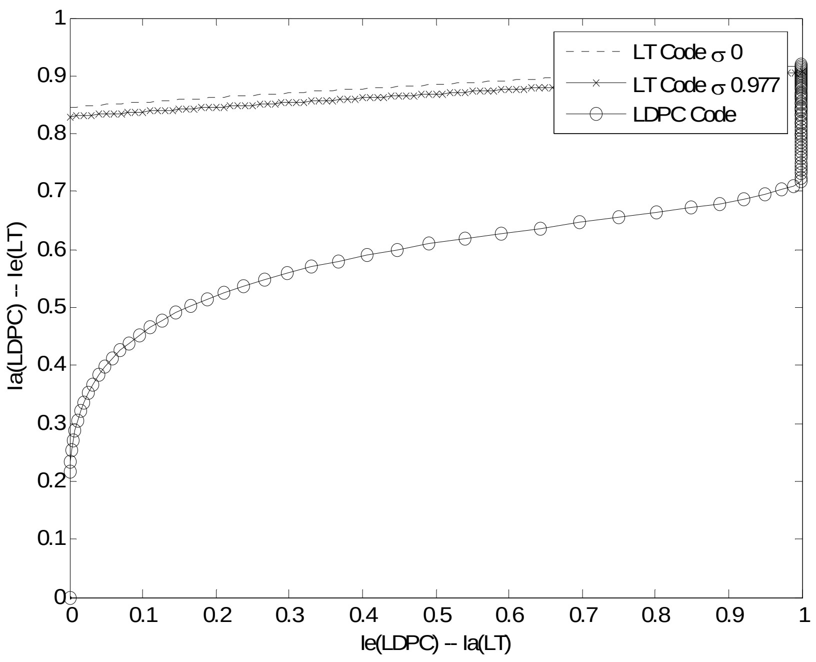

Fig.25 EXIT Chart for various SNRs

Unlike the erasure channel scenario, where the EXIT charts are accurate, in the BIAWGN Channel, we make assumptions on the densities when obtaining the EXIT charts and hence, do not necessarily reflect the actual performance. In order to obtain good designs that do not affect the analysis and the actual performance dramatically, we need to make as fewer assumptions on the densities as possible. As mentioned earlier, DDE is a suitable alternative to actual DE and we shall employ DDE wherever possible during our design phase.

4.5.1 Optimization of the LT Code for a Given LDPC Code

Let us say that we are given a $\lambda(x)$ and $\rho(x)$, then we can obtain the bit node profile, $L(x)$ using the relation between edge and node degree profiles. Our objective is to find the degree profiles $\omega(x)$ of the LT code for a fixed complexity α that closely match the LDPC code for a range of SNR $\in \mathcal{C}$. Hence the design problem is a joint optimization problem, in the sense that we have to optimize our LT code profile for all the SNRs of interest.

When the LDPC profiles are known, then the EXIT curve can be obtained under the GA of the apriori information. Since the EXIT curve is a plot of I_A vs. I_E, it does not imply any underlying distribution on the apriori or extrinsic LLRs. Hence, to obtain the extrinsic LLR, we assume that the apriori distribution is a Gaussian with the mean chosen suitably such that $J(\mu_A) = I_A$. Given $\rho(x)$ and $\lambda(x)$ we can perform DDE for an input I_{A-LDPC} to obtain the extrinsic information, $I_{E-LDPC} = I_{A-LT}$ and thus plot the EXIT curve.

For the rateless code to be successfully decoded, we need to ensure that at every point along the LDPC curve defined by (I_{A-LDPC}, I_{E-LDPC}), a desired LT profile should be able to provide more extrinsic information i.e., $I_{E-LT}(I_{A-LDPC}, \sigma) > I_{A-LDPC}$ for all $\sigma \in \mathcal{C}$. This condition ensures that the decoding continues until all the bits are decoded i.e., $I_{E-LDPC} = 1$ or $I_{E-LT} = 1$.

4.5.1.1 EXIT Function for the LDPC

This function implements DDE on the given LDPC profiles, $\lambda(x)$ and $\rho(x)$ for a given apriori density, f_{A-LDPC} along the bit node. When the mutual information of the output node to input node message, $I_{o\to i}^{\infty-}$ of the LT code is given instead, we can obtain

$$f_{A-LDPC} = \sum_i \tau_i \left(\otimes_i \mathcal{N}\left(J^{-1}(I_{o\to i}^{\infty-})\right)\right) \tag{4.12}$$

The density evolution equations within the LDPC are:

$$f_{b\to c}^{l} = f_{A-LDPC} \otimes \sum_i \lambda_i (\otimes_{i-1} f_{c\to b}^{l}) \tag{4.13}$$

$$f_{c\to b}^{l+1} = \sum_i \rho_i (\boxplus_{\,i-1} f_{b\to c}^{l}) \tag{4.14}$$

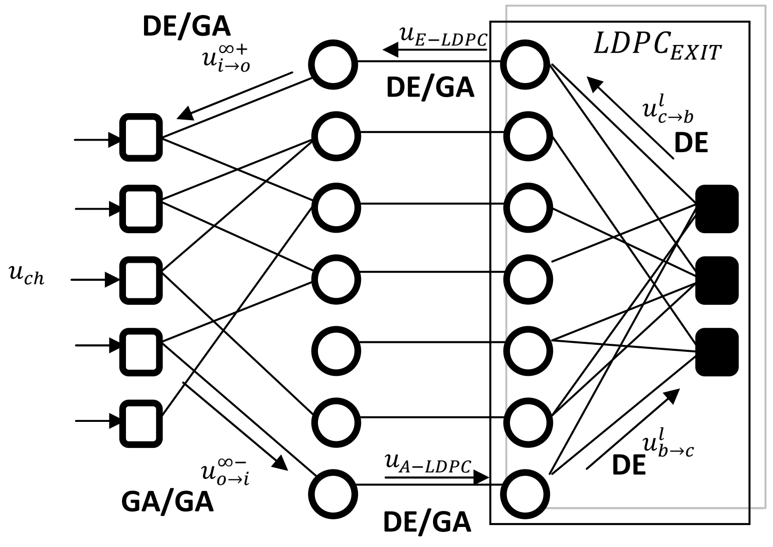

Fig.26 LDPC EXIT function (Gaussian)

