Improved Rate-Compatible Low-Density Parity-Check Codes with Applications to Wireless Channels

by

Shaikh Faisal Zaheer

A Thesis Presented to the
DEANSHIP OF GRADUATE STUDIES

In Partial Fulfillment of the Requirements
for the Degree

MASTER OF SCIENCE

IN

Telecommunication Engineering

KING FAHD UNIVERSITY
OF PETROLEUM AND MINERALS

Dhahran, Saudi Arabia

May 2006

KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS

DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **Shaikh Faisal Zaheer** under the direction of his thesis advisor(s) and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE In Telecommunication Engineering.**

THESIS COMMITTEE

16/5/06
Dr. Salam A. Zummo (Chairman)

الاندلسي 16-5-2006
Dr. M. Adnan Landolsi (Co – Chairman)

16-5-2006
Dr. Saud A. Al – Semari (Member)

Dr. Maan A. Kousa (Member)

20-5-2006
Dr. Ali H. Muqaibel (Member)

Department Chairman
Dr. Jamil M. Bakhashwain

for

Dean of Graduate Studies
Dr. Mohammad A. Al-Ohali

Date June 6th, 2006

عمادة الدراسات العليا
DEANSHIP OF GRADUATE STUDIES

Dedicated to

My Beloved Parents

ACKNOWLEDGEMENTS

My deep appreciation and heartfelt gratitude goes to my thesis advisor Dr. Salam Zummo. He made himself readily available, and always had a patient ear, even for all the nonsense I had to say. I am extremely grateful to him for all his ultra-prompt help/e-mail replies and for his numerous proofreads that greatly improved this work.

I am also extremely grateful to my co-advisor Dr. Mohammad Adnan Landolsi for his guidance throughout the course of this research work. The focus on semi-random LDPC codes is very much due to Dr. Landolsi's previous work with them. I really enjoyed the many invaluable dicussions with him. I also acknowledge Dr. Asrar for his cooperation in providing the TRL lab facility.

I also thank the members of the evaluation committee Dr. Saud Al-Semari, Dr. Maan Kousa and Dr. Ali Muqaibel for serving on the committee. Their valuable comments have been immensely useful in helping me improve this thesis.

Acknowledgement is due to my very good friend Mudassir Masood for helping me on issues relating to LaTeXand MATLAB.

Sincere friendship is the spice of life. I owe thanks to my house mates, colleagues

and my friends for their help, motivation and pivotal support. A few of them are Dr. Hafiz M. Afzal, Juned Laiq, Mohammad Moinudin, Saad Azhar, Imran Azam, Imran Naseem, Adnan Shahab, Saad bin Mansoor and many others; all of whom I will not be able to name here. They made my work and stay at KFUPM very pleasant and joyful.

Family support plays a vital role in the success of an individual. I would like to thank my respected parents and siblings from the core of my heart. Their prayers and encouragement always help me take the right steps in life.

Contents

List of Tables

List of Figures

THESIS ABSTRACT

Name:	Shaikh Faisal Zaheer
Title:	Improved Rate-Compatible Low-Density Parity-Check Codes with Applications to Wireless Channels
Degree:	MASTER OF SCIENCE
Major Field:	Telecommunication Engineering
Date of Degree:	May 2006

Strong rate-compatible codes are important to achieve high throughput in hybrid automatic repeat request with forward error correction (ARQ/FEC) systems in networks utilizing packet data transmission. This thesis focuses on the construction of efficient rate-compatible low density parity check (RC-LDPC) codes over a wide rate range. Two LDPC code families are considered: the regular LDPC family which is known for good performance and low error floor, and semi-random LDPC which offers performance similar to Regular LDPC codes and also enables low encoding complexity. Efficient RC-LDPC codes are constructed and a hybrid ARQ/FEC system using RC-LDPC codes is evaluated. Compared to existing ARQ/FEC systems [1], the proposed system based on semi-random LDPC codes offers the advantages of low encoding complexity and higher throughput. The application of rate-compatible semi-random codes to systems employing coded cooperation diversity is investigated. Additionally an algorithm is developed for the removal of small loops from semi-random LDPC codes.

Keywords*: rate-compatible, low-density parity-check (LDPC) codes, puncturing, extending, automatic repeat request (ARQ).*

King Fahd University of Petroleum and Minerals, Dhahran.
May 2006

ملخص الرسالة

الأسم: شيخ فيصل زاهر
العنوان : شيفرات التفقد المتكافئة ذات النسبة التوافقية المنخفضة الكثافة للشبكات اللاسلكية
الدرجة : درجة الماجستير في الهندسة الكهربائية
التخصص الرئيسي: الأتصالات
تاريخ الحصول على الدرجة : حزيران 2006

كفاءة رموز النسبة المتوافقة مهمة للحصول على طاقة انتاجية عالية في الأنظمة الهجينة ذات الطلب المعاد الالي بتصحيح الخطأ التقادمي المستخدمة في الشبكات التي تبث المعطيات بطريقة حزمية (ARQ/FEC). هذه الرسالة تركز على بناء رموز مراقبة متكافئة فعالة ذات كثافة منخفضة على مدى نسبي عريض (RC-LDPC).
سيتم النظر في استخدام فئتان من فئات شفرة ال LDPC :وهاتان الفئتان هما :اولا", فئة شفرة ال LDPC العادية و التي تتميز بالاداء الجيد و معدل حطأ منخفض , ثانيا", فئة ال LDPC النصف عشوائية و التي تقدم اداءا" مماثلا" لفئة شفرة ال LDPC العادية و التي تمكن من الحصول على فك تشفير منخفض التعقيد. سيتم بناء نظم تشفير RC-LDPC فعالة و سيتم تقييمها على انظمة ال ARQ/FEC.
بالمقارنة مع انظمة ال ARQ/FEC المعمول بها حاليا" , وجد ان نظام ال ARQ/FEC و المزود بفئة شفرة ال LDPC النصف عشوائي يتميز بكونه منخفض التعقيد في فك التشفير و تكون طاقته الانتاجية مرتفعة على قيم SNR عالية القيمة .
سيتم البحث في استخدام شيفرات نصف عشوائية متوافقة النسبة في انظمة تعتمد تنويع التعاون المشفر و اخيرا " , سنقوم ببناء خوارزمية لازالة الدوائر الصغيرة من فئة تشفير ال LDPC النصف عشوائية .

كلمات دالة : نسبة متوافقة , شيفرة التفقد المتكافئ المنخفض الكثافة (LDPC), الطلب المعاد الالي (ARQ)

جامعة الملك فهد للبترول و المعادن, الظهران
ايار 2006

Chapter 1

Introduction

A major concern in data communications is how to control transmission errors caused by the channel impairments so that error-free data can be delivered to the user. An approach to this problem is the use of channel coding, that is use of error-detecting and/or error-correcting codes [1, 2]. There are two basic categories of error-control schemes for digital communications: automatic repeat request (ARQ) schemes and forward-error correction (FEC) schemes. ARQ combines error detection and retransmission strategies to ensure that data is delivered accurately despite occurrence of errors during transmission. On the other hand FEC tries to correct errors at the receiver.

Introducing channel coding in a communication system entails the transmission of redundant information along with the user information. ARQ schemes require the insertion of a small number of bits which can be used to detect errors (such as a CRC check bits [2]) to be transmitted along with the user information, while FEC schemes require insertion of bits that can be used to correct errors. To maximize the system throughput (defined as the ratio of user information bits to total transmitted bits) the amount of redundant bits used in an ARQ or FEC system have to be minimized. This entails the use of powerful error-correcting codes and efficient ARQ schemes.

Hybrid ARQ schemes combine both ARQ and FEC. This work is concerned with the design of hybrid ARQ schemes based on low-density parity-check (LDPC) codes. LDPC codes were introduced by Gallager [3]. They have been shown to offer - over a variety of channels - performance comparable to or better than that offered by other state-of-the-art codes such as turbo codes [4]. In fact, it is an irregular LDPC code (for definition see Section 2.1) with block length 10^7 that currently holds the distinction of being the world's best performing rate-1/2 code, outperforming all other known codes, and falling only 0.0045 dB short of the Shannon limit for the AWGN channel [4]. Finite-length LDPC codes have also been shown to outpermform turbo codes [5, 6]. This chapter introduces the basic background and the literature review on the work related to the thesis. Furthermore, the main objectives of the thesis are presented.

1.1 Background

The characteristics of wireless channels impose fundamental limits on the transmission range, data rate and the quality of a wireless communication service. The performance limits are influenced by several factors, most significantly the propagation environment and interference [7].

1.1.1 Characteristics of the Wireless Channel

The effects of the wireless channel on the received signal power are typically classified into large-scale and small-scale effects [7]. Large-scale effects involve the variation of the mean of the received signal power over large distances relative to the signal wavelength. On the other hand, small-scale effects involve the fluctuations of the received signal power over distances commensurate with the wavelength. Rapid variations in the received signal are caused by the multipath reception resulting from receiving different copies of the transmitted signal due to reflection, diffraction or scattering of the signal off surrounding objects before arriving over different paths at the receiver. These reflected signals arrive at the receiver at different delays resulting in random phase and amplitude of the received signals. This phenomenon is called multipath fading [7, 8].

Multipath fading is caused by superposition of the multiple versions of the transmitted signal received at the receiver at a given delay. The delay spread of the channel is defined as the time delay between the first signal component received at the receiver, and the component that takes the longest path from the transmitter to the receiver. The delay spread is characterized by its standard deviation, called the root mean square (RMS) delay spread of the channel. If the product of the RMS delay spread and the signal bandwidth is much less than unity, the channel is said to suffer from flat fading [8]. The discrete-time model of the received signal in a flat fading channel is

$$r_t = \sqrt{E_s}\alpha_t s_t + \eta_t, \tag{1.1}$$

where E_s is the average signal energy, α_t is the channel gain modeled as a zero-mean complex-valued Gaussian random variable with unity variance, i.e., $E[\alpha^2] = 1$, and η_t is a sample of a zero-mean additive white Gaussian noise (AWGN) with double-sided power spectral density $\frac{N_0}{2}$. The quantity $\frac{E_s}{N_0}$ is called the signal-to-noise ratio (SNR) per symbol. When the channel gain $|\alpha_t|$ follows a Rayleigh probability density function (pdf), the channel is said to be a *Rayleigh fading channel*. When a line-of-sight (LOS) path is present between the transmitter and receiver in addition to moving scatterers, the channel gain has a Rice distribution and the channel is said to be a *Rician fading channel* [9]. Another probability distribution that has been used extensively to model the envelop of fading channels is the Nakagami-m distribution [10], which was shown to fit empirical measurements very well.

The relative motion between the transmitter and the receiver (or vice versa) causes the frequency of the received signal to be shifted relative to that of the transmitted signal. The frequency shift, or Doppler frequency, is proportional to the velocity of the receiver and the frequency of the transmitted signal [7]. A signal undergoes *slow fading* when the bandwidth of the signal is much larger than the Doppler spread (defined as a measure of the spectral broadening caused by the Doppler frequency). The combination of the multipath fading with its time variations causes the received signal to degrade severely. This degradation of the quality of the recieved signal caused by fading needs to be counterbalanced by various techniques such as diversity and channel coding.

1.1.2 Channel Coding

The fundamental theory of error-correcting codes is often traced back to Shannon who proved the channel coding theorem in [11]. This theorem states that there exists an explicit upper bound, called the channel capacity, on the rate at which "information" can be transmitted reliably over a given communication channel. In paticular, the capacity of a bandlimited AWGN channel with bandwidth W, is given by

$$C = W \log_2(1 + E_s/N_0), \qquad \text{bits per second (bps)}, \tag{1.2}$$

where we assume perfect Nyquist signaling (i.e no inter-symbol-interference). The AWGN channel model approximately models many practical digital communication and storage systems. The proof of the theorem demonstrates that for any transmission rate R less than or equal to the channel capacity C, there exists a coding scheme that achieves an arbitrarily small probability of error. Conversely, if R is greater than C, no coding scheme can achieve reliable communication. However, since this is an existence theorem, it gives no guidance as to how to design appropriate coding schemes or how complex they may be to implement.

Channel coding improves the performance by adding redundant bits to the information bit stream that are used by the receiver to correct errors introduced by the channel, thus reducing the average bit error rate (BER). This approach enables a reduction in the transmit power required to achieve a target BER. Conventional FEC codes reduce the required transmit power for a given BER at the expense of increased signal bandwidth or a reduced data rate [2].

Conventional FEC codes use block or convolutional code designs. In block codes, parity bits are added to blocks of information bits. On the other hand, convolutional codes map a sequence of information bits onto a sequence of coded bits in sequential manner. Trellis codes combine channel code design and modulation to reduce the BER without bandwidth expansion or rate reduction [2]. Recent advances in coding technology, such as turbo codes [12] and LDPC codes [4] offer performance that approaches the channel capacity of AWGN and fading channels.

Linear Block Codes

A binary block code uses an encoder that accepts a block of message bits, and generates a block of coded bits (called a codeword) at the output. A code is *linear* if the addition of any two valid codewords results in another valid codeword. Similarly, a code is *cyclic* if a cyclic shift of any valid codeword results in another valid codeword [2]. In a binary code, each element of a codeword is a bit of value 0 or 1, whereas each element of a codeword in a non-binary code is a symbol (e.g., bytes). In this work we focus only on binary codes.

A block code is referred to as an (n,k) code if the size of the message is k bits, and the size of the overall codeword is n bits. Therefore, the number of redundant bits added to every k information bits is (n-k). The term systematic is used for codes in which the codeword contains the message bits in unaltered form as in Figure 1.1. Systematic codewords are formed by appending additional bits to the message bits. These additional bits are called redundancy or parity bits. A systematic block code is also specified by its generator matrix $G = [I_k \; P]$, where I_k is ($k \times k$) identity matrix and P is a $k \times (n\text{-}k)$ matrix that determines the (n-k) parity check bits. The systematic block code can also be specified by a parity-check matrix H of the form $H = [P^T \; I_{n-k}]$, where P^T is the transpose of the matrix P.

A (n,k) parity-check code is a linear block code whose codewords satisfy a set

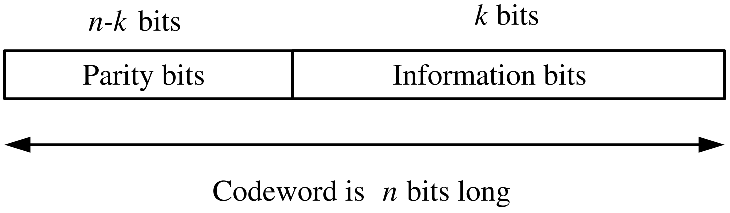


Figure 1.1: Codeword structure of a systematic linear block code

of (n-k) linear parity-check equations. It is traditionally defined by its (n-k) × n parity-check matrix H, whose (n-k) rows specify the (n-k) equations. For example, if the first equation specifies that bits 3 and 7 of a codeword must be equal, then the first row of H contains a one in columns 3 and 7 and zeros elsewhere. A parity-check code $\mathcal{C}$ is the set of codewords satisfying all parity-check equations, i.e., $\mathcal{C} = \{\mathbf{c} : \mathbf{c}H^T = \mathbf{0}\}$, where $\mathbf{c}$ and $\mathbf{0}$ are vectors of n elements each. Because each codeword of length n conveys k information bits, the code rate is defined as $R = k/n$.

An example of the structure of a codeword is shown in Figure 1.1. Increasing the number of parity bits (and thus increasing the value of n) will decrease the code rate; but it will allow the code to correct more errors. The *Hamming weight* of a codeword $\mathbf{c}$ denoted $\mathrm{w}(\mathbf{c})$, is defined to be the number of nonzero components of $\mathbf{c}$. For example, if $\mathbf{c} = (110101)$, then $\mathrm{w}(\mathbf{c}) = 4$. The *Hamming distance* between two codewords $\mathbf{c}_1$ and $\mathbf{c}_2$, denoted $\mathrm{d}(\mathbf{c}_1, \mathbf{c}_2)$, is the number of positions in which they differ [2]. For example if $\mathbf{c}_1 = (110101)$ and $\mathbf{c}_2 = (111000)$, then $\mathrm{d}(\mathbf{c}_1, \mathbf{c}_2) = 3$. Clearly, $\mathrm{d}(\mathbf{c}_1, \mathbf{c}_2) = \mathrm{w}(\mathbf{c}_1 + \mathbf{c}_2) = \mathrm{w}(\mathbf{c}_3)$ (addition is modulo-2), where $\mathbf{c}_3$ (for linear codes) is a codeword. Therefore, the distance between any two codewords equals the weight of one of the codewords and the *minimum distance* d_{min} for a linear block code equals the minimum weight of its nonzero codewords.

1.1.3 Codes Defined on Graphs

Since 1948, when Claude Shannon introduced the notion of channel capacity [11], the ultimate goal of coding theory has been to find practical capacity-approaching codes. Approaching the Shannon limit within a few decibels (dBs) was possible with practical decoding complexity, by using convolutional codes. However, reducing this gap required impractical complexity until the discovery of turbo codes [12]. One of the important innovations in turbo codes was the introduction of a class of low-complexity suboptimal decoding rules called the *iterative message-passing* algorithms. Using an iterative message-passing decoder, turbo codes provide excellent performance and a small gap to the Shannon limit with a low (practical) decoding complexity. Fig. 1.2 [13] compares the typical performance of a turbo code and a convolutional code over an uncorrelated Rayleigh fading channel. This amazing performance of turbo codes drew a lot of attention to the field of study, which soon extended to a more general class of codes called codes defined on graphs.

The advantage of codes defined on graphs is that they can be decoded using message-passing algorithms. The two important features of message-passing decoding which make codes defined on graphs so attractive, are its performance which is (potentially) very close to the optimal performance and its practical complexity which (for a fixed number of iterations) increases linearly with the length of the code. This, in turn, allows for the use of very long codes. Therefore, about 50 years

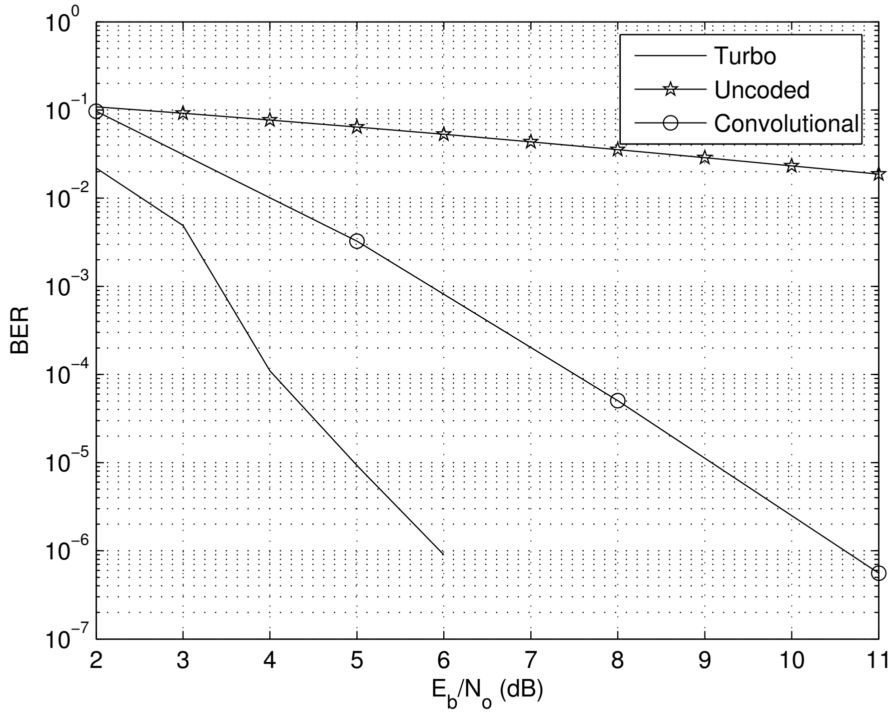


Figure 1.2: Comparison of the performance of a rate-$\frac{1}{2}$ (5/7,5/7,1) turbo code and a rate-$\frac{1}{2}$ (5,7) convolutional code over an uncorrelated Rayleigh fading channel.

after Shannons work, coding specialists are now able to design codes which can perform close to the Shannon limit with a reasonable decoding complexity. Moreover, for some channels, they have learned how the capacity can be achieved in principle, although the decoder requires an increasing complexity as the codes performance approaches capacity [14].

A Brief History of Codes Defined on Graphs

Interestingly, a graphical understanding of these cdoes was formed after the discovery of some of the later-called codes defined on graphs. Not only has this understanding helped coding theorists in analyzing these codes and in designing decoding algorithms for them, they also learned how to design their codes to get the best out of a given decoding algorithm.

The graphical understanding of codes started with the Tanner graphs for linear codes [15]. Later, Wiberg discovered that turbo codes can also be represented graphically [16]. Soon after this discovery, it was shown in [17] and [18] that the turbo decoding algorithm on the graphical representation of a turbo code is a special case of the belief propagation on the general Bayesian networks [19].

Parallel to the research on turbo codes and influenced by the focus on turbo codes, in 1996 MacKay and Neal [20], and Sipser and Spielman [21] rediscovered a long forgotten class of codes, i.e., LDPC codes. This class of codes was originally proposed in 1962 by Gallager [3], but were considered too complex at the time of their discovery. LDPC codes drew a lot of attention as they had an extremely good performance. For example they can be designed to perform a few hundredths of a decibel away from the Shannon limit over the AWGN channel. Another feature of LDPC codes is their simple graphical representation, which is based on Tanner's representation of linear codes [15]. This simple structure allows for accurate asymptotic ($n \to \infty$) analysis of LDPC codes [22] as well as the design of good irregular LDPC codes, optimized under specific constraints.