Density evolution is performed starting with densities $f_{c\to b}^{0}$ and $f_{b\to c}^{0}$ being initialized to an impulse at 0 until the densities reach the fixed point, $f_{c\to b}^{\infty}$. At this point, the extrinsic density and information at the output of the LDPC block are computed as

$$f_{E-LDPC} = \sum_i L_i (\otimes_i f_{c\to b}^{\infty}) = f_{A-LT} \tag{4.15}$$

$$I_{A-LT} = 1 - \int_{-\infty}^{+\infty} \log(1 + e^{-L})\, f_{E-LDPC}(L) dL \qquad (4.16)$$

Hence, the LDPC function shown in Fig.26 defines a mapping $I_{o\to i}^{\infty-}$ to f_{E-LDPC}/f_{A-LT}

$$f_{A-LT} = LDPC_{EXIT}(I_{o\to i}^{\infty-}) \qquad (4.17)$$

4.5.1.2 LP Setup Using GA

A given apriori information on the LDPC decoder, I_{A-LDPC} corresponds to an output node to input node message fixed point information rate in the previous iteration of the LT decoder, $I_{o\to i}^{\infty-}$ and the relation is given by,

$$I_{A-LDPC} = \sum_i \tau_i J(iJ^{-1}(I_{o\to i}^{\infty-})) \qquad (4.18)$$

Using GA of the densities within the LT, the information update equations are:

$$\mu_{i\to o}^{l} = J^{-1}(I_{A-LT}) + J^{-1}\left(\sum_i \tau_i J\left((i-1)J^{-1}(I_{o\to i}^{l})\right)\right) \qquad (4.19)$$

$$I_{o\to i}^{l+1} = \sum_i \omega_i I\left(\mathcal{N}(\mu_{ch}) \boxplus \left(\boxplus_{i-1} \mathcal{N}(\mu_{i\to o}^{l})\right)\right) \qquad (4.20)$$

$$I_{E-LT} = \sum_i \tau_i J(iJ^{-1}(I_{o\to i}^{\infty+})) \qquad (4.21)$$

Since we would like to design codes that work with a range of SNRs, we have to enforce the constraints for each SNR ∈ $\mathcal{C}$ which corresponds to a channel LLR density mean, μ_{ch}. It is clear that the density evolution equations are linear in the coefficient of $\omega(x)$ given $I_{o\to i}$ and I_{A-LT}. However, they cannot be expressed as a linear function of $\omega_i{}'s$ for a given $I_{A-LT} = I_{E-LDPC}$ and I_{A-LDPC}. So in practice, we need the information transferred from the output node to the input node at the fixed point density given by, $I_{o\to i}^{\infty-}$ corresponding to I_{A-LDPC} to formulate the design problem as a LP.

Since, we fix α for the optimization problem and $\tau(x)$ is known, the density of the output node to the input node message of the LT decoder can be approximated by a Gaussian with an appropriate mean using a τ^{-1} function.

The map between I_{A-LDPC} and $I_{o\to i}^{\infty-}$ is one-to-one, we can obtain an inverse function τ^{-1} which returns $I_{o\to i}^{\infty-}$ for a given I_{A-LDPC}. This can be implemented using a look-up table given $\tau(x)$.

4.5.1.3 LP Setup Using DDE

To get good designs, we need to avoid approximations and so we employ DDE wherever possible. Within the LT decoder we obtain the following information update equations,

$$f_{A-LDPC} = \sum_i \tau_i(\otimes_i f_{o\to i}^{\infty-}) \tag{4.22}$$

$$f_{i\to o}^{l} = f_{A-LT} \otimes \sum_i \tau_i(\otimes_{i-1} f_{o\to i}^{l}) \tag{4.23}$$

$$f_{o\to i}^{l+1} = \sum_i \omega_i\left(\mathcal{N}(\mu_{ch}) \boxplus \left(\boxplus_{i-1} f_{i\to o}^{l}\right)\right) \tag{4.24}$$

$$f_{E-LT} = \sum_i \tau_i(\otimes_i f_{o\to i}^{\infty+}) \tag{4.25}$$

It is clear that the density evolution equations are linear in the coefficients of $\omega(x)$ given $f_{o\to i}$ and f_{A-LT}. However, it is impossible to enforce this constraint for all possible pdfs $f_{o\to i}$ and f_{A-LT}. Hence, we relax the design problem by making a GA on $f_{o\to i}^{\infty+}$ and $f_{o\to i}^{\infty-}$ to denote the densities by the mean, $\mu_{o\to i}^{\infty+} = J^{-1}\left(I(f_{o\to i}^{\infty+})\right)$ and $\mu_{o\to i}^{\infty-} = J^{-1}\left(I(f_{o\to i}^{\infty-})\right)$ respectively. We start with a Gaussian pdf of mean, $\mu_{o\to i}^{\infty-}$ and obtain f_{A-LDPC} as given above. Then we use the LDPC EXIT function described in section 4.5.1.1 to obtain f_{E-LDPC}. With the apriori density of LT, $f_{A-LT} = f_{E-LDPC}$, we obtain the information $I_{o\to i}^{l+1}$ using

$$\begin{aligned} I_{o\to i}^{l+1} &= \sum_i \omega_i I\left(f_{ch} \boxplus \left(\boxplus_{i-1} f_{i\to o}^{l}\right)\right) \\ &= \sum_i \omega_i I\left(f_{ch} \boxplus \left(\boxplus_{i-1}\left(f_{A-LT} \otimes \sum_i \tau_i (\otimes_{i-1} f_{o\to i}^{l})\right)\right)\right) \end{aligned} \tag{4.26}$$

where, $f_{o\to i}^{l} = \mathcal{N}(J^{-1}(I_{o\to i}^{l}))$ and $f_{ch} = \mathcal{N}(\mu_{ch})$

4.5.1.4 LP Formulation

The best profile $\omega_{opt}(x)$ for a given complexity, α and maximum degree of $\omega(x)$, $\omega_{deg-max}$ can be obtained as the solution to the linear programming problem

$$\min_{\omega} \sum_{i=0}^{\omega_{deg-max}} \frac{\omega_i}{(i+1)} \tag{4.27}$$

subject to:

1. Using the τ^{-1} function

$$x > \sum_i \omega_i I\left\{\mathcal{N}\left(\frac{2}{\sigma^2}\right) \boxplus \left[\boxplus_{i-1} \mathcal{N}\left(J^{-1}(I_{A-LT}) + J^{-1}\left(\sum_i \tau_i J\left((i-1)J^{-1}(x)\right)\right)\right)\right]\right\}$$

$$\forall x \in [0, \tau^{-1}(I_{A-LDPC}(I_{A-LT}))]$$
$$\forall I_{A-LT} \in (0,1)$$
$$\forall \sigma \in \mathcal{C}$$

or using the LDPC EXIT function

$$x > \sum_i \omega_i I\left\{\mathcal{N}\left(\frac{2}{\sigma^2}\right) \boxplus \left[\boxplus_{i-1}\left(LDPC_{EXIT}(I_{o\to i}^{\infty-}) \otimes \sum_i \tau_i \left(\otimes_{i-1} \mathcal{N}\left(J^{-1}(x)\right)\right)\right)\right]\right\}$$

$$\forall x \in [0, I_{o\to i}^{\infty-}]$$
$$\forall I_{o\to i}^{\infty-} \in (0,1)$$
$$\forall \sigma \in \mathcal{C}$$

2. $\sum_i \omega_i = 1$
3. $0 \le \omega_i \le 1 \qquad \forall\, i \in 0, 1, 2, \ldots\, \omega_{max-deg}$

This objective function ensures that the resulting $\omega(x)$ has the least overhead requirements i.e., minimizes r_{LT}. The first constraint implies that the resulting LT process with $\omega(x)$, does not reach a fixed point below the LDPC curve for all values of the apriori to the LT decoder and for the class of channels under consideration. This constraint is usually enforced on a finely sampled region below the LDPC curve. The last two constraints ensure that the coefficients of $\omega(x)$ form a valid probability distribution.

4.5.2 Optimization of the LDPC Code for a Given LT Code

Let us say that we are given a $\omega(x)$ and $\tau(x)$. Our objective is to find the degree profiles $\rho(x)$ of the LDPC code for a fixed $\lambda(x)$ that closely match the LT code for the class of channels, $\mathcal{C}$ under consideration. $\lambda(x)$ and r_{LDPC} determine the complexity of the resulting rateless code partially.

When the LT profiles are known, then the EXIT curve can be obtained under the Gaussian approximation of the apriori information i.e., since the EXIT curve is a plot of I_A vs. I_E, it does not imply any underlying distribution on the apriori or extrinsic LLR. Hence, to obtain the extrinsic LLR, we assume that the apriori distribution is a Gaussian with the mean chosen suitably such that $J(\mu_A) = I_A$. Given a $\omega(x)$ and $\tau(x)$ we can perform DDE for an input I_{A-LT} and obtain the extrinsic information, $I_{E-LT} = I_{A-LDPC}$ and thus plot the EXIT curve.

For the rateless code to be successfully decoded, we need to ensure that at every point along the LT curve defined by (I_{A-LT}, I_{E-LT}), a desired LDPC profile should be able to provide more extrinsic information i.e., $I_{E-LDPC}(I_{A-LT}) > I_{A-LT}$ for all SNRs $\in \mathcal{C}$. This condition ensures that the decoding continues until all the bits are decoded i.e., $I_{E-LDPC} = 1$ or $I_{E-LT} = 1$.

4.5.2.1 EXIT Function for the LT

This function implements DDE on the given LT profiles, $\tau(x)$ and $\omega(x)$ for a given apriori density, f_{A-LT} along the bit node. When the mutual information along the check node to the bit node, $I_{c\to b}^{\infty-}$ of the LDPC code is given, we can obtain

$$f_{A-LT} = \sum_i L_i\left(\otimes_i \mathcal{N}\left(J^{-1}(I_{c\to b}^{\infty-})\right)\right) \tag{4.28}$$

The density evolution equations within the LT are:

$$f_{i\to o}^{l} = f_{A-LT} \otimes \sum_i \tau_i(\otimes_{i-1} f_{o\to i}^{l}) \tag{4.29}$$

$$f_{o\to i}^{l+1} = \sum_i \omega_i\left(\mathcal{N}\left(\frac{2}{\sigma^2}\right) \boxplus (\boxplus_{\,i-1} f_{i\to o}^{l})\right) \tag{4.30}$$

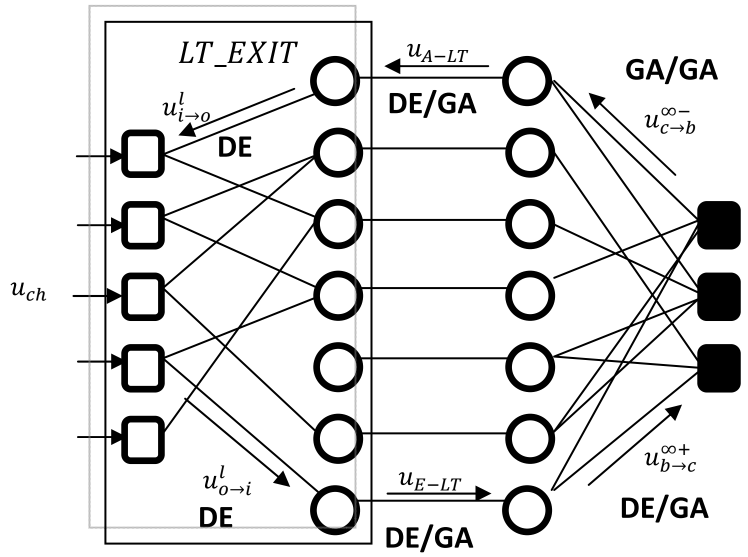

Fig.27 LT EXIT function (Gaussian)