Since the rediscovery of LDPC codes, there has been a lot of research activities and improvements in the area of codes defined on graphs. Undoubtedly, research on LDPC codes has played and will continue to play a central role in this field, as many of the new classes of codes which are defined on graphs are influenced by the structure of LDPC codes. Examples of developments in the area of codes defined on graphs include the following

- Irregular LDPC codes: As shown in [23], irregular LDPC (see Section 2.1) codes can significantly outperform regular LDPC codes. All LDPC codes which approach the Shannon limit on different channels are irregular LDPC codes.

- Capacity achieving LDPC codes for the Binary Erasure Channel (BEC): Shokrollahi *et. al.* found a family of irregular LDPC codes that could achieve the capacity of the BEC [24, 25].

- Density evolution analysis of LDPC codes: An accurate asymptotic analysis of LDPC codes under different decoding schemes was proposed in [22]. The idea is to follow the evolution of the density of the messages in the decoder. Using this analysis, the design of good irregular LDPC codes, which has already been studied for the BEC became possible for other channel types.

- Gaussian approximation of the analysis of turbo and LDPC codes [26, 27]: Due to the high computational complexity of the density evolution approach, approximations of density evolution attracted many researchers. In particular, approximating the true message density with a Gaussian density seemed to be very effective.

- Extrinsic Information Transfer (EXIT) chart analysis: EXIT chart analysis [28] is similar to density evolution, except that it follows the evolution of a single parameter that represents the density of messages. This evolution can be visualized in a graph called an EXIT chart. EXIT charts have become very popular, as they provide deep insight into the behaviour of iterative decoders [14, 26, 29].

The research in the area of codes defined on graphs is still very active and there are many open problems under study.

1.1.4 Automatic Repeat Request (ARQ)

In wireless networks transmission takes place in the form of packets, which are blocks of information bits. Another way - in addition to channel coding - to reduce the link errors prevalent in wireless systems is to implement retransmissions. Automatic repeat request (ARQ) combines error detection and retransmission to ensure that data is delivered accurately despite occurrence of errors during transmission. Based on retransmission strategies, there are three basic types of ARQ schemes: stop-and-wait ARQ, go-back-N ARQ, and selective-repeat ARQ [2].

The stop-and-wait scheme represents the simplest ARQ procedure and was implemented in early error-control systems. In a stop-and-wait ARQ error-control system, the transmitter sends a codeword to the receiver and waits for an acknowledgment. A positive acknowledgment (ACK) from the receiver indicates that the transmitted codeword has been successfully received, and the transmitter sends the next codeword in the queue. A negative acknowledgment (NACK) from the receiver indicates that the transmitted codeword has been detected in error; the transmitter then resends the codeword and again waits for an acknowledgment. Retransmissions continue until the transmitter receives an ACK.

In the basic go-back-N ARQ scheme, the transmitter continuously transmits codewords in order and then stores them pending receipt of an ACK/NACK for each. The acknowledgment for a codeword arrives after a round-trip delay, defined as the time interval between the transmission of a codeword and the receipt of an acknowledgment for that codeword. During this interval, N - 1 other codewords are also transmitted. Whenever the transmitter receives a NACK indicating that a particular codeword, say codeword i, was received in error, it stops transmitting new codewords. Then it goes back to codeword i and proceeds to retransmit that codeword and the N - 1 succeeding codewords which were transmitted during one round-trip delay. At the receiving end, the receiver discards the erroneously received word i and all N - 1 subsequently received words, whether they are error-free or not. Retransmission continues until codeword i is positively acknowledged. In each retransmission for codeword i, the transmitter resends the same sequence of codewords. As soon as codeword i is positively acknowledged, the transmitter proceeds to transmit new codewords. In a selective-repeat ARQ error-control system, codewords are also transmitted continuously. However, the transmitter only resends those codewords that are negatively acknowledged (NACKed). After resending a NACKed codeword, the transmitter continues transmitting the new codewords.

The combination of ARQ and FEC is a powerful method of increasing the system efficiency. This combination is called Hybrid ARQ [2]. In *type-I Hybrid* ARQ, a packet is encoded for both error detection and error correction. When the packet arrives at the receiver it is first decoded by the FEC decoder and then checked for errors [30]. If errors are detected, a retransmission request is sent back to the transmitter. Otherwise the packet is accepted. *Type-II Hybrid* ARQ adapts to changing channel conditions through the use of *incremental redundancy* [30]. Mandelbaum [31] was the first to propose punctured codes (puncturing denotes deletion of parity bits) for transmitting redundancy in incremental steps. A packet is encoded for both error detection and error correction. In this scheme some of the parity bits are punctured before transmission. At the receiver the packet is decoded by the FEC decoder. If errors are detected then the transmitter sends a group of parity bits that were not sent to the receiver. The receiver appends these bits to the received packet allowing for increased error correction capability.

1.1.5 Rate-Compatible (RC) Codes

In [32] the concept of punctured codes was modified for the generation of a family of codes by adding a rate-compatibility restriction to the puncturing rule. The restriction implies that all the coded bits of a high-rate punctured code are used by the lower-rate codes. In other words, the high-rate codes are embedded into the lower-rate codes of the family. If the higher rate codes are not sufficiently powerful to decode channel errors, only supplemental bits which were previously punctured (deleted) have to be transmitted in order to improve the code. For block codes, rate-compatible codes can be obtained by *puncturing*, *extending* or a combination of the two approaches. The range of code rates for a family of rate-compatible codes is defined as

$$R = \frac{P}{P+l}, \quad l = 1, \ldots, l', \ldots, L. \tag{1.3}$$

For a family of codes obtained from a mother code of rate $\frac{P}{P+l'}$, the code rates $\frac{P}{P+1}$ to $\frac{P}{P+l'-1}$ would be obtained through puncturing, while the code rates $\frac{P}{P+l'+1}$ to $\frac{P}{P+L}$ would be obtained through extending.

Puncturing

A code is punctured by deleting parity bits (information bits can also be deleted resulting in a shortened code [33]). The punctured code rate has a higher rate than the original code. Puncturing enables higher bandwidth efficiency at the expense of degradation in performance [33].

Since we are interested in rate-compatible punctured codes, the following restriction needs to be enforced: for a series of desired rates $R_{j+n} > R_{j+n-1} > ... > R_j > R$, where R is the rate of the mother (non-punctured) code, the punctured parity bits that yield rate R_{l+n} have to form a subset of the punctured parity bits that yield rate R_{l+n-1} (the high-rate codewords are embedded in the low-rate codewords [32]).

Extending

A code is extended by annexing additional parity check bits. The extended code has a lower rate than the original code. Extending leads to codes of increased minimum distance [33] and better performance. If H is the matrix (of size $m \times n$) representing the original code, H_{ext} (which is the extended version of size $((m+u) \times (n+u))$ will contain additional rows and columns. These additional rows and columns have to be added so that there is a strong dependency between the columns of the parity-check matrix of the original code and the newly added columns [1, 5].

Extending builds RC codes from high rates to low rates by adding more parity bits. For RC-LDPC codes built from extending, the initial transmission corresponds to a LDPC code, which has a good FER in the first transmission. Then additional parity bits are added to reduce the rate in such a way that the extended code provides sufficiently better performance compared to the original code.

Another motivation for using extending concerns the encoding complexity. Un-

$$\begin{tikzcd}
H = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \arrow[d, shift left=2ex, "Puncturing"] \\
H = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix} \arrow[u, shift left=2ex, "Extending"]
\end{tikzcd}$$

Figure 1.3: The effect of Puncturing and Extending on a parity-check matrix H.

like puncturing where all parity bits are generated at the encoder regardless whether they will be used, extending allows bits to be generated only as needed, thus avoiding unnecessary computations at the encoder and the decoder.

For rate-compatible extended codes obtained through extending, the following restriction needs to be enforced: for a series of desired rates $R_{j+n} < R_{j+n-1} < ... < R_j < R$, where R is the rate of the mother (non-extended) code, the additional parity bits that yield rate R_{l+n-1} have to form a subset of the parity bits that yeild rate R_{l+n} (the high-rate codewords are embedded in the low-rate codewords [32]).

1.1.6 Coded Cooperation

Diversity is considered an effective tool for combating multipath fading [8]. Diversity is achieved by effectively transmitting or processing independently faded copies of the signal. Among diversity techniques, transmit diversity relies on the principle that signals transmitted from geographically separated transmitters experience independent fading, which results in a significantly improved performance compared to systems with no diversity [34, 35]. Since most wireless networks operate in a multiuser mode, *user cooperation* [36, 37] can be employed to provide diversity. In user cooperation, mobile units share their antennas to achieve uplink transmit diversity as illustrated in Figure 5.1. Since signals transmitted by different users undergo independent fading paths to the base station (BS), this approach achieves spatial diversity through the partner's antenna.

In conventional user cooperation the partner repeats the received bits (via either forwarding or hard detection). Recently, a new framework for user cooperation was proposed [38, 39, 40] and is called *coded cooperation.* Unlike conventional user cooperation schemes, symbols in coded cooperation are not repeated by the partner. Instead, the codeword of each user is partitioned into two parts: one part is transmitted by the user, and the other part is sent by his partner. Coded cooperation provides significant performance gains for a variety of channel conditions. In addition, by allowing different code rates through rate-compatible coding [32], coded cooperation provides a great degree of flexibility to adapt to channel conditions.

1.2 Literature Survey

1.2.1 Rate-Compatible Punctured Convolutional and Turbo Codes

Punctured convolutional codes were first introduced by Cain *et.al.* [41] mainly for the purpose of obtaining simpler Viterbi decoding. They obtained codes of rates 2/3 and 3/4 by puncturing rate-1/2 codes. These punctured codes were almost as good as the best known codes.

In [42] a criterion was proposed for selection of the puncturing pattern for turbo codes. The rate-compatible punctured turbo (RCPT) coded system of [42] was shown to outperform the RCPC codes of Hagenauer [32]. In [43, 44] the authors proposed a technique for finding the accurate weight distribution of punctured turbo codes. This technique enabled the study and comparison of different puncturing patterns and provided guidelines for designing good puncturing patterns. In [45] the design criteria for search of good rate-compatible systematic turbo codes were proposed and compared.

1.2.2 Rate-Compatible LDPC Codes

The conventional approach of puncturing achieves a range of higher code rates by successively puncturing larger fractions of the codeword bits of a low-rate code.

Using regular LDPC codes in [1] it was shown that puncturing has a larger adverse impact when the mother code is of low rate than when the mother code is of high rate. Hence, for a fixed desired rate (after puncturing), it is desirable to choose the mother code such that the percentage of punctured bits is as small as possible. However, this will result in a limited range of achievable code rates. To overcome this problem, extending was used to build codes of lower rates. An ARQ system employing the RC-LDPC codes of [1] was shown to perform on par with ARQ systems from [42] using RCPT codes.

The work in [1] was extended in [5] to the case of irregular LDPC codes by employing both puncturing and extending. An ARQ system employing this scheme was shown to outperform the system of [1] and systems based on turbo codes [42] by up to 0.5 dB. In [46] optimal puncturing distributions for irregular LDPC codes were obtained from the perspective of minimizing threshold (defined as the SNR value above which the probability of decoding error approaches zero, while for values of SNR lower than threshold the probability of decoding error is nonzero [4]). The theoretical performance of punctured LDPC codes was analyzed with Gaussian approximation [4]. Based on the analysis, a design rule for good puncturing distributions was proposed. The results apply to LDPC codes of large block length (simulation results are given for a block-length of 131072 bits). The punctured LDPC codes obtained using this approach were shown to outperform codes obtained using random puncturing. Recently in [47], a systematic method has been proposed for finding good puncturing distributions for finite-length LDPC codes. The idea is based on the fact that a punctured node will be recovered with reliable messages when it has 1) more neighboring checknodes, and 2) each of the checknodes has more reliable neighbors (variable nodes) except for the punctured one. For example, a punctured variable node that has checknodes whose remaining neighboring variable nodes are unpunctured will have nonzero messages from the checknodes in the first iteration. The process for a punctured node to have messages from checknodes is called *recovery*. The punctured node in the preceding example will be called a one-step-recoverable (1-SR) since the node is recovered in the first iteration. The 1-SR nodes and unpunctured nodes will help recover some of the remaining punctured nodes in the second iteration, and so on. In general, the punctured nodes recovered in the ith iteration are called i-SR nodes. It is assumed that the more iterations a punctured node needs for its recovery, the less statistically reliable the recovery message is. Thus, it is better to puncture nodes that require a smaller number of iterations, which results not only in less iterations to decode codewords but also in better performance at a given code rate. This method enables the design of punctured finite-length LDPC codes that outperform randomly punctured LDPC codes.

1.2.3 Coded Cooperation

In [39, 40], the performance of coded cooperation diversity employing RCPC codes with two users is investigated. Furthermore, in [40],the application of turbo codes to coded cooperation is also investigated. In [48], the error performance of coded cooperation diversity employing RCPC codes with multiple (≥ 2) cooperating users is analyzed.

1.3 Thesis Contributions

LDPC codes are one of the most important codes defined on graphs. This is due to their excellent performance as well as their simple yet flexible structure. LDPC codes offer the following advantages over turbo codes [49]:

- The complexity of (belief propagation) decoding is less than that of turbo-codes [50, 51], and being fully parallelizable, can potentially be performed at significantly greater speeds [52].

- Very low complexity decoders that closely approximate belief-propagation in performance have been designed for these codes [22].

- LDPC decoding is verifible in the sense that decoding to a correct codeword is a detectable event. Therefore the need for a error-detecting code is obviated [1].

Furthermore, LDPC codes have lower error floors [14, 53]. LDPC codes are already used in some standards such as ETSI EN 302 307 for digital video broadcasting [54] and IEEE 802.16 (Broadband Wireless Access Working Group) for coding on orthogonal frequency division multiple access (OFDMA) systems [55].

As stated above, in this work we focus on the design of efficient Hybrid ARQ schemes that utilize RC-LDPC codes. For such schemes it is desirable to use small block lengths (specially for wireless systems that are designed for mobile use) due to the following reasons

- Reduced encoding and decoding complexity.
- A reduced retransmission frame size in case of a frame error leads to an increase in the throughput.

Therefore this work focuses on designing RC-LDPC codes for relatively small block lengths in the range of 512-2048 bits. Furthermore, the existing RC-LDPC codes for finite block-lengths (with the exception of [47]) use *random* puncturing. In this work, we explore systematic puncturing techniques that result in punctured LDPC codes with good performance. Additionally,

- We investigate the application of RC-LDPC codes for wireless networks employing coded cooperation diversity.
- Removal of small loops from a LDPC code improves the performance in terms

of lower BER and reduced error floor [56]. In this work we develop an algorithm for removing small loops from semi-random LDPC codes.

.

1.4 Thesis Outline

In Chapter 2 we provide the necessary background on LDPC codes and their decoding algorithms. In Chapter 3 RC-LDPC codes are designed based on the regular family of LDPC codes. In particular, a heuristic algorithm is proposed for selecting puncturing patterns that results in codes with low error floor.

In Chapter 4 RC-LDPC codes based on the Semi-Random family of LDPC codes are developed. A puncturing pattern for this class of codes which offers good performance for both the low and high SNR regions is proposed. In addition two efficient methods of extending this class of codes are also proposed. The designed rate-compatible LDPC codes outperform existing systems based on regular LDPC codes.

In Chapter 5 the use of punctured semi-random LDPC codes for Coded Cooperation diversity is investigated. Furthermore, the effect of varying the cooperation level is investigated for different interuser channels.

Small cycles degrade the performance of the LDPC decoder. In Chapter 6 an alogrithm is developed for the removal of small cycles from a class of efficiently-encodable LDPC codes called Semi-Random LDPC codes [57]. The algorithm enables the design of codes with low error floors.

In Chapter 7 we provide a summary of the conclusions drawn from this work, and some suggestions for future work.

Chapter 2

LDPC Codes

LDPC codes are block codes defined by a sparse parity-check matrix. They were first proposed in 1962 by Robert Gallager [3], along with an elegant iterative decoding scheme whose complexity grows only linearly with the code block length. Despite their promise, LDPC codes were largely forgotten for several decades, primarily because the computers at that time were not powerful enough to use them. In 1993 Berrou *et. al.* [12] proposed turbo codes. This new encoding/decoding technique, with complexity that is slightly larger than that of convolutional codes, enables performance approaching the Shannon capacity of the AWGN channel within a fraction of a decibel. The invention of turbo codes led the research community to focus on iterative decoding algorithms.

In 1995 LDPC codes were rediscovered by MacKay and Neal [53], sparking a flurry of further research on coding theory. Today the value of LDPC codes is widely recognized. Their remarkable capacity-approaching performance ensures that they will not be forgotten again. In contrast to many existing coding schemes, LDPC codes offer both better performance and lower decoding complexity.

2.1 Code Structure

A low-density parity-check (LDPC) code is defined by a parity-check matrix that is sparse. A regular (*j*,*l*) LDPC code is defined by an (n-k) × n parity-check matrix having exactly j ones in each column and exactly l ones in each row, where $j < l$ and both are small compared to n [2]. An irregular LDPC matrix is also sparse, but not all rows and columns contain the same number of ones [2].

Figure 2.1 shows the parity-check matrix of a (3,6) LDPC code. By the definition of regular LDPC codes, every parity-check equation involves exactly l bits, and every bit is involved in exactly j parity-check equations. Observe that the fraction of ones in a regular (*j*,*l*) LDPC matrix is l/n. The "low density" terminology derives from the fact that this fraction approaches zero as $n \rightarrow \infty$ [58]. In contrast, the average fraction of ones in a purely random binary matrix (with independent components equally likely to be zero or one) is 1/2.

Any parity-check code (including an LDPC code) may be specified by a Tanner graph, which is essentially a visual representation of the parity check matrix H [2]. Recall that an (n-k) × n parity-check matrix H defines a code in which the

n bits of each codeword satisfy a set of (n-k) parity-check equations. The Tanner graph contains n "variable" nodes, one for each codeword bit, and (n-k) "check" nodes, one for each of the parity-check equations. Figure 2.2 shows the Tanner graph corresponding to the H matrix of Figure 2.1. The variable nodes are depicted using circles, while the check nodes are depicted using squares. The check nodes are connected to the variable nodes they check through *edges.* Specifically, an edge connects a check node x to a variable node y if and only if the x-th parity check involves the y-th bit, or more succinctly, if and only if $H_{x,y} = 1$, where $H_{x,y}$ is element corresponding to the x-th row and y-th column in a parity-check matrix H. The graph is said to be bipartite since it contains two distinct types of nodes, variable nodes and check nodes [2]. Note that in the bipartite graph there can be no direct connection between any two nodes of the same type [2].

For the special case of a (j,l) regular LDPC code, each bit is involved in j parity check equations. Hence, the number of edges emanating from a variable node is always j and the variable node is said to be of *degree* j [2]. Similarly, because each parity check equation involves l bits, the number of edges emanating from each check node is always l and the check node is said to be of *degree l.*

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & \textcircled{1} & 1 & 1 & \textcircled{1} & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & \textcircled{1} & 0 & 0 & \textcircled{1} & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Figure 2.1: A regular (3,6) parity-check matrix *H*, the circled 1s show a 4-loop

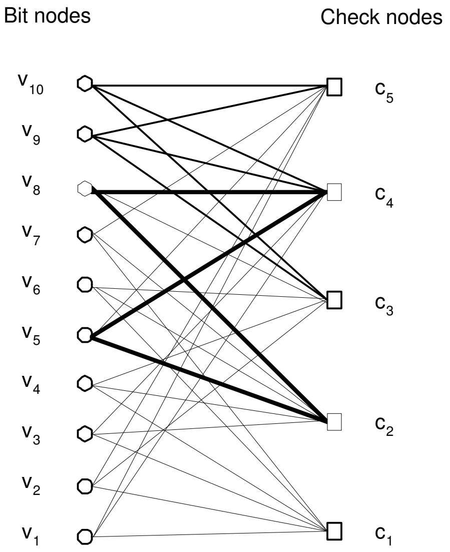


Figure 2.2: Tanner graph corresponding to H matrix of Figure 2.1, the bold edges show a 4-loop

Since the minimum distance of a linear block code is the weight of the codeword with the minimum Hamming weight, the sparse structure of LDPC codes ensures large minimum distance. This can be explained as follows: $d_{min} = min\{w_H(\mathbf{c}) : \mathbf{c}H^T = \mathbf{0}\}$. The operation $\mathbf{c}H^T$ adds selected rows of H^T and it would take a large number of such rows to sum to $\mathbf{0}$ if H is sparsely populated with 1s [59].

In the bipartite graph representing a LDPC code, a loop (or cycle) is a closed path with no repeated nodes, and must therefore be of even length [2]. There is at most one edge between any two nodes, and so the shortest length a loop can have is 4. Such loops are referred to as 4-1oops. In general a loop of length m is called an m-loop. The girth of the graph is defined as the length of the shortest loop. A *Stopping set* S is a subset of V, the set of variable nodes, such that all neighbors of S

are connected to S at least twice [60]. The *stopping number* of a code is the size of its smallest stopping set, and the stopping number lower bounds the minimum distance of the code [61]. The stopping number of a code can be increased by increasing its girth, and hence codes with larger girth have lower error floors [61].

In the absence of loops, the iterative decoding algorithm converges to the maximum-likelihood solution [62]. However, most random constructions of the parity-check matrix contain loops of small lengths [56].

2.2 Semi-Random LDPC Codes

For the purpose of encoding, the parity-check matrix H has to be transformed into the systematic form using Gaussian elimination [2]. The generator matrix G is then obtained from the systematic form of the H matrix. This transformation usually destroys the sparseness of the H matrix, resulting in a complex encoding process. Recent contributions have shown that LDPC codes are also amenable to simple encoding structures [63, 64]. In [57], a method has ben proposed for the construction of the parity-check matrix which enables simple encoding while at the same time provides performance similar to regular random codes. These LDPC codes are called semi-random codes (Figure 2.3). The parity-check matrix is obtained by concatenation of a deterministic sub-matrix with a randomly constructed sub-matrix.