Density evolution is performed starting with densities $f_{o\to i}^{0}$ and $f_{i\to o}^{0}$ being initialized to an impulse at 0 until the densities reach the fixed point, $f_{o\to i}^{\infty}$. At this point, the extrinsic density and information at the output of the LT block is computed as

$$f_{E-LT} = \sum_i \tau_i(\otimes_i f_{o\to i}^{\infty}) \; = \; f_{A-LDPC} \tag{4.31}$$

$$I_{A-LDPC}(\sigma) = 1 - \int_{-\infty}^{+\infty} \log(1 + e^{-L})\, f_{E-LT}\big(L(\sigma)\big) dL \tag{4.32}$$

Hence, the LT function shown in Fig.27 defines a mapping $(I_{c\to b}^{\infty-}, \sigma)$ to f_{E-LT}/f_{A-LDPC}

$$f_{A-LDPC} = LT_{EXIT}(I_{c\to b}^{\infty-}, \sigma) \tag{4.33}$$

Since we would like to design codes that work with a range of channels, we have to enforce the constraints for each of these channels with different channel LLR means, $\sigma \in \mathcal{C}$.

4.5.2.2 LP Setup Using GA

A given apriori Information on the LT decoder, I_{A-LT} corresponds to a check node to bit node message fixed point information rate in the previous iteration of the LDPC decoder, $I_{c\to b}^{\infty-}$ and the relation is given by,

$$I_{A-LT} = \sum_i L_i J(iJ^{-1}(I_{c\to b}^{\infty-})) \tag{4.34}$$

Using GA of the densities within the LDPC, the information update equations are:

$$\mu_{b\to c}^{l} = J^{-1}(I_{A-LDPC}) + J^{-1}\left(\sum_i \lambda_i J\left((i-1)J^{-1}(I_{c\to b}^{l})\right)\right) \tag{4.35}$$

$$I_{c\to b}^{l+1} = \sum_i \rho_i I\left(\boxplus_{i-1} \mathcal{N}\left(\mu_{b\to c}^{l}\right)\right) \tag{4.36}$$

$$I_{E-LDPC} = \sum_i L_i J(iJ^{-1}(I_{c\to b}^{\infty+})) \tag{4.37}$$

It is clear that the density evolution equations are linear in the coefficient of $\rho(x)$ given $I_{c\to b}$ and I_{A-LDPC}. However, they cannot be expressed as a linear function of ρ_i's for a given $I_{A-LDPC} = I_{E-LT}$ and I_{A-LT}. So in practice, we need the information transferred from the check node to bit node at the fixed point density given by, $I_{c\to b}^{\infty-}$ corresponding to I_{A-LT} to formulate the design problem as a LP.

Since, we fix $\lambda(x)$ and $L(x)$ for the optimization problem, the density of the check node to the bit node message of the LDPC decoder can be approximated by a Gaussian with an appropriate mean using a L^{-1} function.

The map between I_{A-LT} and $I_{c\to b}^{\infty-}$ is one-to-one for a given noise variance σ^2, we can obtain an inverse function L^{-1} which returns $I_{c\to b}^{\infty-}$ for a given I_{A-LT} and σ. This can be implemented by using a look-up table given $L(x)$.

4.5.2.3 LP Setup Using DDE

To get good designs, we need to avoid approximations and so we employ DDE wherever possible. Within the LDPC decoder we obtain the following information update equations,

$$f_{A-LT} = \sum_i L_i(\otimes_i f_{c\to b}^{\infty-}) \tag{4.38}$$

$$f_{b\to c}^{l} = f_{A-LDPC} \otimes \sum_i \lambda_i(\otimes_{i-1} f_{c\to b}^{l-1}) \tag{4.39}$$

$$f_{c\to b}^{l+1} = \sum_i \rho_i(\boxplus_{i-1} f_{b\to c}^{l}) \tag{4.40}$$

$$f_{E-LDPC} = \sum_i L_i(\otimes_i f_{c\to b}^{\infty+}) \tag{4.41}$$

It is clear that the density evolution equations are linear in the coefficients of $\rho(x)$ given $f_{c\to b}$ and f_{A-LDPC}. However, it is impossible to enforce this constraint for all possible pdfs $f_{c\to b}$ and f_{A-LDPC}. Hence we relax the design problem, by making a GA on $f_{c\to b}^{\infty+}$ and $f_{c\to b}^{\infty-}$ to denote the densities by the mean, $\mu_{c\to b}^{\infty+} = J^{-1}\big(I(f_{c\to b}^{\infty+})\big)$ and $\mu_{c\to b}^{\infty-} = J^{-1}\big(I(f_{c\to b}^{\infty-})\big)$. Hence, we start with a Gaussian pdf of mean, $\mu_{c\to b}^{\infty-}$ and obtain f_{A-LT} as given above. Then we use the LT EXIT function described in section 4.5.2.1 to obtain f_{E-LT} for all $\sigma \in \mathcal{C}$. With the apriori density of LDPC, $f_{A-LDPC} = f_{E-LT}$, we obtain the information $I_{c\to b}^{l+1}$ as

$$\begin{aligned} I_{c\to b}^{l+1} &= \sum_i \rho_i I\big(\boxplus_{i-1} f_{b\to c}^{l}\big) \\ &= \sum_i \rho_i I\left(\boxplus_{i-1}\left(f_{A-LDPC} \otimes \sum_i \lambda_i (\otimes_{i-1} f_{c\to b}^{l})\right)\right) \end{aligned} \tag{4.42}$$

where, $f_{c\to b}^{l} = \mathcal{N}(J^{-1}(I_{c\to b}^{l}))$

4.5.2.4 LP Formulation

The best profile $\rho_{opt}(x)$ for a given complexity, $\lambda(x)$ and maximum degree of $\rho(x)$, $\rho_{deg-max}$ can be obtained as the solution to the linear programming problem

$$\min_{\rho} \sum_{i=0}^{\rho_{deg-max}} \frac{\rho_i}{(i+1)} \tag{4.43}$$

subject to:

1. Using the L^{-1} function

$$x > \sum_i \rho_i I\left[\boxplus_{i-1} \mathcal{N}\left(J^{-1}\big(I_{A-LDPC}(\sigma)\big) + J^{-1}\left(\sum_i \lambda_i J\big((i-1)J^{-1}(x)\big)\right)\right)\right]$$

$$\forall x \in \left[0, L^{-1}\big(I_{A-LT}(I_{A-LDPC}(\sigma))\big)\right]$$
$$\forall I_{A-LDPC}(\sigma) \in (0,1)$$
$$\forall \sigma \in \mathcal{C}$$

or using LT EXIT function

$$x > \sum_i \rho_i I\left[\boxplus_{i-1}\left(LT_{EXIT}(I_{c\to b}^{\infty-}, \sigma) \otimes \sum_i \lambda_i \left(\otimes_{i-1} \mathcal{N}\big(J^{-1}(x)\big)\right)\right)\right]$$

$$\forall x \in [0, I_{c\to b}^{\infty-}]$$
$$\forall I_{c\to b}^{\infty-} \in (0,1)$$
$$\forall \sigma \in \mathcal{C}$$

2. $\sum_i \rho_i = 1$

3. $0 \leq \rho_i \leq 1 \qquad \forall\, i \in 0, 1, 2, \ldots\ \rho_{max-deg}$

This objective function ensures that the resulting $\rho(x)$ maximizes the rate of the LDPC i.e., maximizes r_{LDPC}. The first constraint implies that the resulting LDPC code with $\rho(x)$, does not reach a fixed point below the LT curves corresponding to all the SNRs of interest and for all values of the apriori to the LDPC decoder. This constraint is usually enforced on a finely sampled region below the LT curve. The last two constraints ensure that the coefficients of $\rho(x)$ form a valid probability distribution.

4.6 Results

4.6.1 Design

We designed a rateless code, Profile A i.e., LDPC and LT profiles using LP Optimizations with the Gaussian approximation described in sections 4.5.1.2 and 4.5.2.2. Profile B is the one profile of the Raptor code proposed by Etesami and Shokrollahi.

Profile A: Low complexity

$$\lambda(x) = 0.3x^1 + 0.55\mathrm{x}^2 + 0.15x^3$$

$$\rho(x) = 0.2066x^4 + 0.6572\,x^5 + 0.001x^{41} + 0.1352\,x^{42}$$

$$\omega(x) = 0.0001 + 0.8044x + 0.1102x^{40} + 0.0853x^{100}$$

$$\tau(x) = e^{1.5(x-1)}$$

$$r_{LDPC} = 0.5847$$

$$Design\ Overhead, \epsilon_{design} = 0.042 \qquad Actual\ Overhead, \epsilon_{DE} = 0.062$$

$$\sigma_{design} = [0{,}0.977]$$

Profile B: From [9],

$$\Omega(x) = 0.0060x^1 + 0.4920x^2 + 0.0339x^3 + 0.2403x^4 + 0.0060x^5 + 0.0950x^8 + 0.0490x^{14} + 0.0180x^{30} + 0.0356x^{33} + 0.0330x^{200}$$

$$\tau(x) = e^{12.6128(x-1)}$$

$$Overhead, \epsilon_{DE} = 0.065$$

Since the Profile A was designed using GA, the design overhead ϵ_{design} is different from the actual overhead obtained using DDE. The actual overhead was obtained as the worst case overhead for the range of BIAWGN Channels defined by channel noise variance from 0 to 0.9545 (0.9772).

From the complexity comparison table, we can clearly see that Profile A is less complex than Profile B for the entire range of SNRs under consideration. Moreover, the overhead in both cases are almost same. However, it is important to note that, the complexity shown in the table for Profile B does not include the complexity of the high rate LDPC code.

TABLE 1. Complexity comparison between existing & Proposed Profile

Noise Variance, σ	Number of edges per bit Profile A	Number of edges per bit Profile B
0	7.3809	12.6128
0.3	7.3859	12.6364
0.5	7.6453	13.8185
0.7	8.4145	17.3227
0.8	8.9571	19.7497
0.977	10.1371	25.1703

From the EXIT curves shown in Fig.28, Fig.29, Fig.30, Fig.31 and Fig.32, it is clear that iterative decoding of the rateless codes constructed using Profile A will converge for all the SNRs within the design range. Hence, the designed rateless code is near universal for the BIAWGN Channel for the entire range of SNRs.