The systematic codeword is expressed as $\boldsymbol{c} = [\boldsymbol{p}\ \boldsymbol{d}]$, where $\boldsymbol{p}$ is a vector containing

$$H = \left(\underbrace{\begin{matrix} 1 & & & 0 \\ 1 & 1 & & \\ & 1 & 1 & \\ 0 & & 1 & 1 \end{matrix}}_{\textit{Deterministic}} \middle| \underbrace{\begin{matrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{matrix}}_{\textit{Random}} \right)$$

Figure 2.3: A Semi-Random parity-check matrix

the parity bits and $\boldsymbol{d}$ is a vector containing the data bits. The parity-check matrix is decomposed as $H = [H^{\boldsymbol{p}}\ H^{\boldsymbol{d}}]$. Since every codeword $\boldsymbol{c}$ should satisfy the parity-check equations, we can write

$$\begin{bmatrix} H^p & H^d \end{bmatrix} \begin{bmatrix} \boldsymbol{p} & \boldsymbol{d} \end{bmatrix}^T = \mathbf{0} \tag{2.1}$$

From (2.1), the parity vector $\boldsymbol{p}$=$\{p_i\}$ can be calculated from the information sequence $\boldsymbol{d}$=$\{d_i\}$ as [57]

$$p_1 = \sum_j h^d_{1j} d_j \quad \text{and} \quad p_i = p_{i-1} + \sum_j h^d_{ij} d_j \pmod 2. \tag{2.2}$$

As shown by (2.2) the encoding can be carried out recursively, and the complexity of the encoding grows linearly with the block-length.

2.3 Decoding

Decoding is preferred via the iterative sum-product algorithm (also known as the belief propagation algorithm) [2]. This algorithm closely approximates the maximum-likelihood decoding rule with a complexity that grows linearly with the code block length [3]. In the following the decoding process is described following the notation given in [58].

2.3.1 Background and Terminology

The probability distribution for a binary random variable $c \in \{0,\ 1\}$ is uniquely specified by the single parameter $p = \Pr[c = 1]$, since $\Pr[c = 0] = 1$ - p. Alternatively, the probability distribution is also uniquely specified by the ratio given by

$$\lambda = \log \frac{Pr[c = 1]}{Pr[c = 0]}\ . \tag{2.3}$$

The sign of λ indicates the most likely value for c; λ is positive when 1 is more likely than 0, and λ is negative when 0 is more likely than 1. Moreover, the magnitude $|\lambda|$ is a measure of certainty or reliability of λ. At one extreme, if $\lambda = 0$ then 0 and 1 are equally likely. At the other extreme, if $\lambda = \infty$ then $c = 1$ with probability 1, and $\lambda =$ - ∞ implies that $c = 0$ with probability 1.

Given a random bit $c \in \{0,\ 1\}$, let r denote an observation whose pdf depends on c according to the function *f(*$r|c$*)*. When c is fixed and *f(*$r|c$*)* is viewed as a function of r, it is called a conditional pdf. On the other hand, when r is fixed, then *f(*$r|c$*)* as a function of c is called the likelihood function [58].

Before making an observation, the *a priori* probabilities for c are $\Pr[c = 1]$ and $\Pr[c = 0]$. After making an observation, these probabilities change to the *a posteriori* probabilities (APP) $\Pr[c = 1|r]$ and $\Pr[c = 0|r]$. Because of the Bayes rule, the a posteriori probability is proportional to the likelihood function:

$$Pr[c = 1|r] = \frac{f(r|c)Pr[c = 1]}{f(r)} \;. \tag{2.4}$$

Hence the a posteriori probabilities can be expressed as:

$$\log \frac{Pr[c = 1|r]}{Pr[c = 0|r]} = \log \frac{f(r|c = 1)}{f(r|c = 0)} + \log \frac{Pr[c = 1]}{Pr[c = 0]} \;. \tag{2.5}$$

The first term on the right-hand side is called the log-likelihood ratio (LLR). Strictly speaking, the second term on the right-hand side is a log-probability ratio, and the left-hand side is a log-APP ratio. The second term on the right-hand side is more commonly called the a priori LLR, and the left-hand side is called the a posteriori LLR. If c is equally likely to be zero or one, then the a priori LLR is zero, and the a posteriori LLR is equal to the LLR.

2.3.2 The Tanh Rule

Let $\phi(\mathbf{c})\in \{0,\, 1\}$ denote the value of the parity bit of a vector $\mathbf{c} = [c_1, \ldots, c_n]$ of n bits, so that $\phi(\mathbf{c}) = 0$ if there are an even number of ones in $\mathbf{c}$, and $\phi(\mathbf{c}) = 1$ if the number of ones in $\mathbf{c}$ is odd. If the bits are independent, the a priori LLR for the value of the parity bit $\phi(\mathbf{c})$ (i.e $\lambda_{\phi(\mathbf{c})}$) obeys the tanh rule [4]

$$\tanh\left(\frac{-\lambda_{\phi(\mathbf{c})}}{2}\right) = \prod_{i=1}^{n} \tanh\left(\frac{-\lambda_i}{2}\right), \tag{2.6}$$

where λ_i denotes the a priori LLR for the i-the bit in $\mathbf{c}$ (given in (2.3)).

2.3.3 The Decoding Problem

Consider the problem of decoding a LDPC code with a parity-check matrix H over a flat fading channel with AWGN at the receiver, so that the t-th element of the receiver observation vector $\mathbf{r} = [r_1,\ldots,r_n]$ is related to the transmitted codeword $\mathbf{c}$ $= [c_1,\ldots,c_n]$ by

$$r_t = -\alpha_t(-1)^{c_t} + \eta_t, \tag{2.7}$$

where α_t is the channel gain affecting the t-th bit in $\mathbf{c}$ and η_t is a zero-mean Gaussian random variable with variance $\frac{N_o}{2}$, where $\frac{N_o}{2}$ is the double-sided power spectral density. The detector that minimizes the probability of error for the t-th bit would calculate the aposteriori LLR:

$$\begin{aligned}\lambda_t &= \log \frac{Pr[c_t = 1|\mathbf{r}]}{Pr[c_t = 0|\mathbf{r}]} \\ &= \log \frac{Pr[c_t = 1|r_t, \{r_{i\neq t}\}]}{Pr[c_t = 0|r_t, \{r_{i\neq t}\}]},\end{aligned} \tag{2.8}$$

and then decide $c_t = 1$ if $\lambda_t > 0$, and c_t= 0 otherwise. Applying Bayes rule, the numerator in (2.8) can be written as

$$\begin{aligned}\Pr[c_t = 1|r_t, \{r_{i\neq t}\}] &= \frac{f(r_t, c_t = 1, \{r_{i\neq t}\})}{f(r_t, \{r_{i\neq t}\})} \\ &= \frac{f(r_t|c_t = 1, \{r_{i\neq t}\})f(c_t = 1, \{r_{i\neq t}\})}{f(r_t|\{r_{i\neq t}\})f(\{r_{i\neq t}\})} \\ &= \frac{f(r_t|c_t = 1)\Pr[c_t = 1|\{r_{i\neq t}\}]}{f(r_t|\{r_{i\neq t}\})}\end{aligned} \tag{2.9}$$

The last equality exploits the fact that, given c_t, r_t is independent of $r_{i\neq t}$. The denominator of (2.8) can be similarly expressed. Hence (2.8) simplifes to

$$\begin{aligned}\lambda_t &= \log\frac{f(r_t|c_t=1)Pr[c_t=1|\{r_{i\neq t}\}]}{f(r_t|c_t=0)Pr[c_t=0|\{r_{i\neq t}\}]}\\ &= \log\frac{f(r_t|c_t=1)}{f(r_t|c_t=0)}+\log\frac{Pr[c_t=1|\{r_{i\neq t}\}]}{Pr[c_t=0|\{r_{i\neq t}\}]}\\ &= \underbrace{\frac{2}{\sigma^2}r_t\alpha_t}_{intrinsic}+\underbrace{\log\frac{Pr[c_t=1|\{r_{i\neq t}\}]}{Pr[c_t=0|\{r_{i\neq t}\}]}}_{extrinsic}, \end{aligned} \tag{2.10}$$

where we used the fact that

$$f(r_t|c_t) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\frac{-(r_t+\alpha_t((-1)^{c_t})}{2\sigma^2} \tag{2.11}$$

The first term in (2.10) represents the contribution from the t-th channel observation, and is called the *intrinsic* information, while the second term represents the contribution from the observations in the other terms in r_t, and is called the *extrinsic* information. Because the j parity-check equations of the code ensure that $c_t = \phi_{(\mathbf{c}(i))}$ for all i = 1,...,j, we can rewrite (2.10) as

$$\lambda_t = \frac{2}{\sigma^2}r_t\alpha_t+\log\frac{Pr[\phi_{\mathbf{c}(i)}=1, i=1,\ldots,j|\{r_{i\neq t}\}]}{Pr[\phi_{\mathbf{c}(i)}=0, i=1,\ldots,j|\{r_{i\neq t}\}]}, \tag{2.12}$$

where $\mathbf{c}_{(i)}$ denotes the set of codeword bits involved in the i^{th} parity check equation excluding c_t. If the graph is cycle-free, the vectors $\mathbf{c}_{(1)}$, $\mathbf{c}_{(2)}$,.., $\mathbf{c}_{(j)}$ are conditionally independent given $\{r_{i\neq t}\}$ and furthermore, the components of $\mathbf{c}_{(i)}$ are themselves conditionally independent given $\{r_{i\neq t}\}$. Hence, (2.12) reduces to

$$\begin{aligned}\lambda_t &= \frac{2}{\sigma^2} r_t + \log \frac{\prod_{i=1}^{j} \Pr[\phi(c_{(i)}) = 1|\{r_{i\neq t}\}]}{\prod_{i=1}^{j} \Pr[\phi(c_{(i)}) = 0|\{r_{i\neq t}\}]} \\ &= \frac{2}{\sigma^2} r_t + \sum_{i=1}^{j} \log \frac{\Pr[\phi(c_{(i)}) = 1|\{r_{i\neq t}\}]}{\Pr[\phi(c_{(i)}) = 0|\{r_{i\neq t}\}]} \\ &= \frac{2}{\sigma^2} r_t \alpha_t + \sum_{i=1}^{j} \lambda_{\phi(\mathbf{c}_{(i)})} \ . \qquad (2.13)\end{aligned}$$

If we introduce

$$\lambda_{i,p} = \log \frac{Pr[c_{i,p} = 1|\{r_{i\neq t}\}]}{Pr[c_{i,p} = 0|\{r_{i\neq t}\}]} \ , \qquad (2.14)$$

where in $\lambda_{i,p}$, i denotes the i-the parity check equation which involves c_t and p denotes the p-th bit involved in the i-the parity check equation, then substituting (2.6) in (2.13), the LLR of the t-th bit becomes

$$\lambda_t = \frac{2}{\sigma^2} r_t \alpha_t - 2 \sum_{i=1}^{j} \tanh^{-1} \left(\prod_{p=2}^{l} \tanh \left(\frac{\lambda_{i,p}}{2} \right) \right) . \qquad (2.15)$$

With the aid of the Tanner graph, we may interpret (2.15) in terms of messages passed from a bit-node to a check-node and vice versa. Suppose the variable node associated with $c_{i,p}$ passes the “message” $\lambda_{i,p}$ to the i-th check node. In turn, the i-th check node collects the (k - 1) incoming messages from the other bits $\mathbf{c}_{(i)}$ involved (beside c_t), computes the a posteriori LLR $\lambda_{\phi(\mathbf{c})}$ for the value of their parity bit, and passes this “message” to the t-th variable node. Finally, the t-th variable node computes λ_t according to (2.15) by summing all of the incoming messages and adding $\frac{2}{\sigma^2} r_t \alpha_t$.

The key result is that $\lambda_{i,p}$ can be calculated iteratively, using an equation of the form (2.13). A simplified form of the algorithm for each codeword is given as follows:

1. Compute variable node-to-checknode messages (first half of iteration).

2. Compute checknode-to-variable node messages (second half of iteration).

3. Update APP LLR for all variable nodes.

4. Apply hard decision on codeword bits, and check whether the decoded frame is a valid codeword.

5. If not a valid codeword, repeat steps 1-4 for a number of iterations.

6. Stop if the codeword is valid or the maximum number of iterations reached.

Figure 2.4 shows the effect of the maximum decoder iterations for LDPC codes. It can be seen that increasing the maximum number of decoder iterations leads to an improvement in the average performance. It can also be seen that when iterations are set to more than 5, the performance gain achieved by increasing the number of maximum iterations reduces with the increase in the iterations. Increasing the iterations from 2 to 5 leads to a gain of 1.5 dB at BER of 10^{-4}, while increasing the iterations from 5 to 10 leads to a gain of 0.33 dB at BER of 10^{-4}.

The performance curves for LDPC codes can be categorized into two regions: the *waterfall* and *error floor* regions, as shown in Figure 2.5. Decoding failures occur when the LDPC decoder fails to converge to a valid codeword. These errors occur prevalently at lower SNR where a rapid improvement in the LDPC code performance is observed (waterfall region). The suboptimality of the LDPC decoer is caused by the presence of small loops. The LDPC decoder operates optimally at high SNR region, since the reduction in the average number of decoder iterations required to converge reduces the effect of loops. Maximum Likelihood decoding errors occur in this region, and since the Maximum Likelihood performance is limited by the d_{min} of the code, an error floor is observed in the high SNR region.

Density evolution [22] is a technique for tracking the pdfs of the messages in the Tanner graph of an LDPC code, under the assumption that $n \to \infty$. The notion of convergence *threshold* was introduced, which is defined as the SNR value above which the probability of decoding error approaches zero, while for values of SNR lower than threshold the probability of decoding error is nonzero. Using density evolution, given the initial pdf of LLR messages, the pdf of LLR messages at any iteration can be computed. This allows for the design of irregular LDPC codes which perform very close to the Shannon limit using density evolution as a probe, i.e., finding the convergence threshold of different irregular codes by density evolution and choosing the best one. Density evolution requires intensive computations [22]. In [27], the density evolution algorithm was simplified by using the assumption that at each iteration the pdf of the messages is Gaussian.

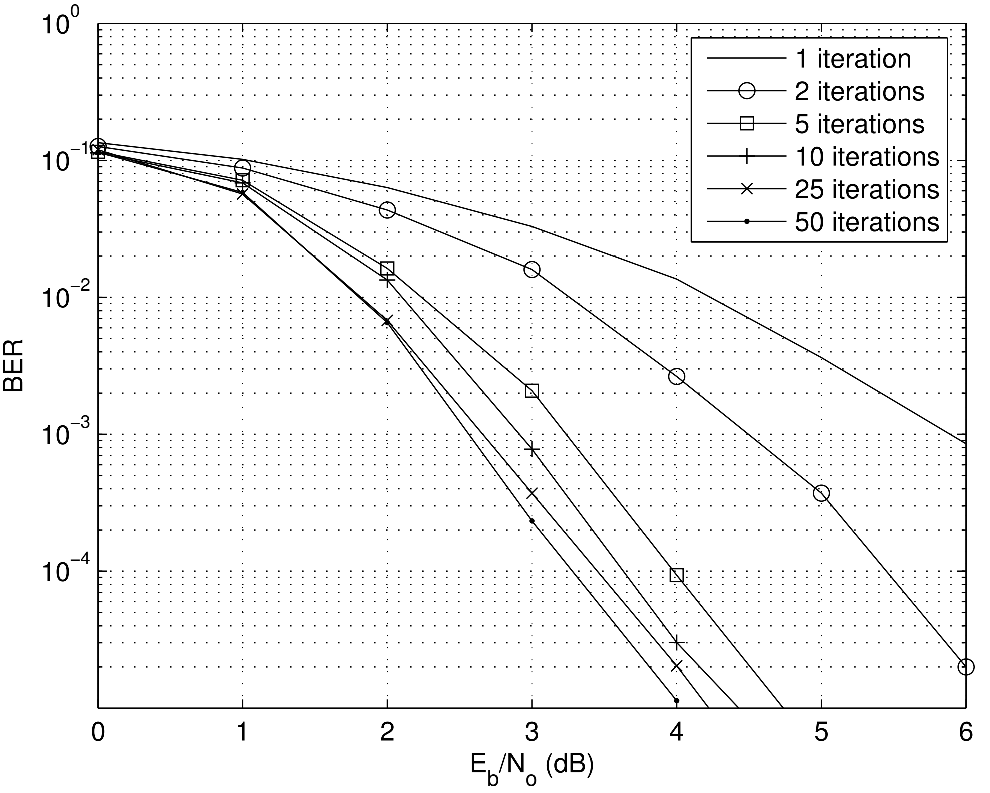


Figure 2.4: Effect of maximum decoder iterations on the performance of LDPC codes, the performance curves shown are for a rate-$\frac{1}{2}$ semi-random LDPC code of girth 4, n=256.

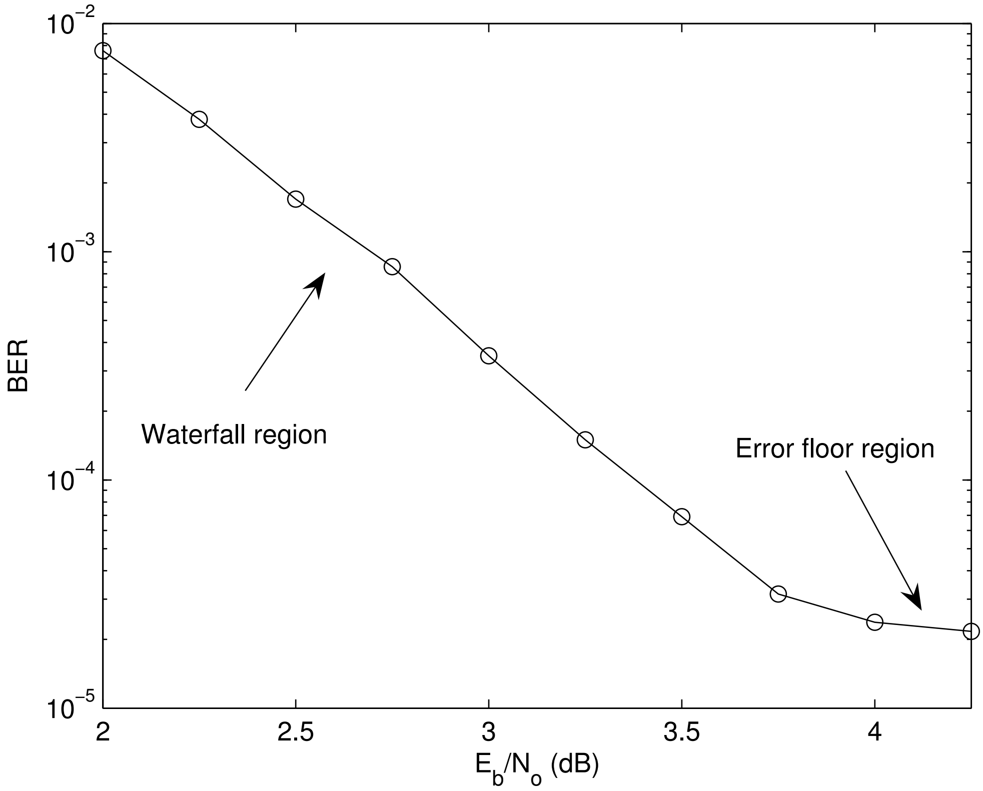


Figure 2.5: Waterfall and error floor regions, the BER shown is for a rate-$\frac{1}{2}$ semi-random LDPC code of girth 4, n=256.

2.3.4 The Decoding Complexity of RC-LDPC Codes

The following table [1] shows the decoding complexity for regular RC-LDPC codes.

Table 2.1: Decoding complexity per iteration for regular RC-LDPC codes.

	puncturing	extending
addition	$3jk/R_o$	$3j'k/R_i$
tanh / $\tanh^{-1}$	$2jk/R_o$	$2j'k/R_i$

In the table, j denotes the column weight of the mother code, j' denotes the column weight of the extended code, R_o denotes the code rate of the mother code, and R_i denotes the code rate of the code obtained after extending the mother code by i levels, given by $R_i = k/\left(n_o + \sum\limits_{v=1}^{i} M_v\right)$, where n_o is the block length of the mother code, and M is the number of parity bits added after one level of extension.

From the table it can be seen that for punctured regular LDPC codes, the decoding complexity is constant, since the mother code is used for decoding of the punctured codes. This statement can be generalized to irregular LDPC codes. This can be explained as follows: for the purpose of decoding, the punctured variable nodes are initialized with erasures, while the non-punctured variable nodes are initialzed with LLRs corresponding to the respective channel observations. Therefore, for any punctured regular or irregular LDPC code, the decoding complexity remains constant since the total number of variable nodes, check nodes and edges in the parity-check matrix remain constant. However, the decoding complexity of extended LDPC codes increases with the levels of extension, since the number of variable nodes, check nodes and edges in the parity-check matrix increase.

2.4 LDPC Code Design Approaches

The construction of an LDPC code is achieved by constructing of a low-density parity-check matrix with prescribed properties. A large number of design techniques exist in the literature, and we introduce some of the more prominent ones in this section.

2.4.1 Regular Codes

Gallager Codes

The original LDPC codes due to Gallager [3] are regular LDPC codes with an H matrix of the form

$$H = \begin{bmatrix} H_1 \\ H_2 \\ \vdots \\ H_{w_c} \end{bmatrix}, \tag{2.16}$$

where the submatrices H_d have the following structure: for any integers μ and w_r greater than 1, each submatrix H_d is $\mu \times \mu w_r$ with row weight w_r and column weight 1. The submatrix H_1 has the following specific form: for $i = 1, \ldots, \mu$ the i^{th} row contains all of its w_r 1's in columns $(i-1)w_r$ to iw_r. The other submatrices are simply column permutations of H_1. It is evident that H is regular and has dimensions $\mu w_c \times \mu w_r$, and has row and column weights w_r and w_c respectively. Gallager [3] showed that the ensemble of such codes has excellent distance properties provided that $w_c \geq 1$ and $w_r > w_c$.

Mackay Codes

Mackay had independently discovered the benefits of binary codes with sparse H matrices. He has proposed in [50] algorithms to generate sparse H matrices. A few of these are listed below in order of increasing algorithm complexity:

1. H is created by randomly generating weight-w_c columns and (as near as possible) uniform row weight.
2. H is created by randomly generating weight-w_c columns, while ensuring weight-w_r rows, and no two columns having overlap greater than 1.
3. H is generated as in algorithm 2, and additionally short cycles are avoided.

2.4.2 Irregular codes

Richardson *et. al.* [49] and Luby *et. al.* [23] defined ensembles of irregular LDPC codes parameterized by the degree polynomials $\lambda(x)$ and $\rho(x)$, defined as

$$\lambda(x) = \sum_{i=2}^{d_l} \lambda_i x^{i-1} \; and \; \rho(x) = \sum_{i=2}^{d_r} \rho_i x^{i-1}, \tag{2.17}$$

where $\lambda_i(x)$ and $\rho_i(x)$ are the fractions of edges belonging to degree-i variable and check nodes, and d_l and d_r are the maximum variable and check node degrees respectively. They showed how to optimize these polynomials for a variety of channels. Optimized in the sense that (assuming message passing decoding) a typical code in the ensemble was capable of reliable communications in worse channel conditions than codes that are outside the ensemble. The worse-case channel condition is called the *decoding threshold* and the optimization of $\lambda(x)$ and $\rho(x)$ is found by a combination of *density evolution* algorithm and an optimization algorithm. The decoding threshold for a given $\lambda(x)$-$\rho(x)$ pair is determined by evaluating the pdf's of the log-likelihood ratios of the code bits. The optimization algorithm optimizes the design of H over the $\lambda(x)$-$\rho(x)$ pairs. In general designs via density evolution are best applied to codes of large block-length since density evolution assumes that $n \to \infty$.

2.4.3 Finite-Geometry Codes

In [65], regular LDPC codes were designed using techniques based on finite-geometries. These LDPC codes fall into the cyclic and quasi-cyclic classes of block codes and lend themselves to simple encoder implementations via shift-register circuits.

2.4.4 RA, IRA and eIRA Codes

A type of code, called a repeat-accumulate code, which has the characteristics of both serial turbo codes and LDPC codes, was proposed in [66]. These codes have been shown to be capable of operation near capacity limits, but they have the drawback that they are naturally low-rate (rate 1/2 or lower).

The RA codes were generalized, yielding *irregular* repeat-accumulate codes [67]. These codes are capable of operation even closer to theoretical limits than RA codes, and they permit higher code rates. A drawback to IRA codes is that they are non-systematic, although they can be put in systematic form at the expense of lowering the rate.

Yang and Ryan [68] have proposed a class of efficiently encodable irregular LDPC codes which are called *extended* IRA (eIRA) codes. For these codes, the encoding can be efficiently performed directly from the H matrix.

Chapter 3

Rate-Compatible Regular LDPC Codes

This chapter is concerned with the design of punctured regular LDPC codes that outperform *randomly* punctured regular LDPC codes. Regular LDPC codes were introduced by Gallager in [3]. A regular (j,l) LDPC code is defined by an $(n\text{-}k) \times n$ parity-check matrix having exactly j ones in each column and exactly l ones in each row, where $j < l$ and both are small compared to n [2]. It was also shown in [3] that for regular codes, the minimum distance increases linearly with block-length if the column weight is greater than or equal to three. Hence the column weight is typically chosen to be greater than or equal to three. The (3,6)-regular LDPC ensemble is the best regular ensemble [49]. Therefore in this work (as in [1]), the regular-(3,6) LDPC ensemble is employed for designing rate-compatible regular

LDPC codes.

Although regular LDPC codes perform close to capacity, they show a larger gap to capacity compared to turbo codes. The main advantage of regular LDPC codes over turbo codes is their lower error floor. The error floor phenomenon arises at high SNR values due to the small minimum distance of turbo codes [14]. Therefore, turbo codes will experience an error floor even under maximum-likelihood decoding [14]. On the other hand, LDPC codes have lower error floors [50]. In addition, it is shown in [50] that LDPC codes can achieve the Shannon limit under optimal decoding. Furthermore, regular LDPC codes have lower error floors as compared to irregular LDPC codes [69].

Rate-Compatible LDPC codes based on (3,6)-regular LDPC codes were designed in [1] by using *random* puncturing in order to obtain codes with rates higher than that of the mother code. However the puncturing pattern(s) chosen were not optimized with respect to any criterion. One method of comparing the performance of LDPC codes with different puncturing patterns would be through exhaustive search using Monte Carlo simulation, which is very complex and time consuming. Therefore approaches that compare different random puncturing patterns without resorting to exhaustive search are required. In the following section a heuristic algorithm for *comparing* random puncturing patterns is presented. We claim that a search employing the proposed algorithm over an ensemble of puncturing patterns will enable the selection of a pattern that outperforms a pattern chosen at random from the ensemble (with high probability).

3.1 The Heuristic Search Algorithm

For a given block length and a given degree distribution of the underlying Tanner graph, the ensemble of short block-length LDPC codes can have considerable variation in performance, specially at high SNR [70]. An efficient heuristic algorithm for finding good LDPC codes based on the *girth distribution* of the Tanner graph was presented in [70]. In this chapter, we use this algorithm to obtain puncturing patterns (needed to design RC-LDPC codes) that result in punctured codes with good performance.

3.1.1 The Girth of a Graph

The *girth* of a LDPC code refers to the length of the shortest loop (or cycle) present in the codes' equivalent Tanner graph. In the absence of loops, the iterative decoding algorithm converges to the maximum-likelihood solution [62]. In [70] this term is used in a wider sense, where the *girth at variable node u* is defined as the length of the shortest cycle that passes through u. The *girth distribution*, $g(l), l = 4, 6, ..., l_{max}$ of a Tanner graph refers to the fraction of the symbol nodes with girth l, where l_{max} is the maximum girth in the graph. The *girth average* for a graph is defined as

$$\sum\nolimits_{k=2}^{l_{\max}/2} g(2k).2k. \tag{3.1}$$

Intuitively, girth distribution is related to the sub-optimality of the iterative decoder. It is well known that for a cycle-free Tanner graph, belief propagation results in optimal decoding [62]. The girth of a symbol node indicates the length of the shortest path, or equivalently the smallest number of iterations, for a message sent by that node to propagate back to the node itself. Before this number of iterations is reached, the "belief" associated with the node is "optimally" propagated to the rest of the graph. To have a performance close to the optimal, it is therefore favorable to make the girth of variable nodes as large as possible, or in other words, to have more symbol nodes with larger girths [70].

The computation of girth at a given node u is carried out as follows: a tree is "grown" step by step starting from the "root" node u. At step k, all the nodes at distance k from u are included into the tree. This procedure is repeated until, at step k, a node connected to at least two nodes included at step k - 1 is included. This identifies the formation of the first cycle. The integer $2k$ is then the girth at node u. The complexity of this algorithm is low and quite manageable for short block lengths [70].

The computation of girth at a given node can also be done by using the *adjacency matrix* [56, 71] of a parity-check matrix and powers of the adjacency matrix. This is the method employed in this work, since it can be done efficiently using matrix manipulation software such as Matlab. Denoting all the nodes of the codes' graph as $v_1, v_2, ..., v_p$, and define the *adjacency matrix* $A = [a_{ij}]$ to be the $p \times p$ symmetric binary matrix

$$a_{ij} = \begin{cases} 1 & \text{if an edge connects } v_i \text{ with } v_j \\ 0 & \text{otherwise} \end{cases}$$

The natural ordering of the nodes for an LDPC graph results in the relationship

$$A = \begin{pmatrix} \mathbf{0} & H \\ H^T & \mathbf{0} \end{pmatrix}, \tag{3.2}$$

For a matrix H of size $(m \times n)$, the girth of a variable node u is l iff

$$A_{ij}^{(l/2)} \geq 2 \quad \text{and } A_{ij}^{(l/2)-2} = 0, \quad \text{for any } i, \, and \, j = u + m. \tag{3.3}$$

In [70], the algorithm based on the girth average of codes was used to compare randomly constructed codes of short block-length. A search was performed over a finite number of randomly generated codes, and it was shown that the code with the highest girth average performed better than all other codes included in the search. As will be shown in the following, punctured codes can also be compared using the heuristic search algorithm based on the *girth average* of the respective punctured codes.

3.1.2 Heuristic Search for Good Puncturing Patterns

The puncturing (removal of parity bits) of a code results in a code of increased code rate and reduced minimum distance [33, 72]. As shown in Figure 3.1, puncturing of a linear block code involves the removal of columns and rows. The rows which are removed correspond to the *nonzero values of the selected columns* [72]. For example, if a certain column to be removed has a 'one' in row number two, then when this column is removed, only row number two can be removed. However, if the column to be removed has ones in rows number two and three, then either row number two *or* row number three can be removed when column this column is removed

The punctured code corresponding to a puncturing pattern would involve the removal of the selected columns and rows corresponding to the nonzero values of the selected columns. For (3,6)-regular codes, each column has three rows which could possibly be removed (since the column weight is 3). Consider the case of puncturing of one parity bit. Three different codes may be obtained since the removal of three different rows (and the same column for each case) results in three different parity-check matrices.

As the number of removed columns increases, the number of codes to be compared increases exponentially, which renders the comparison of punctured codes intractable. Hence the methodology employed for obtaining codes corresponding to different puncturing patterns is as follows: the columns corresponding to the punctured bits are removed from the H matrix, while *none of the rows are removed.* The punctured codes thus obtained are then compared using the *girth average* criteria.

Removing only the punctured columns (and none of the rows) leaves a larger number of ones in the matrix for which the average girth is being computed, as compared to the actual punctured matrix (which has both columns and rows re-

$$\begin{array}{cc}
 & \begin{array}{cccccccc} \textit{punctured column} & & & & & & & \\ \downarrow & & & & & & & \end{array} \\
H = & \left(\begin{array}{cccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{array}\right) \begin{array}{l} \longleftarrow \textit{removed row} \\ \\ \\ \\ \end{array}
\end{array}$$

$$\Big\downarrow \; \textit{Puncturing}$$

$$H = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Figure 3.1: Puncturing of a linear block code (the first column - from the left - and first row are removed).

moved). More ones mean more cycles and probabilistically speaking more small cycles. Therefore it is possible that the girth of a node being computed will come out to be smaller than it would be for the "actual" punctured matrix. Therefore the value of average girth will be smaller than that of the "actual" punctured matrix (this is what is observed through simulation). The important question is: how does this approach affect the comparison between different puncturing patterns? The answer to this question is two-fold:

- The number of ones removed from the original parity-check matrix for each puncturing pattern is the same (since for regular codes the number of ones removed = column weight × number of bits punctured). Therefore probabilistically speaking, the number of variable nodes whose value of girth is affected by this method is the same for all puncturing patterns. This would be supported by the random construction of the code. Therefore if the "actual" punctured code would give value of mean girth of x and this method gives $x - y$, then the difference caused by this method (which is y) would be the same for all patterns with high probability.

- It is easy to see that obtaining the "actual" punctured matrix is viable only for systematic H matrices. For nonsystematic random matrices, the possibilities for the punctured matrices are very large, and would increase with column weight and number of bits punctured.

3.1.3 Simulation Results

To verify the algorithm based on the girth average criteria, a search was performed over 500 random puncturing patterns (for a (3,6)-regular LDPC code of a block-length 256 with 64 parity bits being punctured). Figure 3.2 shows the performance over the AWGN channel of two puncturing patterns, one having the maxiumum girth average and the other having the minimum girth average. There is a significant difference in performance at high SNR, since large girth average leads to large d_{min}.. For low SNR values, the girth has little effect on performance, since the subtleties of the loops' effects on belief propagation are irrelevant when the noise level is high [56]. The puncturing pattern with the minimum girth average yields a code which has a higher error floor as compared to the code resulting from the puncturing pattern with the highest girth average (there is 1 dB difference between the two curves at BER of 10^{-6}). It is therefore clear that the girth average criteria and the method for comparing puncturing patterns is a viable method for selecting good puncturing patterns from an ensemble of random puncturing patterns.

3.2 Rate-Compatible Punctured Codes

In this section, we present the RC-LDPC obtained using the heuristic search method defined in Section 3.1. Consider a range of desired rates $R_1 > R_2 > ... > R_J > R$, where R is the code rate of the mother code. Rate-compatible codes that utilize puncturing patterns selected using the heuristic search criteria can be obtained by the following method:

1. A code with rate R_j is obtained by puncturing p_j bits from the mother code with rate R.

2. Select the number of punctured parity bits p_1 which yield a rate-R_1 code. Generate random patterns of size p_1. Perform heuristic search to obtain the best puncturing pattern corresponding to p_1 punctured bits.

3. Puncturing patterns for obtaining the rate-R_2 can be obtained by selecting random subsets of size p_2 from the pattern selected in the previous step (of size p_1), and performing heuristic search over them.

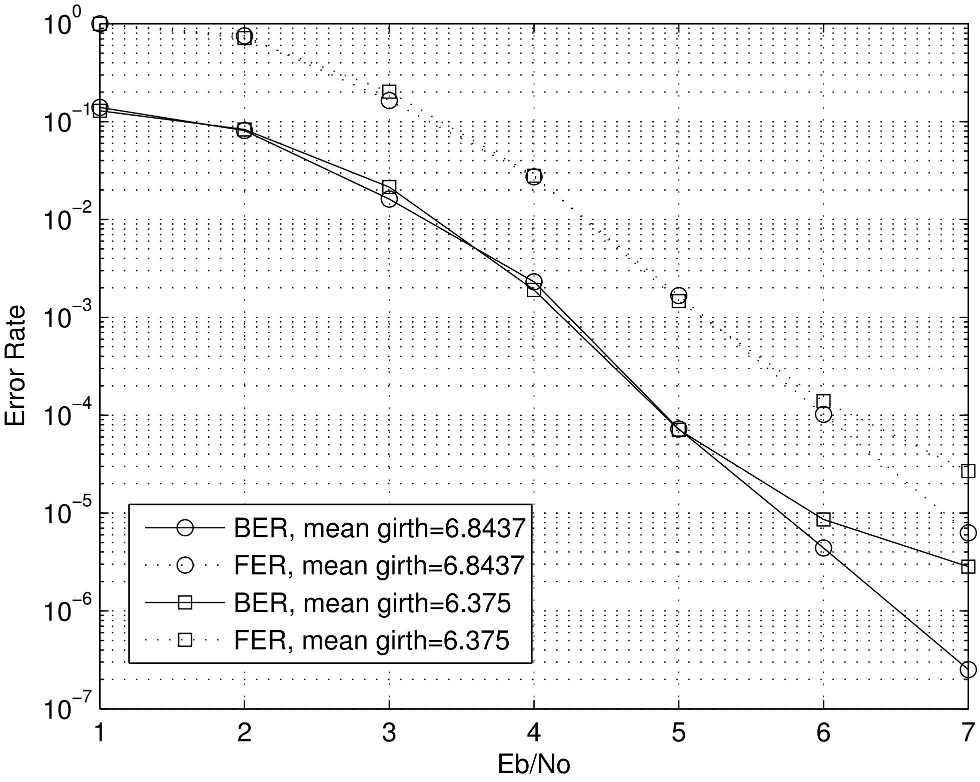


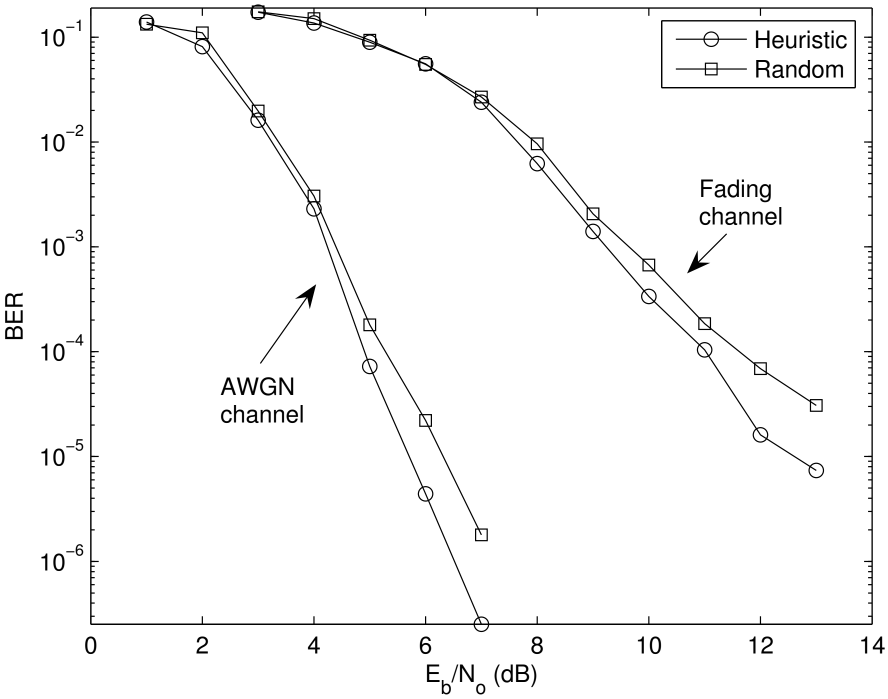
Figure 3.2: Performance of regular LDPC codes with two different puncturing patterns that result in the maximum and the minimum girth average, n=256 and 64 parity bits being punctured.

4. The puncturing patterns required to obtain the codes with rates $R_3, .., R_J$ may be obtained in a similar manner as in step 3.

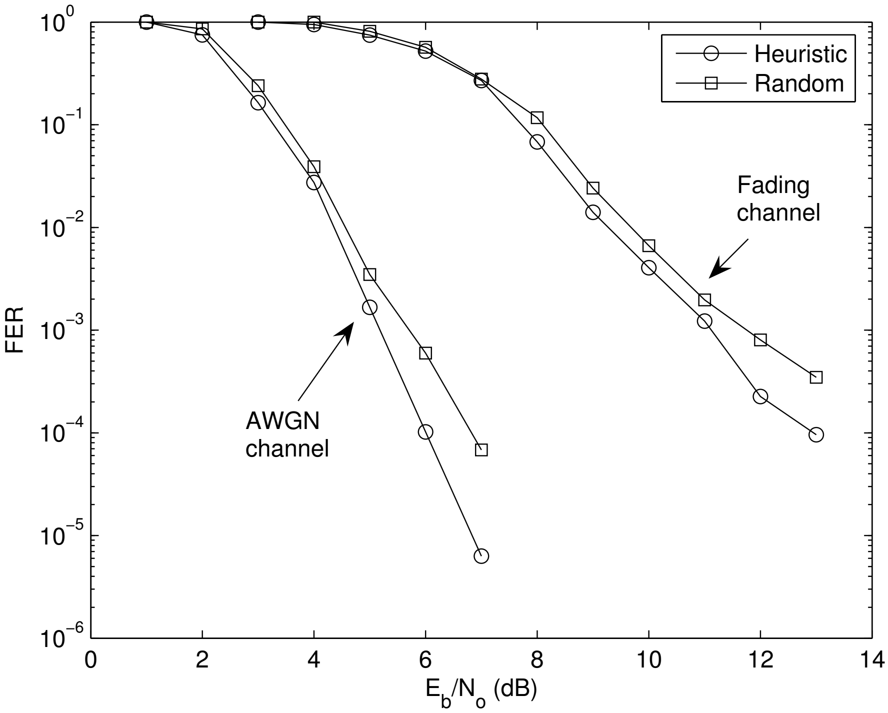
The following figures show the performance of rate-compatible regular LDPC codes of arbitratily selected rates 0.66 and 0.62 when simulated over the AWGN and uncorrelated Rayleigh fading channels. For each code rate, heuristic search was peformed over 500 random patterns. The mother code is a rate-$\frac{1}{2}$ code of block-length 256. The maximum number of decoder iterations is set to 25. In the figures, "Heuristic" denotes the performance of a punctured code obtained by puncturing with a pattern that was selected using the heuristic search algorithm, and "Random" denotes the performance averaged over many random puncturing patterns (during simulation the random pattern was changed after every 100 codewords transmitted over the respective channel). It can be seen that codes punctured according to the heuristic search algorithm perform better than codes punctured randomly, and the difference in performance increases with increasing SNR. This is due to the larger impact of the girth on the performance at high SNR than it is at low SNR: the d_{min} of a code can be increased by increasing its girth, and hence codes with larger girth have lower error floors [61].

Chapter Summary

In this chapter an algorithm has been presented for the selection of puncturing patterns from an ensemble of random puncturing patterns. The algorithm is based on the concept of girth. Since increasing the girth of a LDPC code leads to improved performance, the algorithm selects puncturing patterns that result in punctured codes with large girth. Simulation results verify that the puncturing patterns selected using this algorithm outperform randomly punctured codes.