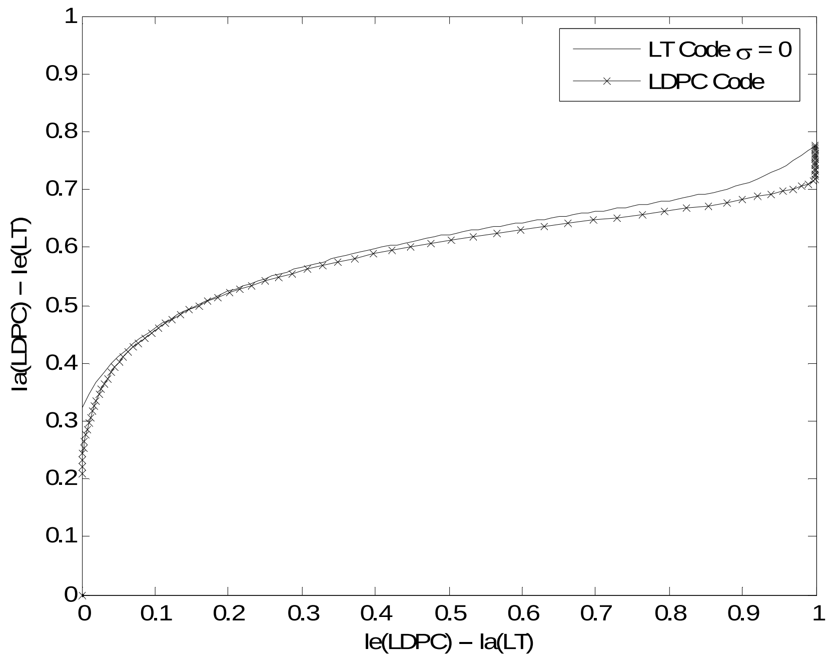

Fig.28 LT & LDPC EXIT curves for $\sigma = 0$

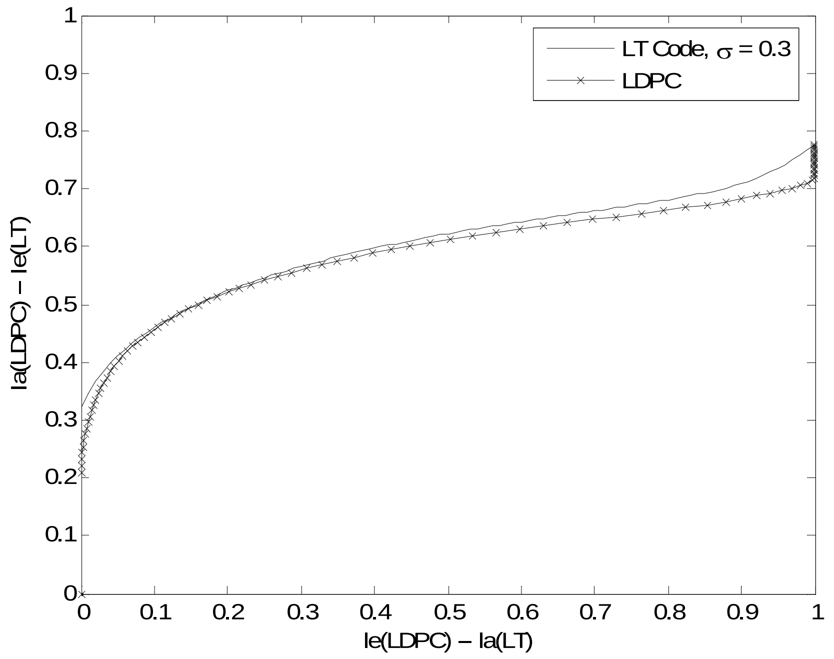

Fig.29 LT & LDPC EXIT curves for $\sigma = 0.3$

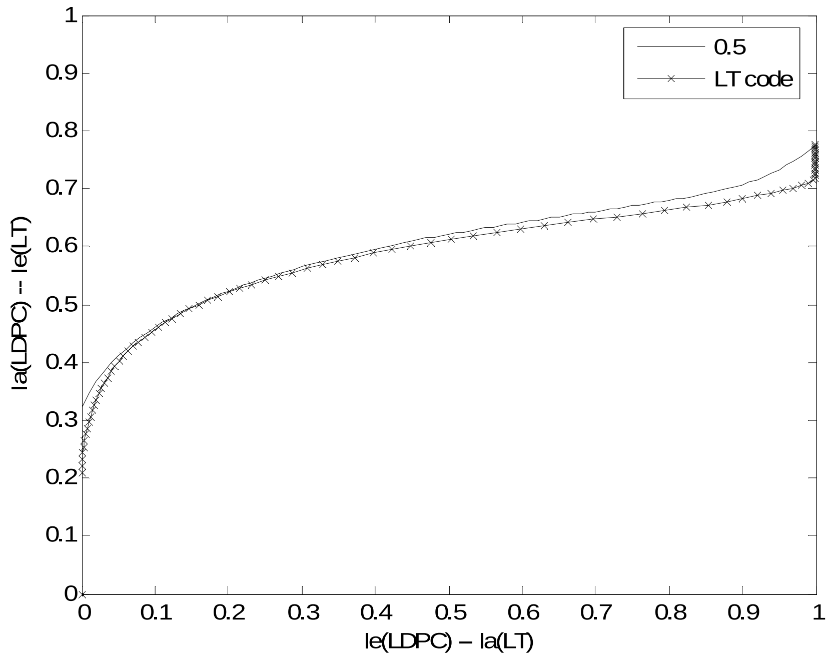

Fig.30 LT & LDPC EXIT curves for $\sigma = 0.5$

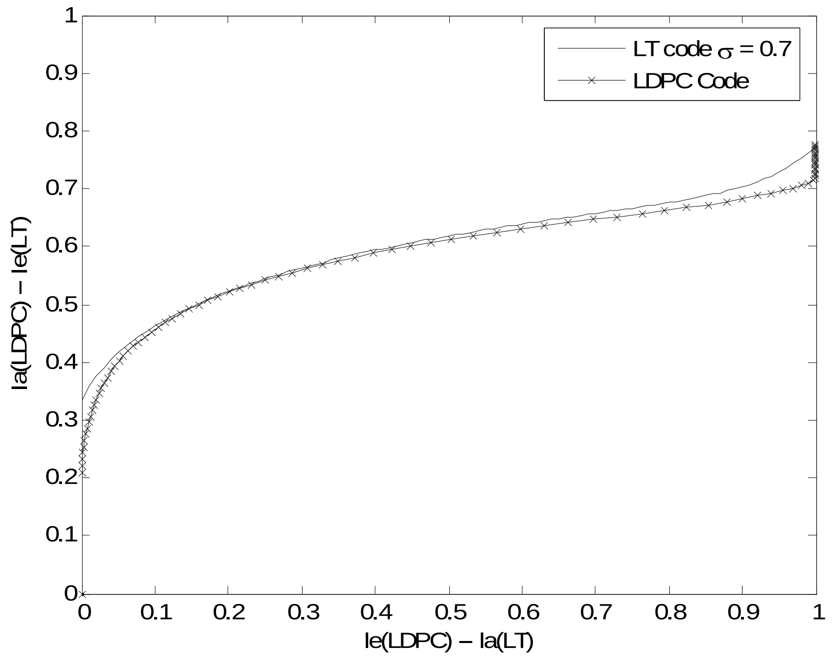

Fig.31 LT & LDPC EXIT curves for $\sigma = 0.7$

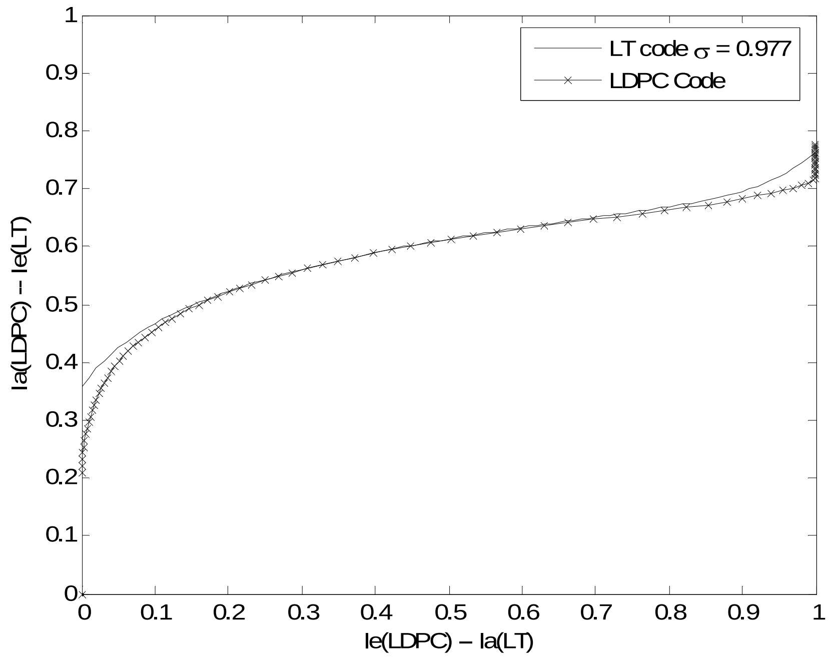

Fig.32 LT & LDPC EXIT curves for $\sigma = 0.977$

4.6.2 Simulation

We simulated the above profiles for a block length of 65536 for various overheads for a maximum of 250 iterations per frame. The simulation results for the designed profile, Profile A is as shown in Fig.33. From the simulation and design results, asymptotically in length, Profile A is a better option to Profile B in terms of complexity as well as overhead for the range of SNRs under consideration. Moreover, this proves the existence of a larger class of codes using this framework of iterative decoding.

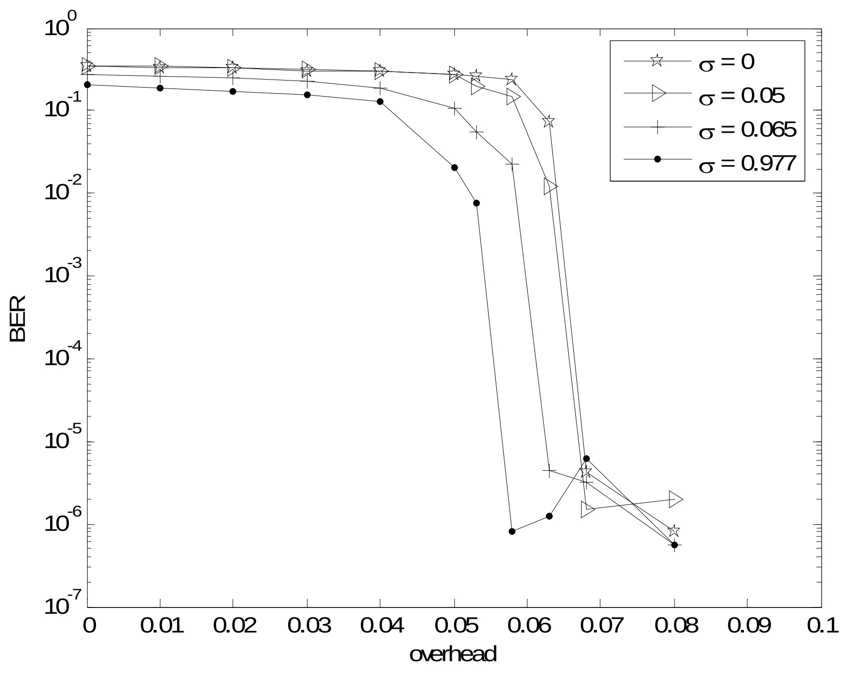

Fig.33 BER vs. overhead curve for $\sigma = \{0, 0.05, 0.065, 0.977\}$

5 CONCLUSION

5.1 Contributions

We proposed an iterative decoder for Raptor-like Codes in section 3.2 and analyzed this class of codes under the joint decoding framework using EXIT charts for the erasure channel in 3.4. In sections 3.5.1 and 3.5.2 we have shown that it is possible to design degree profiles for the LT code for a given LDPC code and for the LDPC code given a LT code respectively using LP. Further, we can design LT codes for incrementally redundant schemes as in section 3.6. From our designs in section 3.7.1, we conclude that there is a broader class of codes that can be designed under the iterative decoding framework and these codes can trade-off performance, complexity and storage. In 3.4, EXIT charts prove as a useful tool in analyzing the performance and the convergence behavior of Raptor-like codes. From the simulation results in section 3.7.2, we have observed that our designs match our analytical characterization. There is a slight deviation from the design parameters due to the fact that we have used asymptotic results to construct finite length codes.