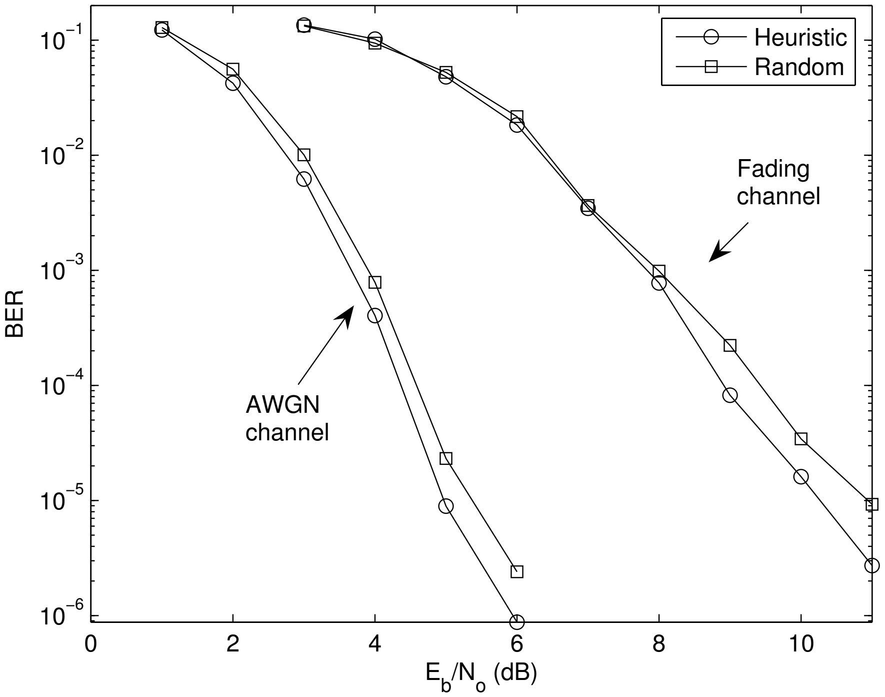


(a)

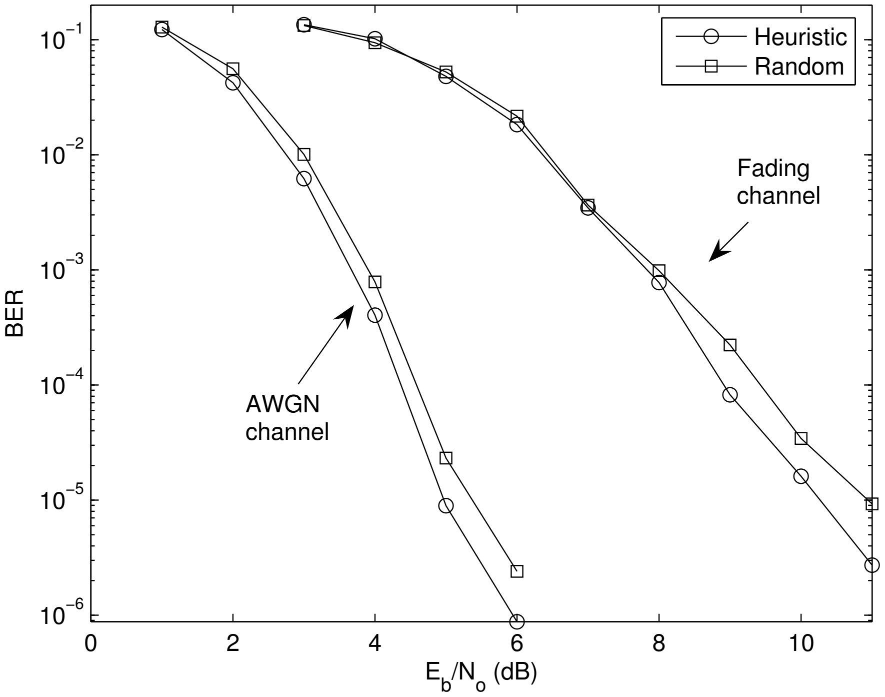


(b)

Figure 3.3: Performance of punctured regular codes of rate 0.66 over the AWGN and Rayleigh fading channels (a) BER (b) FER



(a)



(b)

Figure 3.4: Performance of punctured regular codes of rate 0.62 over the AWGN and Rayleigh fading channels (a) BER (b) FER

Chapter 4

Rate-Compatible Semi-Random LDPC Codes

Semi-random LDPC codes offer several advantages including efficient encoding and good performance. In addition, it will be shown in this chapter that they have several properties that make them good candidates for use in systems employing rate-compatible codes. In this chapter we consider the design of rate-compatible semi-random LDPC codes through puncturing and/or extending.

4.1 Punctured Codes

Puncturing constructs high-rate codes from low-rate codes by deleting parity bits [33]. The transmitter does not transmit the punctured parity bits. For the decoding of a punctured LDPC code, the decoder inserts erasures where the parity bits are punctured and performs the decoding algorithm as in a non-punctured case [1].

Figure 4.1 shows the structure of a semi-random parity-check matrix. The submatrix on the left corresponds to the parity bits, and the submatrix on the right corresponds to the data bits. To obtain punctured semi-random codes, the variable nodes corresponding to the punctured parity bits are replaced with erasures at the decoder. Figure 4.2 shows the location of decoder erasures corresponding to different puncturing patterns that could be used at the decoder. In the figure, “alternate” refers to puncturing of alternate parity bits, “successive” refers to puncturing of successive parity bits and “random” refers to puncturing of a random pattern of parity bits.

$$H = \left(\begin{array}{c|c} \underbrace{\begin{matrix} 1 & & & 0 \\ 1 & 1 & & \\ & 1 & 1 & \\ 0 & & 1 & 1 \end{matrix}}_{Parity} & \underbrace{\begin{matrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{matrix}}_{Data} \end{array}\right).$$

Figure 4.1: A Semi-Random parity-check matrix

Figure 4.3 shows the message passing during decoding of punctured codes, It can be seen that for the case of successive or random puncturing, a successive group of parity bits may be erased. As the decoding proceeds iteratively, after the first (and successive) iteration(s) the LLR values of the punctured parity bits for the case of alternate puncturing will be larger than the LLR values of punctured parity bits for the case of successive or random puncturing. A large LLR magnitude for a variable node implies higher reliability as compared to the reliability of a variable node with a smaller LLR magnitude [58]. Since the LLRs of the punctured parity bits for the case of alternate puncturing converge in a smaller number of iterations as compared to the other two puncturing schemes, the LLRs of information bits will also converge to their correct values (in the probabilistic sense) in a smaller number of iterations. The evolution of LLR magnitudes of the punctured bits for the different puncturing shemes is illustrated in Figure 4.4, which confirms that parity bits punctured "alternately" will converge in fewer iterations as compared to other puncturing schemes. Therefore it is expected that the performance of semi-random LDPC codes that are punctured "alternately" will perform better than other puncturig schemes. This is confirmed by the simulation results shown in Figure 4.5.

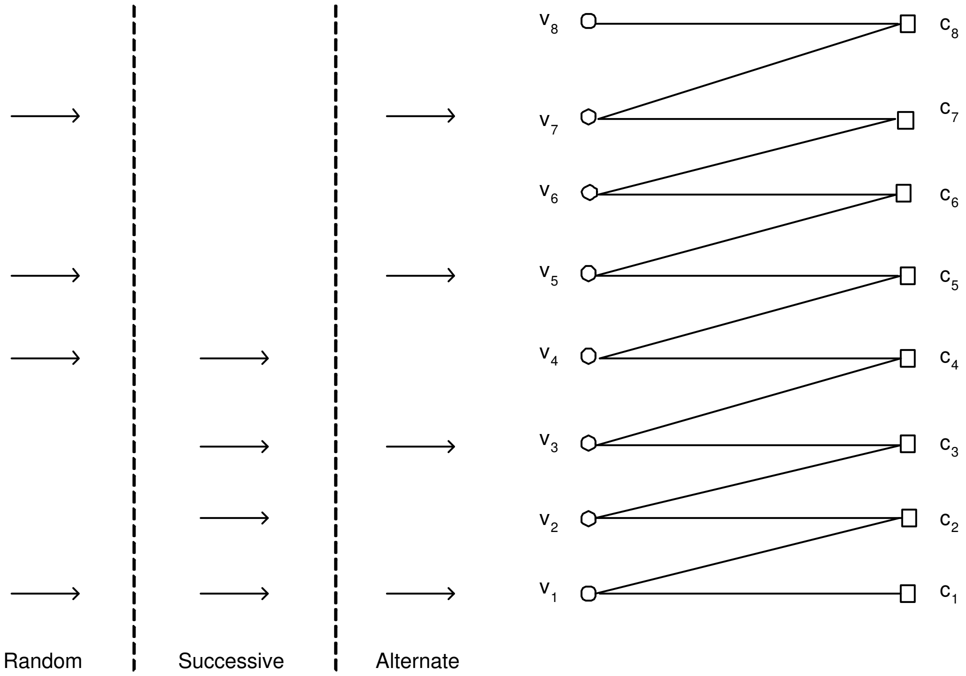


Figure 4.2: Location of decoder erasures corresponding to different puncturing patterns.

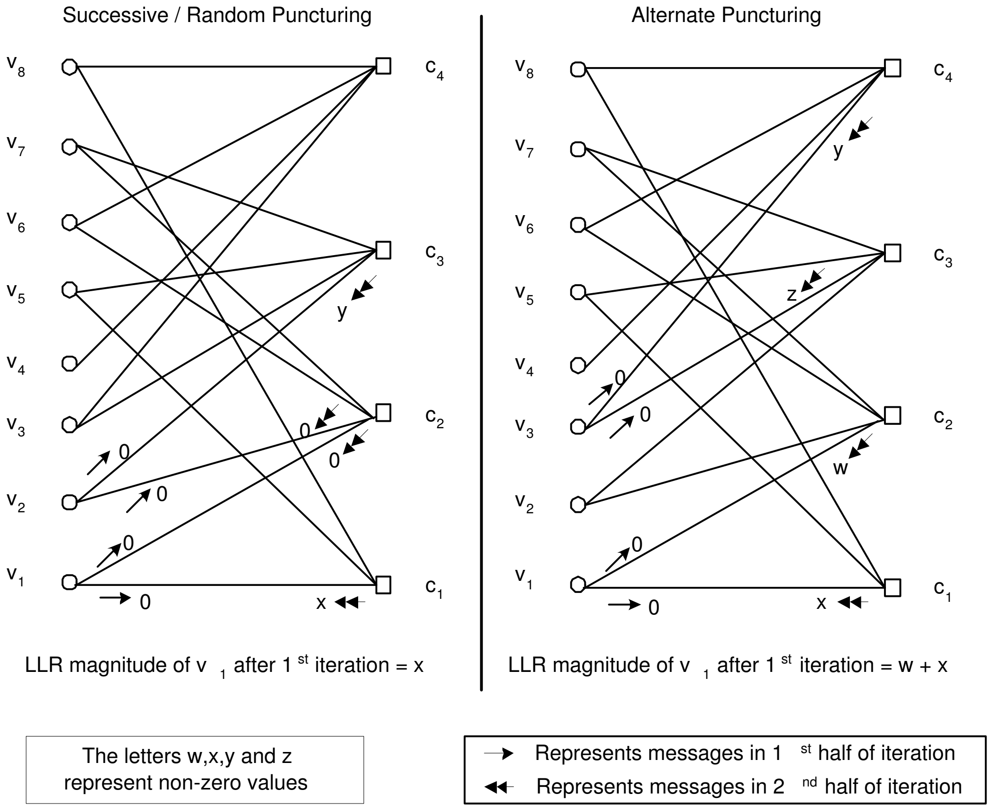

Successive / Random Puncturing
Alternate Puncturing
v8
v7
v6
v5
v4
v3
v2
v1
c4
c3
c2
c1
0
y
x
z
w
LLR magnitude of v 1 after 1 st iteration = x
LLR magnitude of v 1 after 1 st iteration = w + x
The letters w,x,y and z represent non-zero values
Represents messages in 1 st half of iteration
Represents messages in 2 nd half of iteration

Figure 4.3: Message-passing during decoding of punctured semi-random LDPC code corresponding to H matrix of Figure 2.3.

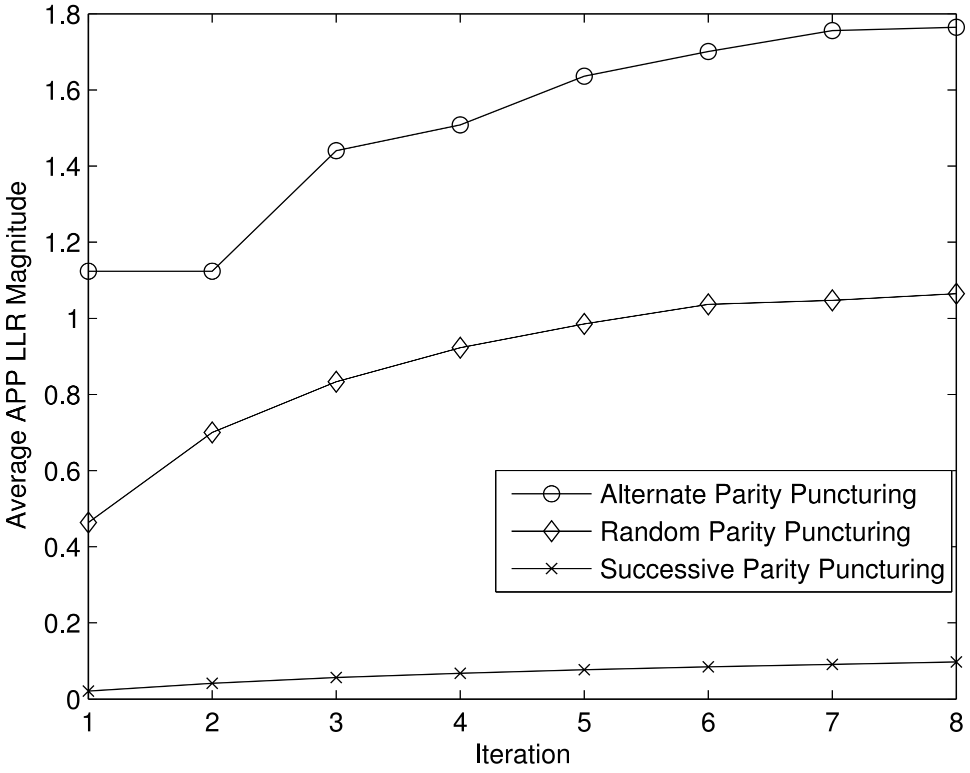
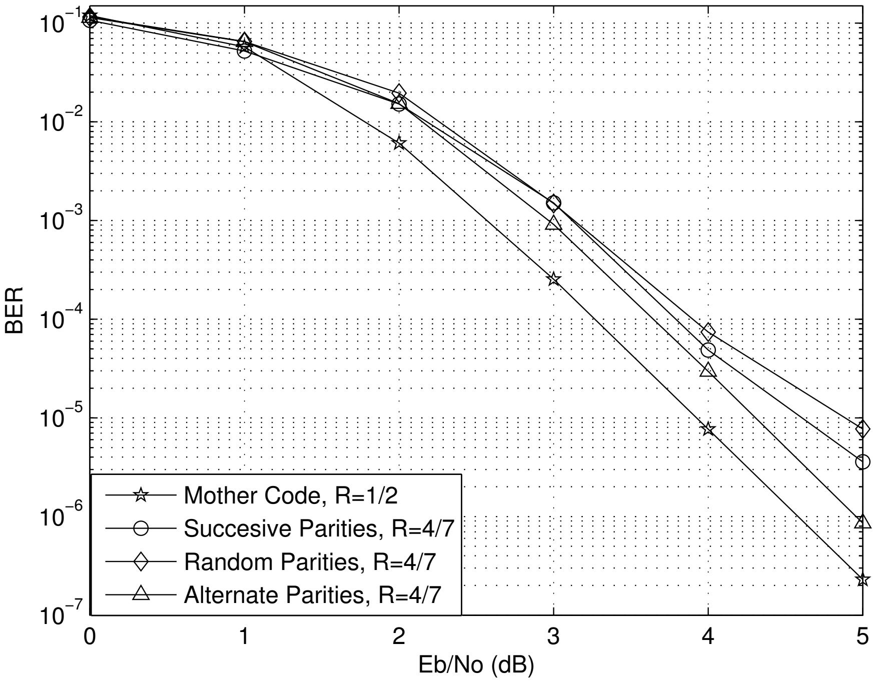
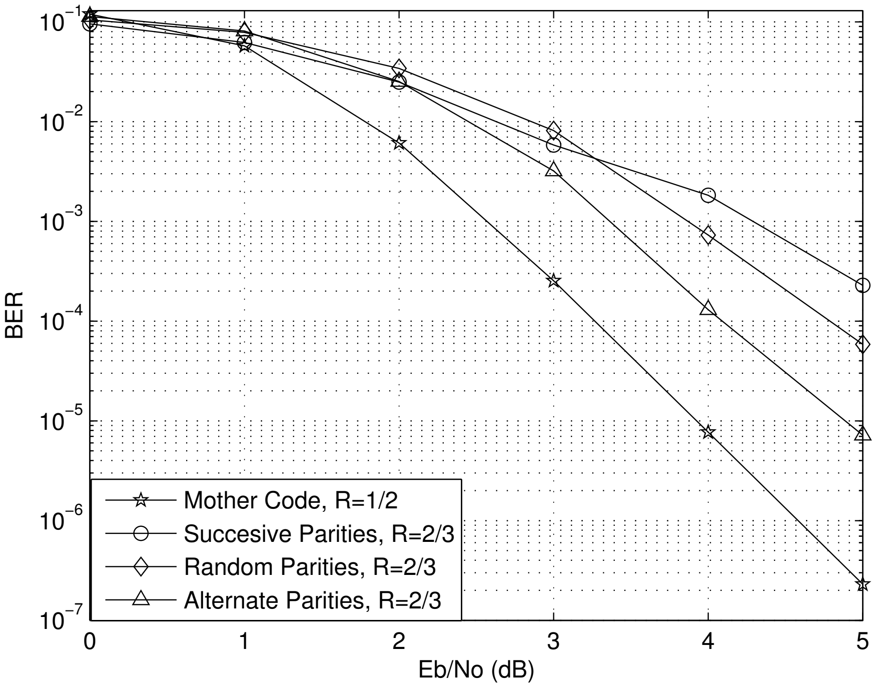


Figure 4.4: Evolution of LLR Magnitudes of Punctured Variable Nodes for semi-random LDPC Code, (Mother Code: n=256, R=1/2, Punctured Code: n=192, R=2/3) at E_b/N_0 = 2.5 dB.



(a)



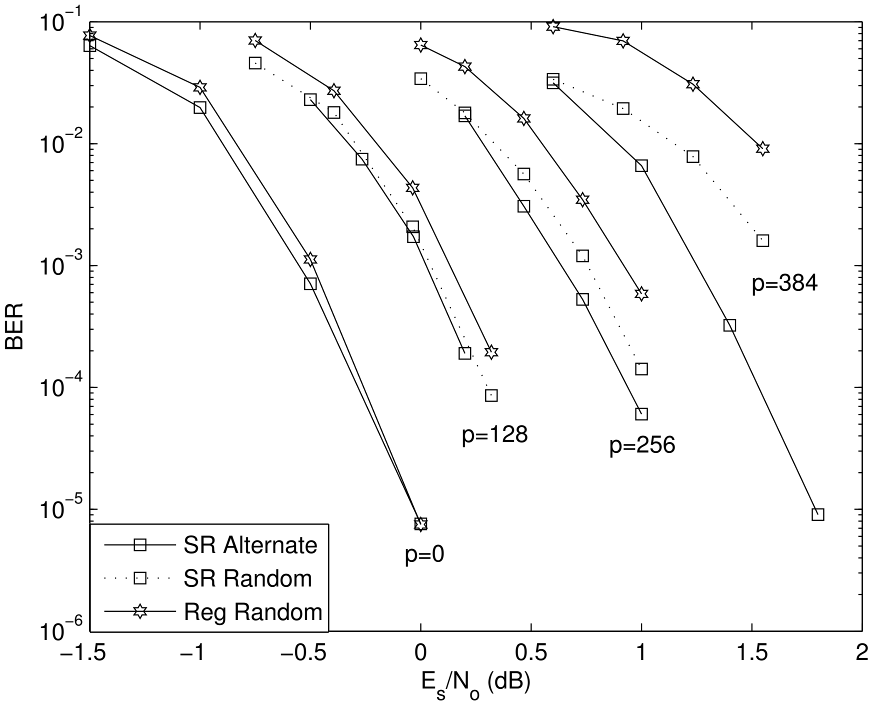
(b)

Figure 4.5: The performance of punctured semi-random LDPC codes, n=256, R=1/2, max-iterations=50 (a) BER of rate-4/7 codes, (B) BER of rate-2/3 codes

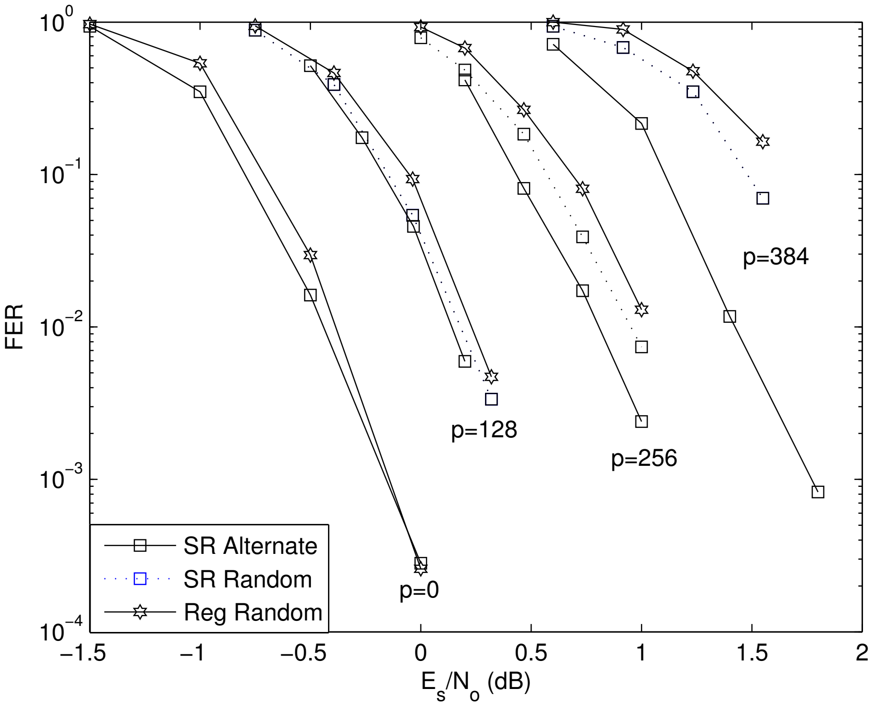
Simulation Results

Figures 4.6 and 4.7 show the performance of punctured semi-random and regular-(3,6) codes over the AWGN and uncorrelated Rayleigh fading channels. The block length n for the mother code and code rates obtained through puncturing correspond to those used in [1]. For the semi-random codes, results for the alternate and random puncturing patterns are shown, whereas for regular codes the results are shown for the random puncturing pattern. In the figures 'p' denotes the number of punctured parity bits, 'SR' denotes semi-random, 'Reg' denotes regular-(3,6), 'Alternate' denotes alternate puncturing pattern and 'Random' denotes a random pattern of parity bits. 'p=0' denotes the mother (non-punctured) code.

From the figures it can be seen that while the mother codes for both the regular-(3,6) and the semi-random codes give similar performance, the performance of the punctured regular (3,6) code is worse that that of the punctured semi-random code, and the difference in performance increases with increasing number of punctured parity bits. This difference in the performance can be explained as follows: for the semi-random matrix, the punctured bits correspond to the parity bits which are of degree two, while for the regular-(3,6) code the punctured bits are of degree three. While one punctured bit of the semi-random code affects two check nodes, the puncturing of one bit of the regular-(3,6) code affects three check nodes. Therefore, semi-random rate-compatible codes obtained via puncturing outperform regular rate-compatible codes obtained via puncturing, when the mother codes of both have similar performance. Furthermore, the performance gain of alternately punctured semi-random codes over the randomly punctured semi-random codes increases with increasing the the number of punctured parity bits. This is due to the reason illustrated in Figure 4.3, where the effect becomes more severe as the number of punctured bits increase.

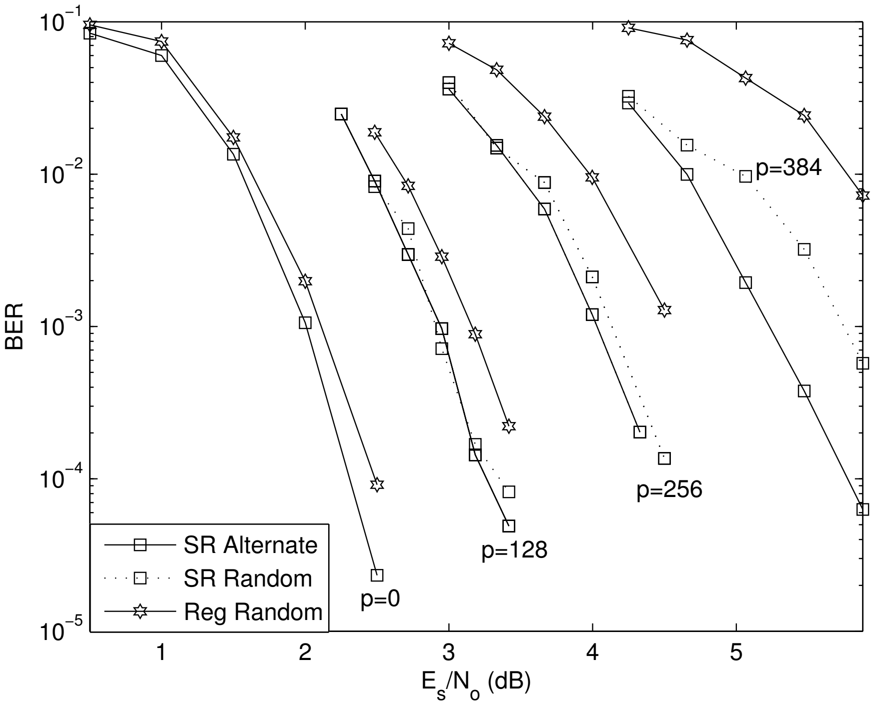


(a)

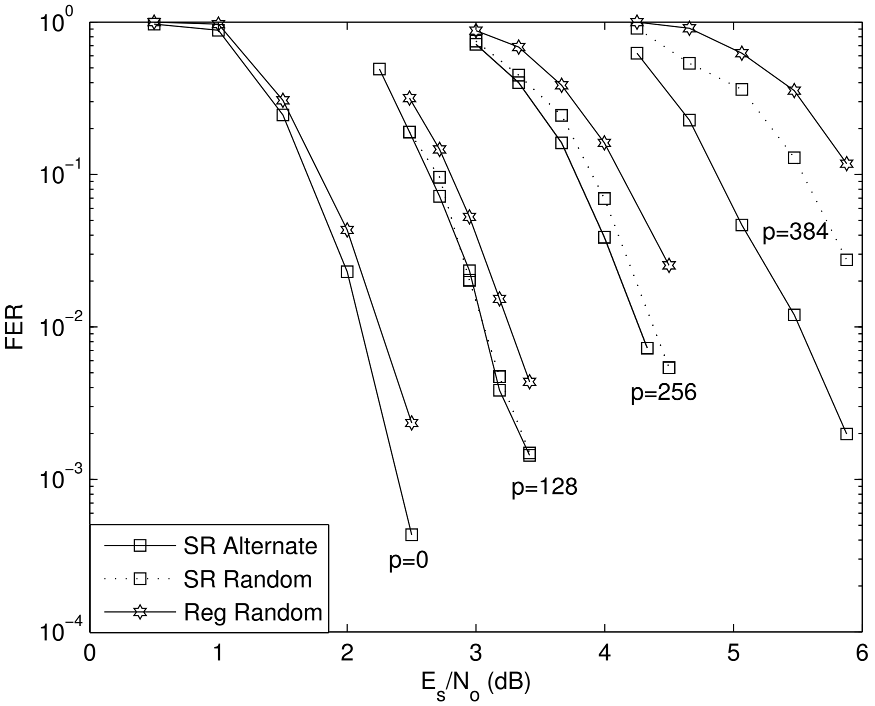


(b)

Figure 4.6: Performance of punctured semi-random and regular codes over the AWGN channel with rates - from left to right - 8/14, 8/13, 8/12 and 8/11, mother code is of rate 8/14 with n=1792, 'SR' denotes semi-random, 'Reg' denotes regular, max-iterations = 50, (a) BER (b) FER



(a)



(b)

Figure 4.7: Performance of punctured semi-random and regular codes over the Rayleigh fading channel with rates - from left to right - 8/14, 8/13, 8/12 and 8/11, mother code is of rate 8/14 with n=1792, 'SR' denotes semi-random, 'Reg' denotes regular, max-iterations = 50, (a) BER (b) FER

4.2 Extended Codes

Extending constructs low-rate codes from high-rate codes by adding parity bits. At the decoder, the lowest rate code is used for decoding. To decode the higher rate codes, missing parity bits are replaced by erasures [5]. For a matrix H of size $(m \times n)$ representing the original code, each level of extension of size u (adding u parity bits) will add u additional rows and u addtional columns to H. The extended matrix H_{ext} will be of size $(m+u) \times (n+u)$. The current literature does not contain methods for the extension of semi-random codes. Figure 4.8 shows the structure of a semi-random parity-check matrix, and the general procedure followed in extending the code. For one level of extension, the following steps have to be implemented:

- Generate $H_{p_{new}}$ of size $(m + u) \times (m + u)$, with the particular "staircase" structure that is required for the deterministic part of a semi-random parity-check matrix (see Figure 2.3).

- Generate $H_{d_{new}}$ which is a concatenation of two matrices: H_d from the unextended matrix, and a matrix H_{sparse} of size $(u) \times (n - m)$.

- The concatenation of $H_{p_{new}}$ and $H_{d_{new}}$ yields a semi-random parity-check matrix with one level of extension (see Figure 4.8).

H_{sparse} is the only part of the extended matrix that can be designed using different methods, since the rest of the matrix is either identical to parts of the unextended matrix, or follows a deterministic construction approach to comply with the standard format of a semi-random parity-check matrix. We have investigated two schemes for the design of the "H_{sparse}" matrix. One scheme is reffered to as the "Extended-Identity" approach since it involves the use of Identity matrices, and the other scheme is called the "Extended-Permuted" approach since it involves the use of a matrix which is a random permutation of a particular matrix. The two methods are discussed in the following sections.

4.2.1 The Extended-Identity Approach

In this approach, H_{sparse} is formed for each level of extension by the concatenation of an identity matrix of size $u \times u$ and matrix of zeros of size $u \times (n - m - u)$, where the size of the non-extended matrix is $m \times n$. Figure 4.9 shows the extension of a semi-random parity-check matrix using this approach for two levels of extension. This method offers a simple and deterministic method of extending a semi-random parity check-matrix. It maintains the sparseness of the extended matrix, and does not lead to the creation of small loops.

$$H = \left(\begin{array}{c|c} \underbrace{\begin{matrix}1&0&0\\1&1&0\\0&1&1\end{matrix}}_{H_p} & \underbrace{\begin{matrix}1&0&1\\0&1&0\\1&1&0\end{matrix}}_{H_d}\end{array}\right)$$

$$\Big\downarrow$$

$$H_{ext} = \left(\begin{array}{c|c} \underbrace{\begin{matrix}1&0&0&0&0\\1&1&0&0&0\\0&1&1&0&0\\0&0&1&1&0\\0&0&0&1&1\end{matrix}}_{H_{p_{new}}} & \underbrace{\overset{\overset{H_d}{\big\downarrow}}{\boxed{\begin{array}{ccc}1&0&1\\0&1&0\\1&1&0\\\hline 0&1&0\\0&0&1\end{array}}}}_{H_{d_{new}}}\end{array}\right) \longrightarrow H_{sparse}$$

Figure 4.8: Extending of semi-random LDPC codes

$$\begin{tikzcd}
H = (H_p \mid H_d) \arrow[d] \\
H^1_{ext} = \left(\underbrace{\begin{array}{ccc} 1 & & \\ 1 & 1 & \\ & 1 & 1 \end{array}}_{H^1_{p_{new}}} \middle| \underbrace{\begin{array}{ccc} & H_d & \\ & & \\ I & 0 & 0 \end{array}}_{H^1_{d_{new}}} \right) \arrow[d] \\
H^2_{ext} = \left(\underbrace{\begin{array}{cccccc} 1 & & & & & \\ 1 & 1 & & & & \\ & 1 & 1 & & & \\ & & 1 & 1 & & \\ & & & 1 & 1 & \end{array}}_{H^2_{p_{new}}} \middle| \underbrace{\begin{array}{ccc} & & \\ & H_d & \\ & & \\ I & 0 & 0 \\ 0 & I & 0 \end{array}}_{H^2_{d_{new}}} \right)
\end{tikzcd}$$

Figure 4.9: Extension of semi-random matrices using the Extended-Identity approach, H is the original non-extended matrix, H^1_{ext} and H^2_{ext} are the extended matrices after one and two levels of extension respectively.

4.2.2 The Extended-Permuted Approach

Another approach for constructing H_{sparse} is to use the permutation of a particular matrix H_{perm} as the matrix H_{sparse} for each level of extension. This approach follows the method of constructing regular codes introduced by Gallager [3], where a parity-check matrix of an LDPC code is formed by the vertical concatenation of a number of submatrices. An example of the matrix H_{perm} which gives good empirical results is shown in Figure 4.10. We can see that it contains two ones in each row. H_{sparse} for each level of extension corresponds to a particular column permutation of H_{perm}. We have also tested extension with an H_{perm} that contains four ones in each row, but the performance is very similar to the H_{perm} with two ones in each row, and therefore the latter was preferred since it enables lower encoding and decoding complexity.

$$H_{perm} = \begin{pmatrix} 1 & 1 & 0 & 0 & . & . \\ 0 & 1 & 1 & 0 & . & . \\ 0 & 0 & 1 & 1 & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \end{pmatrix}$$

Figure 4.10: Extended permuted approach

Simulation Results

Figures 4.12 and 4.13 show the performance of extended semi-random codes over AWGN and Rayleigh fading channels using the "Extended-Identity" approach. Figures 4.14 and 4.15 show the performance of extended semi-random codes using the "Extended-Permuted" approach. In the figures 'ext' denotes the number of parity

$$H = (H_p \mid H_d)$$

$$\big\downarrow$$

$$H^1_{ext} = \left(\begin{array}{c|c} \underbrace{\begin{matrix}1 & & \\ 1 & 1 & \\ & 1 & 1\end{matrix}}_{H^1_{p_{new}}} & \underbrace{\begin{matrix}H_d \\ \\ H^1_{perm}\end{matrix}}_{H^1_{d_{new}}}\end{array}\right)$$

$$\big\downarrow$$

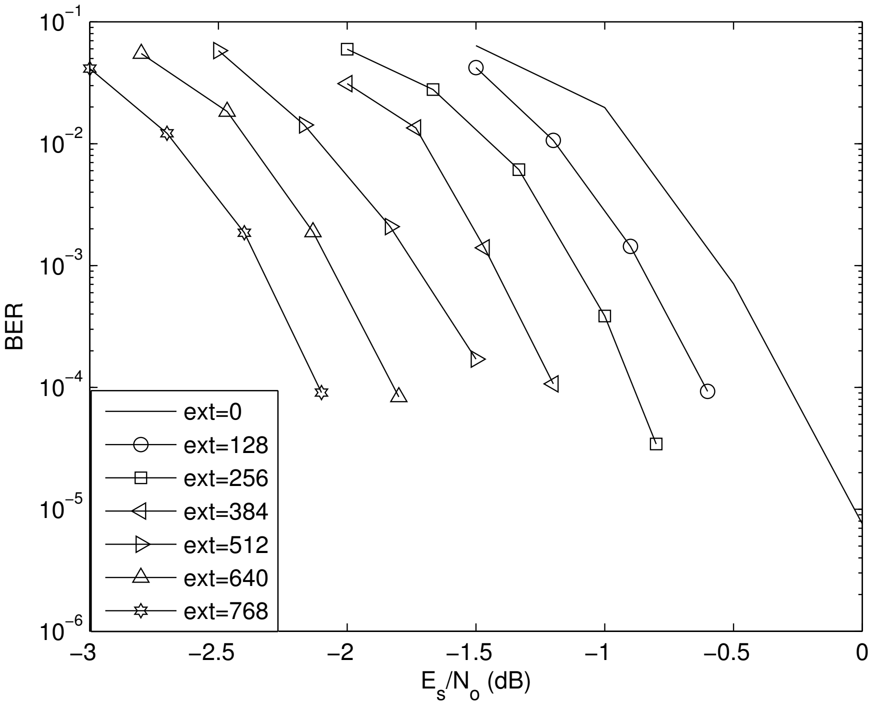
$$H^2_{ext} = \left(\begin{array}{c|c} \underbrace{\begin{matrix}1 & & & & \\ 1 & 1 & & & \\ & 1 & 1 & & \\ & & 1 & 1 & \\ & & & 1 & 1\end{matrix}}_{H^2_{p_{new}}} & \underbrace{\begin{matrix} \\ H_d \\ H^1_{perm} \\ \\ H^2_{perm}\end{matrix}}_{H^2_{d_{new}}}\end{array}\right)$$

Figure 4.11: Extension of semi-random matrices using the Extended-Permuted approach, H is non-extended matrix, H^1_{ext} and H^2_{ext} are the extended matrices after one and two levels of extension respectively, H^1_{perm} and H^2_{perm} are different column permuations of H_{perm}.

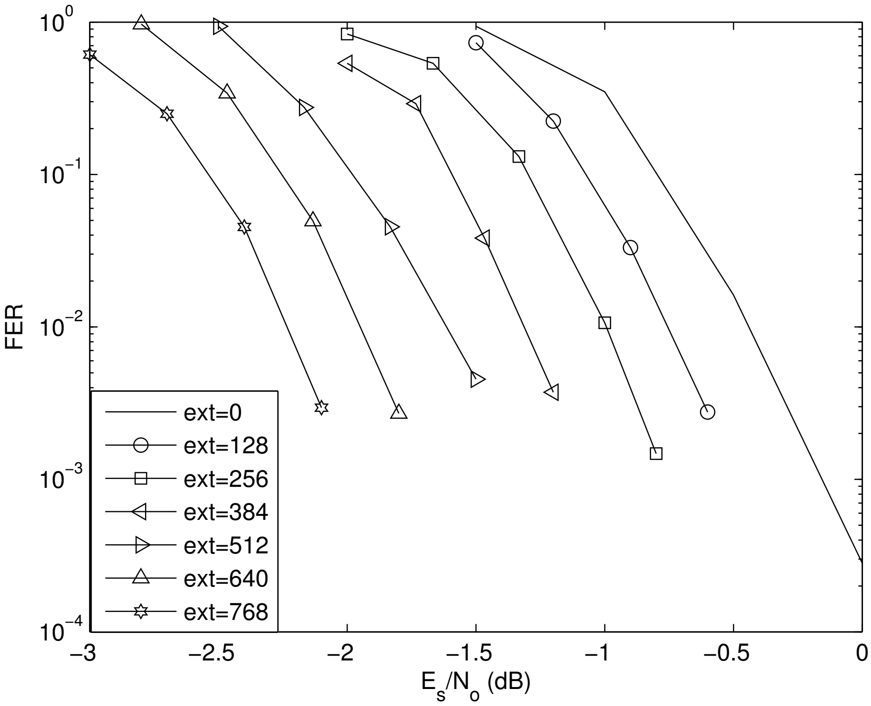
bits added to the mother (non-extended) code. Also, 'ext=0' denotes the mother code.

From the figures we can see that the "Extended-Permuted" approach offers a 0.2-0.3 dB performance advantage over the "Extended-Identity" approach. However the former approach creates four-loops during the extension process, which have to be removed using the loop removal algorithm presented in Chapter 6 and hence the code construction algorithm has higher complexity, while the latter approach has lower complexity because the format for extension is deterministic and the extension does not result in the creation of four-loops.

In addition, difference in performance between the "Extended-Permuted" approach and the "Extended-Identity" approach increases with increasing levels of extension. This can be explained as follows: an extended matrix resulting from extension using the 'Extended-Permuted" approach is relatively denser (the parity-check matrix has more ones) as compared to an extended matrix resulting from extension using the "Extended-Identity" approach. This is due to the fact that "Extended-Identity" approach adds a single one per row in H_{sparse}, while the 'Extended-Permuted" adds two ones per row H_{sparse}. This leads to a slightly larger average column weight for the "Extended-Permuted" approach as compared to the "Extended-Identity" approach. From simulations, it was found that the average column weight of the lowest rate matrices used in the simulations is 3.1 for the "Extended-Identity" approach and 3.4 for the"Extended-Permuted" approach. The difference in the col-

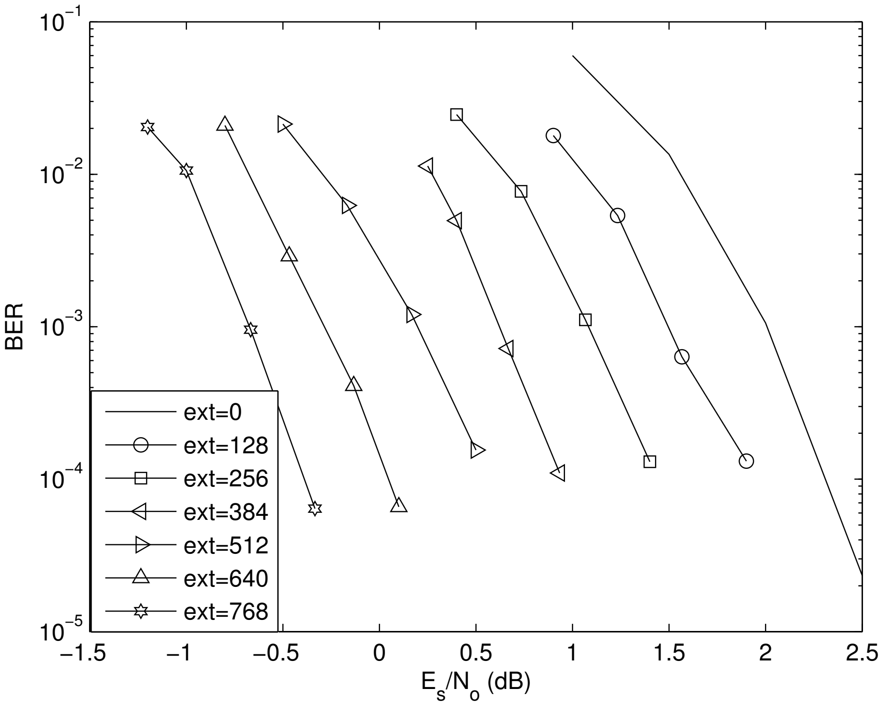


(a)