In the BIAWGN channel, we show that it is possible to achieve good performance for a range of SNRs without losing significantly on the overhead for the entire range. The interesting result is the much smaller complexity required to achieve almost the same performance as that of existing designs [9]. We can design LT and LDPC codes simultaneously for a range of SNRs using EXIT charts and density evolution using LP as shown in 4.5.1 and 4.5.2. Since the designs in the Gaussian case involve some unavoidable approximations, we use DDE (section 4.3) to analyze the performance of the code designed. Using this framework, we can also design and analyze incrementally redundant codes.

5.2 Future Work

Existing design methodology involves designing LDPC and LT codes with random distributions on the input degree and bit degree profiles. In our work, we design the degree distribution of the check node of the LDPC code. We would prefer to introduce a deterministic procedure to design the bit node degree profile of the LDPC code. In other words, we should be able to choose the bit node profile and the Poisson parameter of LT code to suit a particular trade-off.

Currently all our designs are for the asymptotic lengths. However, we would like to extend these design methods to the finite length setting with random walk ideas on EXIT Charts. This will help us analyze and design better codes in the finite length case.

Another interesting property to investigate is to see if the EXIT curves for all the SNRS lies within the hull created by the EXIT curves corresponding to maximum and minimum SNR values. This might provide more insight into the universal code design problem for the BIAWGN Channel.

REFERENCES

[1]. S.Y. Chung, T. Richardson, and R. Urbanke, “Analysis of sum product decoding of low density parity check codes using a gaussian approximation,” *IEEE Trans. Information Theory,* vol. 47, no. 2, pp. 657-670, 2001.

[2]. S.Y. Chung, G. D. Forney, T. Richardson, and R. Urbanke, “On the design of low-density parity-check codes within 0.0045 dB of Shannon Limit,” *IEEE Communication Letters*, vol. 5, no. 3, 2001.

[3]. X. Hu, E. Eleftheriou, and D. Arnold, “Regular and irregular progressive edge-growth tanner graphs,” *IEEE Trans. Information Theory*, vol. 51, no. 1, pp. 386-398, 2005.

[4]. H. Pfister, I. Sason, and R. Urbanke, “Capacity-achieving ensembles for the binary erasure channel with bounded complexity,” *IEEE Trans. Information Theory,* vol. 51, no. 7, pp. 2352-2379, 2005.

[5]. M. Luby, “Information Additive Code Generator and Decoder for Communication Systems,” U.S. Patent,307 487, Oct. 23, 2001.

[6]. M. Luby, “LT-codes,” in Proc. 43rd Annu. *IEEE Symp. Foundations of Computer Science*, Vancouver, BC, Canada, Nov. 2002, pp. 271–280.

[7]. A. Shokrollahi, “Raptor codes,” *IEEE Trans. Information Theory*, vol. 52, no. 6, pp. 2551-2567, 2006.

[8]. “Technical Specification Group Services and System Aspects; Multimedia Broadcast/Multicast Services (MBMS); Protocols and Codecs (Release 6),” 3rd Generation Partnership Project (3GPP), Tech. Rep. 3GPP TS 26.346 V6.3.0, 3GPP, 2005.

[9]. O. Etesami, and, A. Shokrollahi, “Raptor codes on binary memoryless symmetric channels,” *IEEE Trans. Information Theory*, vol. 52, no. 5, pp. 2033-2051, 2006.

[10]. A. Ashikhmin, G. Kramer, and S. Brink, “Extrinsic information transfer functions: Model and erasure channel properties,” *IEEE Trans Information Theory*, vol. 50, no. 11, pp. 2657-2673, 2004.

[11]. K. Bhattad and K. R. Narayanan, "An MSE-based transfer chart for analyzing iterative decoding schemes using a Gaussian approximation," *IEEE Trans. on Information Theory*, vol. 53, no. 1, pp. 22-38, 2007.

[12]. R. G. Gallager, *Low Density Parity-check Codes*. Cambridge, MA: MIT Press, 1963.

[13]. A. Venkiah, C. Poulliat, and D. Declercq, “Analysis and design of raptor codes for joint decoding using information content evolution,” Submitted to *IEEE Int. Symp. on Information Theory* 2007.

[14]. T. Richardson, and R. Urbanke, “Modern Coding Theory,” 2007, available: http://lthcwww.epfl.ch/mct/index.php (accessed : May 2007).

[15]. R. Karp, M. Luby, and A. Shokrollahi, “Finite length analysis of LT codes,” in *Proc. IEEE Int. Symp. Information Theory*, Chicago, IL, Jun./Jul. 2004, p. 39.

[16]. H. Shankar, and K. R. Narayanan, "Memory-efficient sum-product decoding of LDPC codes," *To appear in IEEE Trans on Commn.*

VITA

Sabaresan Mothi Venkatesan received his Bachelor of Engineering degree in Electronics and Communication Engineering from PSG College of Technology (affiliated to Anna University) in May 2005. His undergraduate dissertation was on “A Novel System for Microarray Image and Data Analysis for Mutation Prognosis” which focused on application of signal processing techniques to microarray image processing.

He received his Master of Science degree in Electrical Engineering from Texas A&M University in August 2007. His graduate thesis was on “EXIT Chart Based Analysis and Design of Rateless Codes for the Erasure and Gaussian Channels”. His primary interests include channel coding, information theory and communications.

Mr. Mothi Venkatesan can be reached at CGGVeritas Inc., 10300, Town Park Drive, Houston, TX – 77072. His email address is mothi@aggienetwork.com.