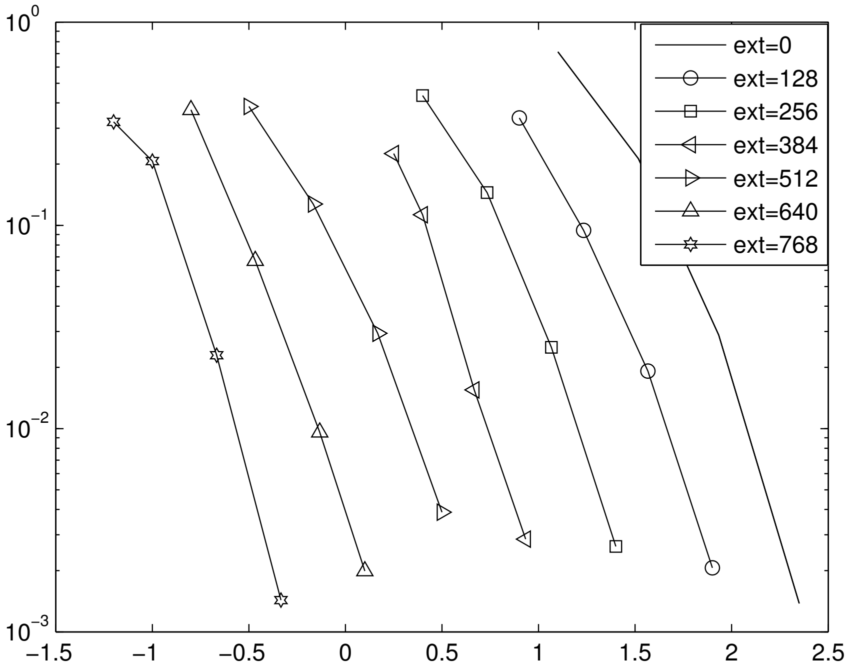


(b)

Figure 4.12: Performance of extended semi-random codes using the Extended-Identity approach over the AWGN channel with rates - from right to left - 8/14,..,8/20, mother code is of rate 8/14 with n=1792 max-iterations = 50, (a) BER (b) FER

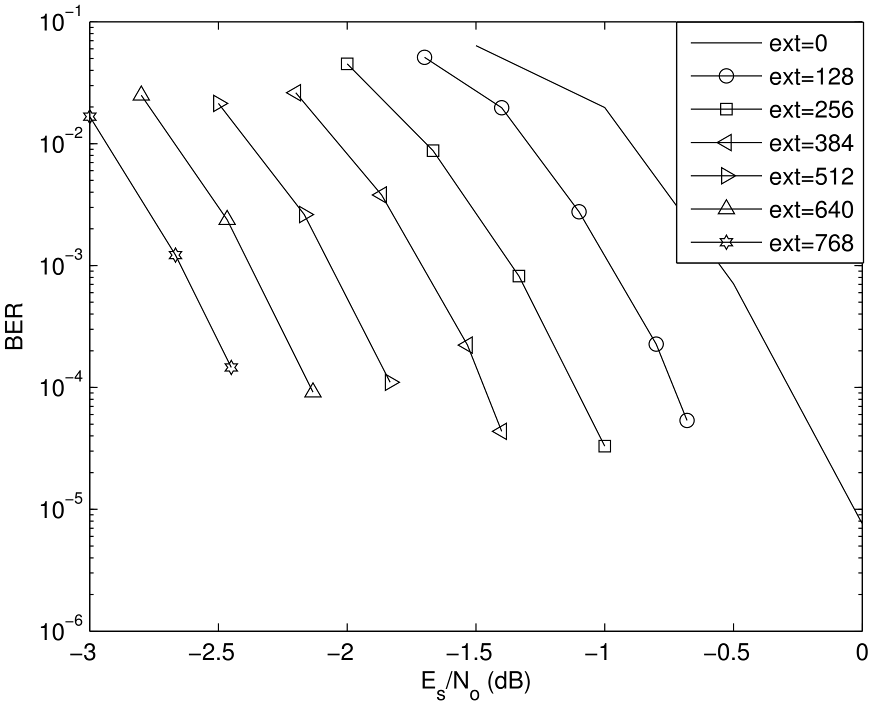


(a)

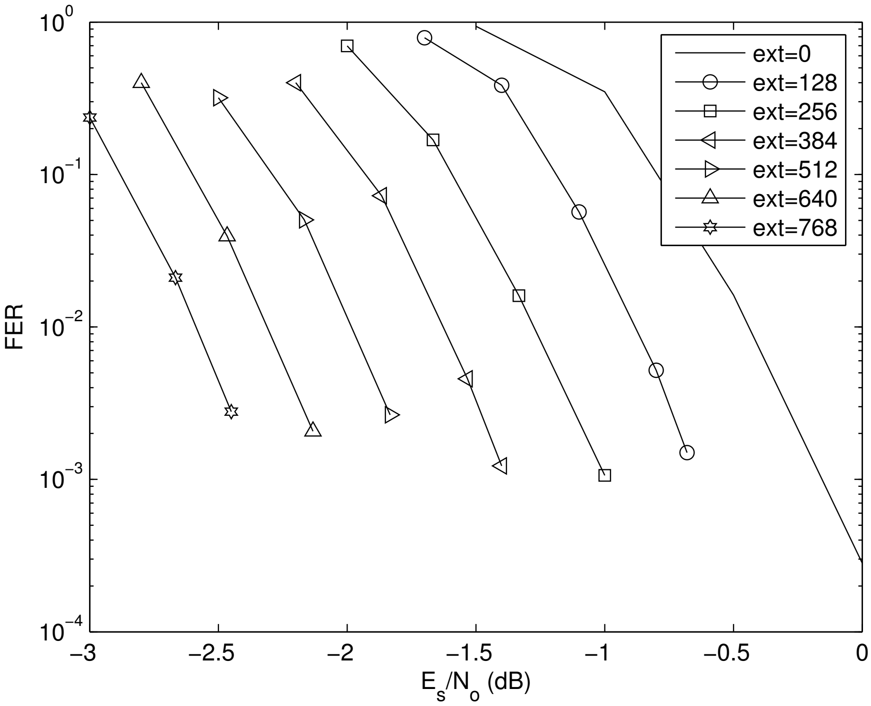


(b)

Figure 4.13: Performance of extended semi-random codes using the Extended-Identity approach over the Rayleigh fading channel with rates - from right to left - 8/14,..,8/20, mother code is of rate 8/14 with n=1792 max-iterations = 50, (a) BER (b) FER

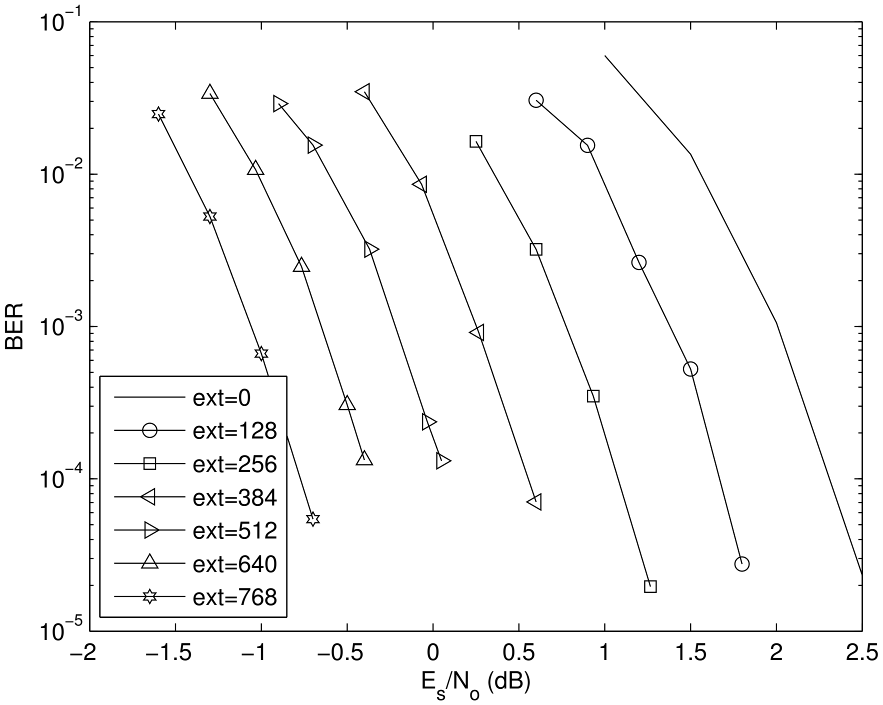


(a)

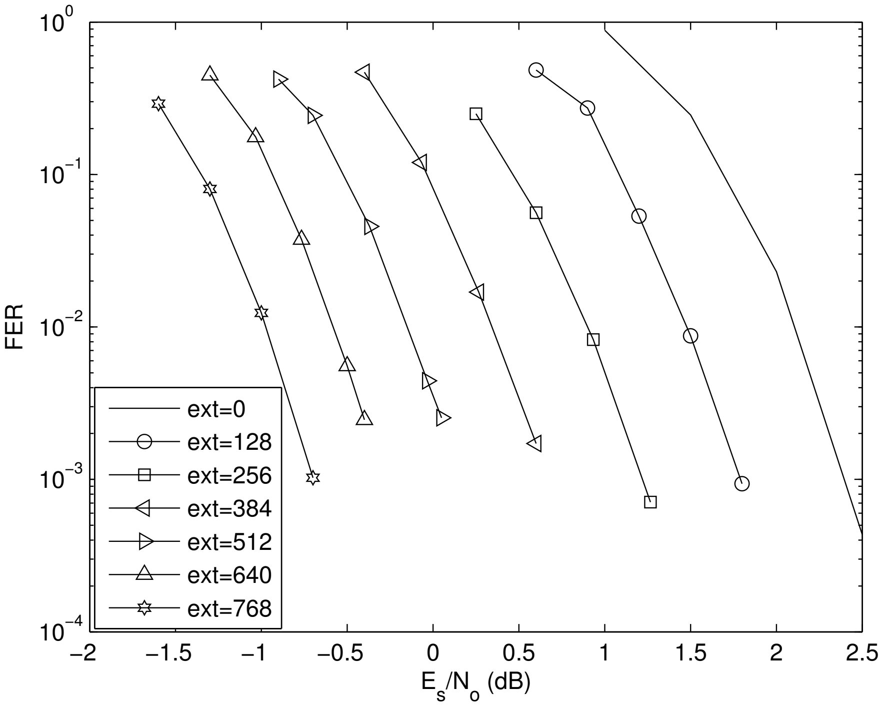


(b)

Figure 4.14: Performance of extended semi-random codes using the Extended-Permuted approach over the AWGN channel with rates - from right to left - 8/14,..,8/20, mother code is of rate 8/14 with n=1792 max-iterations = 50, (a) BER (b) FER



(a)



(b)

Figure 4.15: Performance of extended semi-random codes using the Extended-Permuted approach over the Rayleigh fading channel with rates - from right to left - 8/14,..,8/20, mother code is of rate 8/14 with n=1792 max-iterations = 50, (a) BER (b) FER

umn weight leads to slightly better performance for the “Extended-Permuted” approach, since columns of comparatively larger weight are connected to more check nodes, which improves the performance of the iterative decoder.

4.3 Type-II Hybrid ARQ

ARQ combines error detection and retransmission to improve the reliability of data delivery. A communication system that combines ARQ and FEC is called a Hybrid ARQ system [2]. In *type-I Hybrid* ARQ, a packet is encoded for both error detection and error correction. Upon reception of a packet, it is first decoded by the FEC decoder and then checked for errors [30]. If errors are detected, a retransmission request is sent to the transmitter. *Type-II Hybrid* ARQ adapts to changing channel conditions through the use of *incremental redundancy* [30]. A system using this ARQ scheme employs FEC with a range of code rates. The initial transmittion corresponds to the code with the highest rate (minimum redundancy). At the receiver the packet is decoded by the FEC decoder. If errors are detected then the transmitter sends additional parity bits (that were not sent yet), thereby reducing the rate of the code. The receiver appends these bits to the received packet allowing for increased error correction capability.

The rate-compatible LDPC codes designed via puncturing and extending in the previous sections have been employed to design a type-II hybrid ARQ/FEC system. In type-II ARQ, a packet is first transmitted using the highest rate code. If it is not deemed correctly decoded, an NACK (Negative ACKnowlegement) is fed-back to the transmitter and a new set of parity bits is provided by the transmitter (incremental retransmission). This new set of parity bits, combined with all previous transmissions, is treated as a codeword of a lower rate code in the family which provides stronger error correction capability. This is known as code combining. This procedure continues until all supplemental parity bits are used up, and then the procedure restarts with another "initial transmission". When a new copy of the same coded bits (either data or parity bits) are received, old copies are not discarded, but are combined with the new ones to facilitate decoding. This is known as packet combining. In general, packet combining is performed by averaging the soft decision values obtained from multiple copies of the same packet. Specifically for LDPC codes with the soft message-passing decoder, the input message to the decoder (in log-likelihood ratio (LLR) form) of a bit s_i is obtained by $\left(\sum_{j=1}^{k}\left(\frac{2r_i^{(j)}\alpha_i^j}{\sigma^2}\right)\right)/k$, where $r_i^{(1)}, r_i^{(2)}, ..., r_i^{(k)}$ are the multiple copies received for the same bit s_i. The above strategy is optimal for achieving high throughput either in stop-and-wait ARQ or selective-repeat ARQ systems, under the assumption that the feedback channel is noiseless, that the buffer size is infinite, and that the transmission latency, the feedback channel traffic and the decoding complexity are negligible [1].

Simulation Results

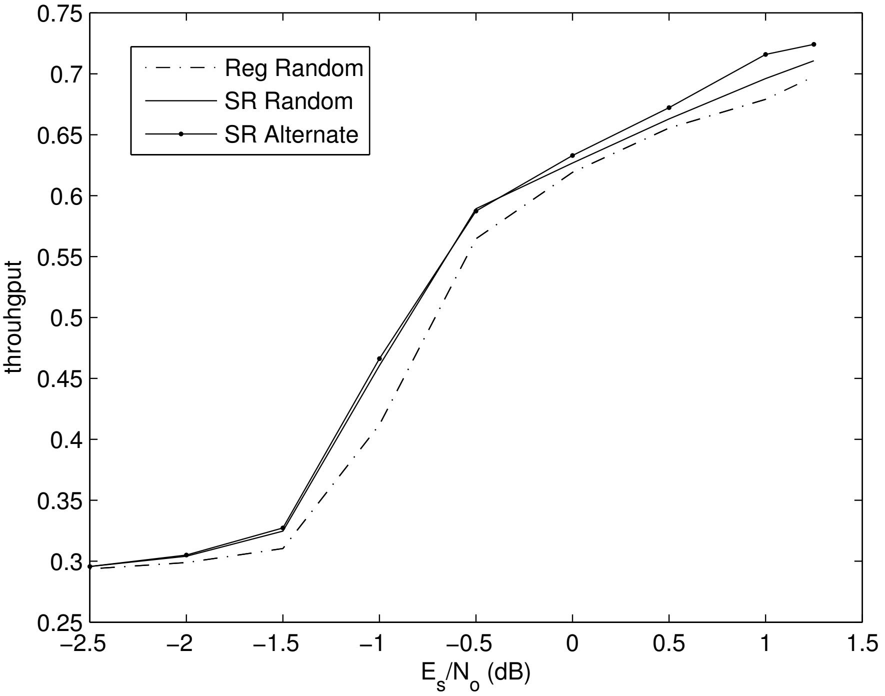
Figures 4.16 to 4.18 show the throughput for the type-II hybrid ARQ scheme described above that is based on rate-compatible LDPC codes. In the figures 'SR' denotes ARQ using semi-random LDPC codes, 'Reg' denotes ARQ using regular-(3,6) LDPC codes, 'Alternate' denotes alternate parity puncturing and 'Random' denotes a random pattern of parity bits. Figure 4.16 shows the throughput for ARQ schemes with rate-compatible LDPC codes employing only puncturing for AWGN and Rayleigh fading channels. The mother code in each case is a rate-8/14 code with blocklength of 1792 bits from which codes of rates 8/13, 8/12 and 8/11 are obtained via puncturing. We observe that codes based on the semi-random family of LDPC codes outperform codes based on the regular-(3,6) family. This is because punctured semi-random LDPC codes outperfom punctured regular-(3,6) codes, as was shown in Figures 4.6 and 4.7.

Figure 4.17 shows the throughput for ARQ schemes employing rate-compatible semi-random codes designed using puncturing *and* extending for the AWGN and Rayleigh fading channels respectively. The mother code is a rate-8/14 code with blocklength of 1792 bits from which codes of rates 8/13, 8/12 and 8/11 are derived via alternate puncturing. Codes with rates 8/15 to 8/20 and designed via extending. The two curves differ in the the form of the extending approach employed. It can be seen that the "Extending-Permuted" approach outperforms the "Extended-Identity" approach at low SNR, since at low SNR the low-rate codes obtained via extending dominate the performance. At high SNR the curves overlap since at high SNR the high rate codes obtained via puncturing dominate the performance.

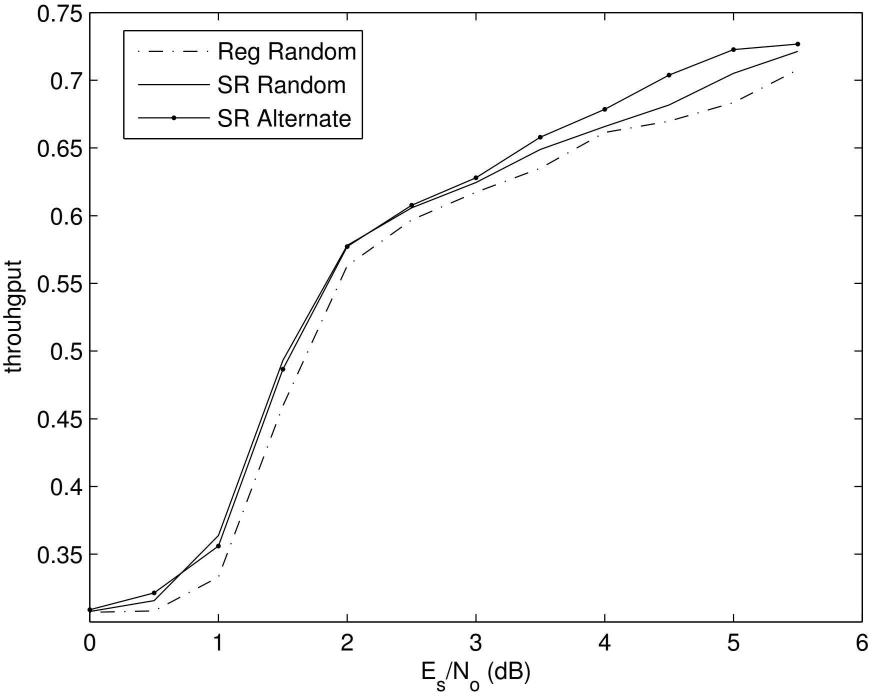
Figure 4.18 shows the throughput for ARQ schemes employing the best-performing rate-compatible semi-random codes designed in this work (using alternate puncturing and “Extended-Permuted” extending), and regular codes from [1] for the AWGN channel. The mother code is a rate-8/14 code with a blocklength of 1792 bits. There are 1024 information bits in each frame. Codes of rates 8/13 to 8/11 are designed via puncturing and rates 8/15 to 8/20 are designed via extending. The code rates that are used for this figure correspond to those used in [1]. It can be seen that the ARQ schemes employing semi-random codes outperform those employing regular codes at high SNR by upto 0.3-0.4 dB. At high SNR the codes obtained via puncturing dominate performance and punctured semi-random codes significantly outperform punctured regular-(3,6) codes as shown in Figures 4.6 to 4.7 and 4.16. It can also be seen that both curves reach the maximum throughput value of approximately 0.727 at the highest SNR. This is the maximum attainable throughput since this is the maximum code-rate ($8/11 \approx 0.727$) among the family of rate-compatible codes being used. Note that although the ARQ scheme employing semi-random LDPC codes achieves better performance by upto 0.4 dB as compared to that based on regular LDPC codes, the former has the big advantage of low encoding complexity, which does not exist for the latter system.

Chapter Summary

In this chapter, methods for the design of rate-compatible semi-random LDPC codes have been proposed. A puncturing pattern has been proposed for the design of rate-compatible punctured semi-random codes that perform better than randomly punctured codes. Furthermore, two approaches for designing extended rate-compatible semi-random LDPC codes have been proposed. A type-II hybird ARQ scheme based on the rate-compatible LDPC codes designed in this chapter has been shown to outperform an existing scheme based on regular LDPC codes by upto 0.4 dB. Additionally, the proposed hybrid ARQ scheme based on semi-random LDPC codes offers the major advantage of low complexity encoding.

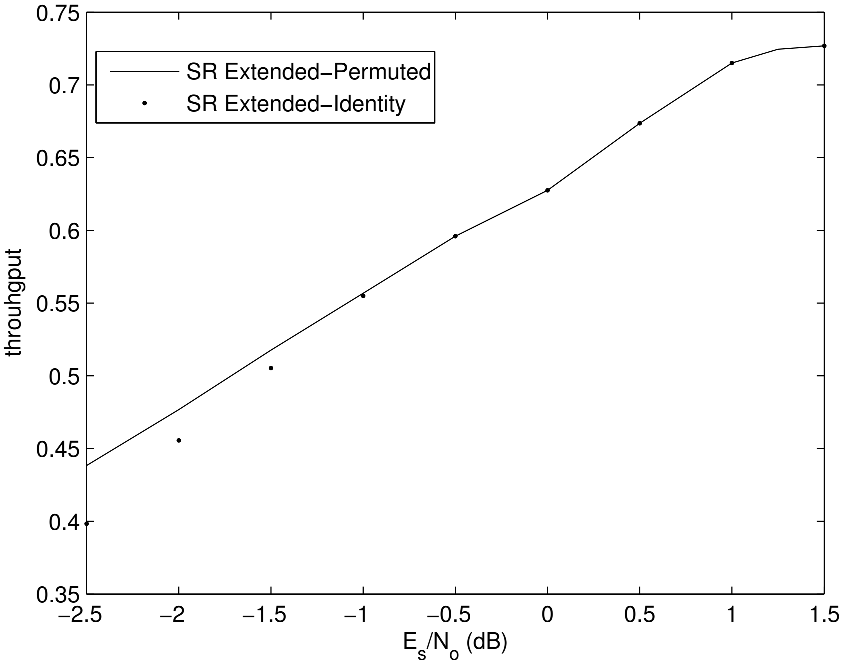


(a)

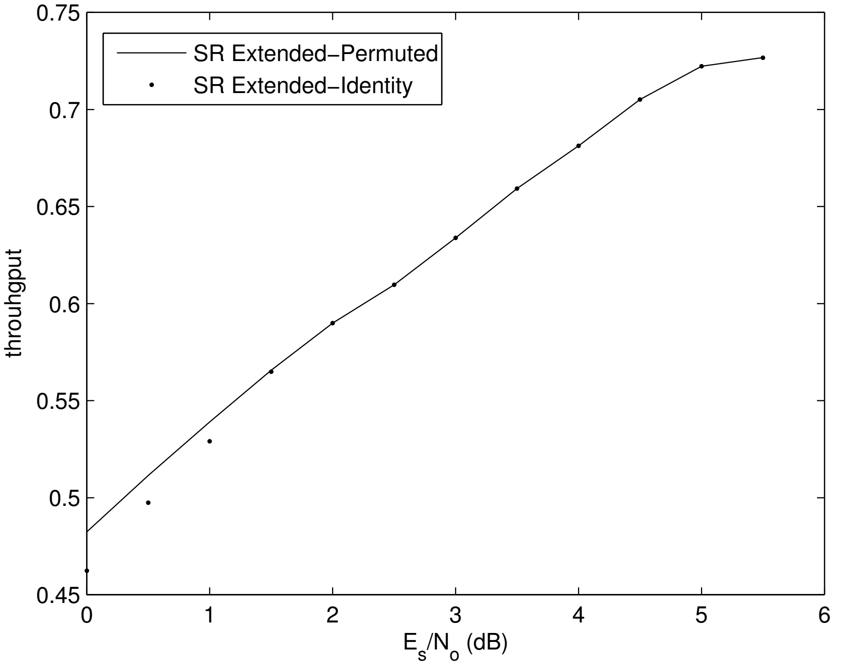


(b)

Figure 4.16: Throughput comparison of ARQ schemes based on semi-random and regular punctured codes, the mother code is a rate-8/14 code with n=1792, code rates 8/13, 8/12 and 8/11 are obtained through puncturing, max-iterations = 50, (a) AWGN channel (b) Rayleigh fading channel



(a)



(b)

Figure 4.17: Throughput comparison of type-II ARQ scheme employing semi-random RC-LDPC codes based on “Extended-Permuted” and “Extended-Identity” approaches, the mother code is a rate-8/14 code with n=1792, code rates 8/13 to 8/11 are obtained through puncturing and code rates 8/15 to 8/20 are obtained through extending, max-iterations = 50, (a) AWGN channel (b) Rayleigh fading channel

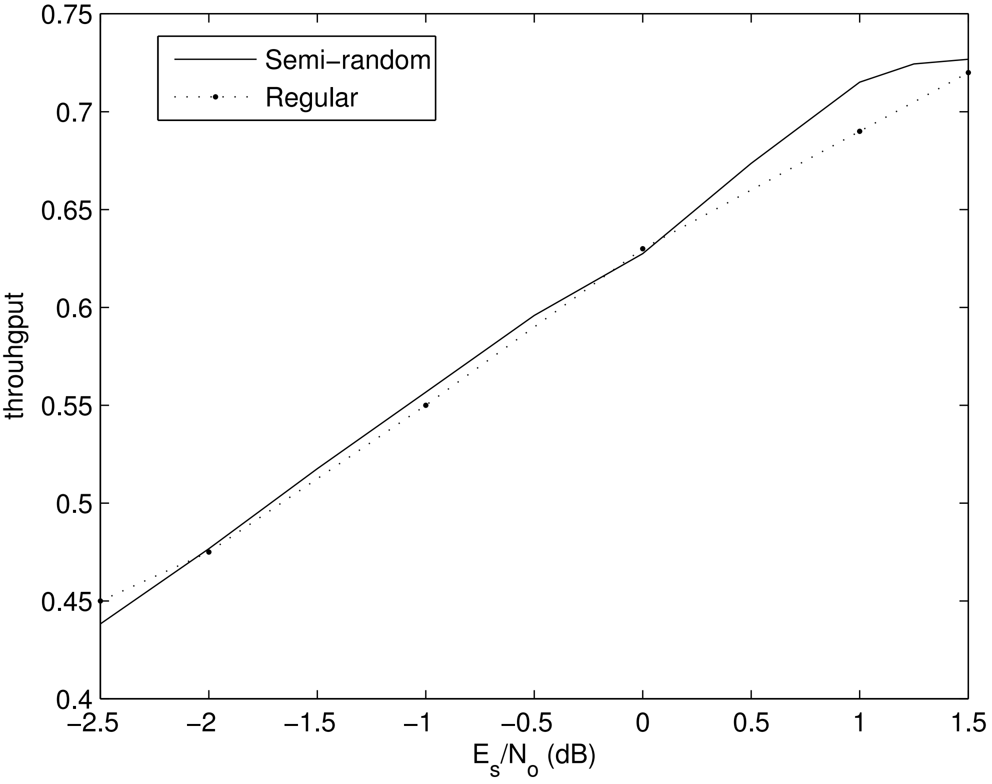


Figure 4.18: Throughput comparison of ARQ schemes based on the proposed rate-compatible semi-random codes designed in this work - using alternate puncturing and Extended-permuted extending - and regular codes from [1], the mother code is a rate-8/14 code with n=1792, code rates 8/13 to 8/11 are obtained through puncturing and code rates 8/15 to 8/20 are obtained through extending.

Chapter 5

LDPC Codes for Coded Cooperation Diversity

As has been demonstrated in the Chapter 4, rate-compatible semi-random LDPC codes outperform rate-compatible regular LDPC codes, and are therefore good candidates for systems employing rate-compatible LDPC codes. In this chapter the application of rate-compatible - specifically *punctured* - semi-random LDPC codes to systems employing *coded cooperation diversity* is investigated. The work presented in this chapter is based on the system model presented in [48].

5.1 System Model

5.1.1 Network Architecture

The coded cooperation scenario is illustrated in Figure 5.1. Coded cooperation starts by forming *clusters* of users, where users in a cluster cooperate to transmit their information to a common BS. The users within a cluster are called *partners*. The selection of users to join or leave a cluster can be based on the quality of the interuser channels or any other factor. Users in a cluster are assumed to operate in a full-duplex mode, i.e., they can transmit and receive simultaneously.

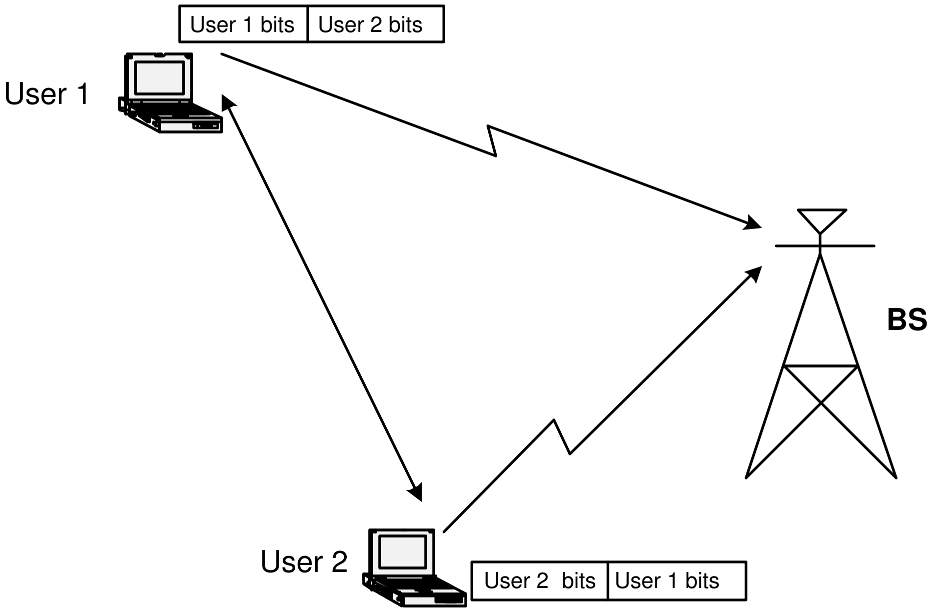


Figure 5.1: Schematic diagram of a 2-user cluster employing coded cooperation.

Let J be the number of cooperating users in a cluster. For each user in the cluster the transmission of each frame spans nT seconds, where n is the number of bits in the frame and T is the bit duration. A frame is formed by encoding k bits (information bits and cyclic redundancy check (CRC) bits) into $n = k/R$ bits, where R is the code rate of the error-correcting code. Partners cooperate by dividing their n-bit frames into J *subframes* containing $n_1, n_2, \ldots, n_J$ bits, where $n = n_1 + n_2 + \ldots + n_J$. The distribution of coded bits over the subframes depends on the coding technique used. In the first n_1T seconds of each frame, each user transmits his first subframe composed of $n_1 = k/R_1$ coded bits, where R_1 is the code rate of the codeword in the first subframe, obtained by puncturing the n-bit codeword. Clearly, $R_1 > R_J = R$. Upon the end of the first subframe, each user decodes the rate-R_1 codewords of his partners.

In the remaining $J-1$ subframes, each user in the cluster transmits one subframe for each of his $J-1$ partners. Each of these subframes contains parity bits of one of his partners which were not sent yet to the BS. Figure 5.1 shows the contents of the J subframes of each user in a 2-user cluster, i.e., $J = 2$. If a user was not able to decode the first subframe of his partner, whom he should send his parity in a given subframe, then he sends his next parity subframe, i.e., the parity subframe that was not yet sent by any of his partners. Thus each user transmits a total of n bits per source block over the J subframes. The *cooperation level,* β is defined as the percentage of the total bits per each source block that each user transmits for his partners, i.e., $\frac{n-n_1}{n}$.

The partitioning of the coded bits in the J subframes may be achieved through puncturing a mother code as in [38], where rate-compatible punctured convolutional (RCPC) codes [32] were used to implement coded cooperation. In this work, the rate-R code is selected from a semi-random LDPC code family designed in Chapter 4. High-rate codewords are obtained through puncturing a rate-R mother code. The parity bits to be transmitted in each subframe are selected according to the puncturing pattern of the code, which is known and fixed to all partners in a cluster. The receiver combines all the received subframes for a user to produce a codeword of a more powerful code (a lower code rate) [32]. The code rates corresponding to different cooperation levels are $R_1 > R_2 > \ldots > R_J = R$.

5.1.2 Physical Link

After encoding the information block, the coded bits are modulated using BPSK. The matched filter output at user u due to user l in the time interval t in the first subframe is modeled by

$$y_{l,u}(t) = \sqrt{E_i} a_{l,u} s_l(t) + z_u(t), \tag{5.1}$$

where $s_l(t)$ is the signal transmitted from user l in time instance t in the first subframe and $z_u(t)$ is an AWGN sample at user u with a Normal distribution given by $\mathcal{N}(0, \frac{N_0}{2})$. Here, E_i is the average received energy through the interuser channel and the average interuser signal-to-noise ratio (SNR) is $\gamma_i = \frac{E_i}{N_0}$. The coefficient $a_{l,u}$ is the gain of the interuser channel between user l and user u. The interuser channels are assumed to be independent and identically distributed (i.i.d) with a Rician or a Nakagami distribution. Rician fading channels arise if a line-of-site (LOS) exists between the transmitter and the receiver [9]. In this model the received signal is composed of two signal-dependent components; namely, the LOS and multipath components. In this case, the pdf of the interuser SNR [73] is given by

$$f_\gamma(\gamma) = \frac{(1+\kappa)}{\gamma_i} \exp\left[-\kappa - \frac{(1+\kappa)\gamma}{\gamma_i}\right] I_0\left(2\sqrt{\frac{\kappa(1+\kappa)\gamma}{\gamma_i}}\right), \qquad \gamma \geq 0, \tag{5.2}$$

where κ denotes the ratio of the LOS energy to the multipath energy and $I_0(.)$ is the zero-order modified Bessel function of the first kind. Nakagami distribution was shown to fit measurements in micro-cellular systems [74], where the received SNR has the pdf [75]

$$f_\gamma(\gamma) = \left(\frac{m}{\gamma_i}\right)^m \frac{\gamma^{m-1}}{\Gamma(m)} \exp\left(-\frac{m\gamma}{\gamma_i}\right), \gamma \geq 0, m \geq 0.5, \tag{5.3}$$

where $\Gamma(.)$ is the Gamma function and $m = \frac{\gamma_i^2}{\mathrm{Var}[\sqrt{\gamma}]}$ is the Nakagami parameter that indicates the fading severity.

When $u = 0$, the signal model in (5.1) represents the uplink channel from user l to the BS, where the received average energy is denoted by E_s and the average uplink SNR is $\gamma_s = \frac{E_s}{N_0}$. The uplink channels from different users are assumed to be i.i.d with a Rayleigh distribution. Moreover, the interuser channels and the uplink channels are assumed to be mutually independent and slow enough such that the fading process stays fixed within a frame. This is a reasonable assumption for slowly moving mobile units that are separated enough in the space [76]. In addition, we assume that the interuser channels are reciprocal as in [36, 37]. At the receivers of users and the BS, coherent detection is employed using perfect channel side information.

5.2 Simulation Results

In the following, we consider coded cooperation with cluster sizes $J = 1, 2$. Each user in the cluster employs a mother code of rate $R = R_J = \frac{1}{4}$. In all cases, the source block is $k = 128$ information bits. The block-size and code rates used correspond to those used in [48]. To minimize decoder complexity and decoding delay, the number of maximum decoder iterations is limited to 10.

Figure 5.2 shows the results for the scenario when both users have the same average SNR for their uplink channels. The BER curves are shown for various interuser channel average SNR values. In the figure, 'Perfect interuser channel' denotes perfect Rayleigh interuser channels, i.e., infinite interuser SNR. Coded cooperation clearly achieves impressive gains compared with the non-cooperative system. Even when the interuser channel has an average SNR of 0 dB, much less than that of the user uplink channels, coded cooperation still provides a gain of 1.5-2.25 dB, which is quite significant.

For coded cooperation, two major factors affect the performance:

1. The average probability of cooperation defined as the average probability that

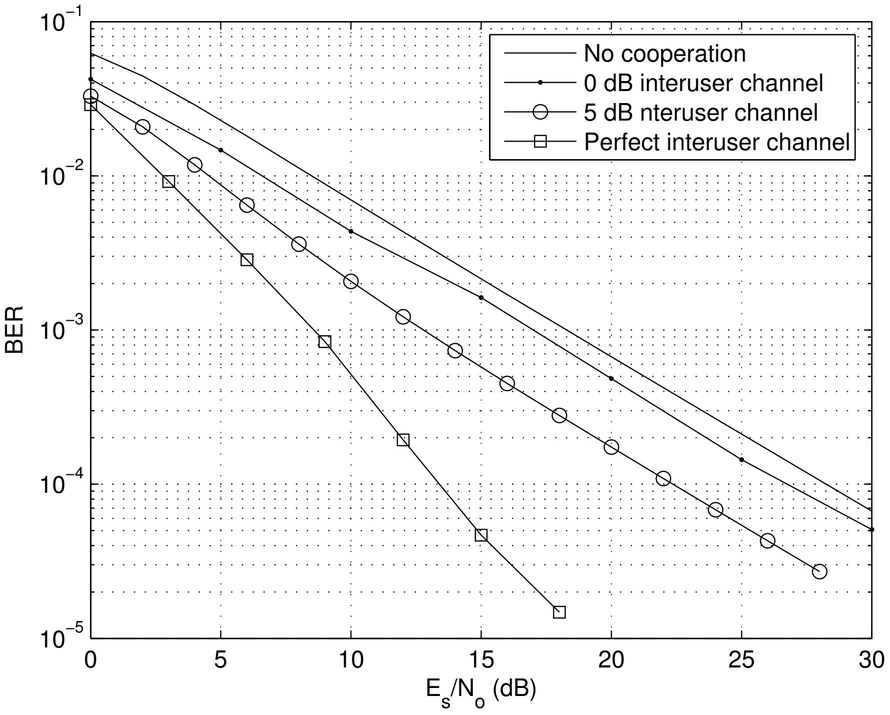


Figure 5.2: Performance of coded cooperation employing semi-random RC-LDPC codes for different values of the average SNR of the interuser channel.

a user is able to decode his partner, which depends on the code rate used during the first subframe and the quality of the interuser channel.

2. The diversity achieved through cooperation, which varies with the cooperation level.

Increasing the cooperation level (and therefore increasing the code rate used during the first subframe of transmission), reduces the probability that a partner will be able to decode his partner and therefore reduces the average probability of cooperation. However, in the event that cooperation occurs (i.e., a user decodes his partner), increasing the level of cooperation increases the diversity available at the BS, which improves the performance.

Figure 5.3 shows the performance for a 2-user cluster for different levels of cooperation for the case of a perfect interuser channel. A perfect interuser channel implies full cooperation, i.e, each user is always able to deocde his partner. Therefore increasing the cooperation level leads to increased diversity and hence to improved performance.

The average probabilities of cooperation for different interuser channels (obtained from simulations) with an average interuser SNR of 5 dB corresponding to different cooperation levels are shown in Table 5.2. Figure 5.4 shows the performance for a 2-user cluster for different levels of cooperation for the case of a Rayleigh interuser channel with an average interuser SNR of 5 dB. It can be seen that a cooperation level of 37.5% (which implies that half the parity bits are punctured during the first subframe) performs better than the cooperation levels of 25%, 50% and 62.5%. This can be explained as follows. The cooperation level of 25% corresponds to the lowest code rate (among the code rates corresponding to the different cooperation levels used), and therefore leads to high average probability of cooperation (the value obtained from simulation is 83%). However, it offers less diversity as compared to the 37.5% cooperation level. On the other hand, the 62.5% cooperation level offers the largest diversity advantage, but suffers from a low average probability of cooperation. The 37.5% cooperation level provides the best performance since it leads to good average probability of cooperation *and* offers good diversity.

Table 5.1: The average probabilities of cooperation over Rayleigh, Nakagami (m=3) and Rician (κ=10 dB) interuser channels with average SNR of 5 dB, and cooperation level of β.

β	Rayleigh	Nakagami	Rician
25%	0.83	0.97	0.98
37.5%	0.78	0.96	0.97
50%	0.71	0.91	0.96
62.5%	0.56	0.77	0.8

The same trend can be observed for the case of the Nakagami interuser channel with a Nakagami parameter of m=3, as shown in Figure 5.5, and the Rician interuser channel with a Rician factor κ=10 dB, as shown in Figure 5.6. For these two cases the advantage offered by the 37.5% cooperation level over the other two cooperation levels in more pronounced since the average probability of cooperation increases and in both cases is very close to the maximum that is observed over these two channels.

Chapter Summary

In this chapter rate-compatible punctured semi-random LDPC codes were used to investigate the performance of coded cooperation diversity in wireless networks. Simulation results show that coded cooperation achieves impressive gains compared with the non-cooperative system. Furthermore, the effect of varying the cooperation level for the 2-user cluster size has been investigated, and it has been shown that for semi-random LDPC codes, the cooperation level corresponding to puncturing half of the parity bits offers the best performance over a variety of interuser channel conditions, since it leads to a high average probability of cooperation and provides a good amount of diversity at the BS.

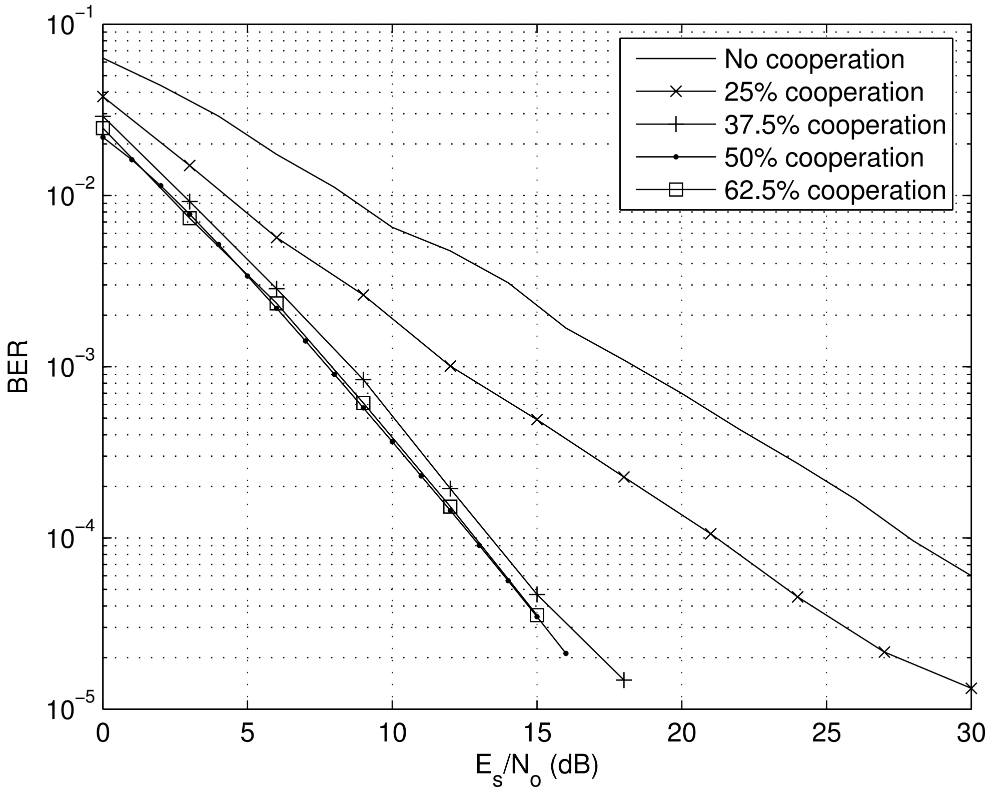


Figure 5.3: Performance of coded cooperation for the perfect interuser channel and varying levels of cooperation.

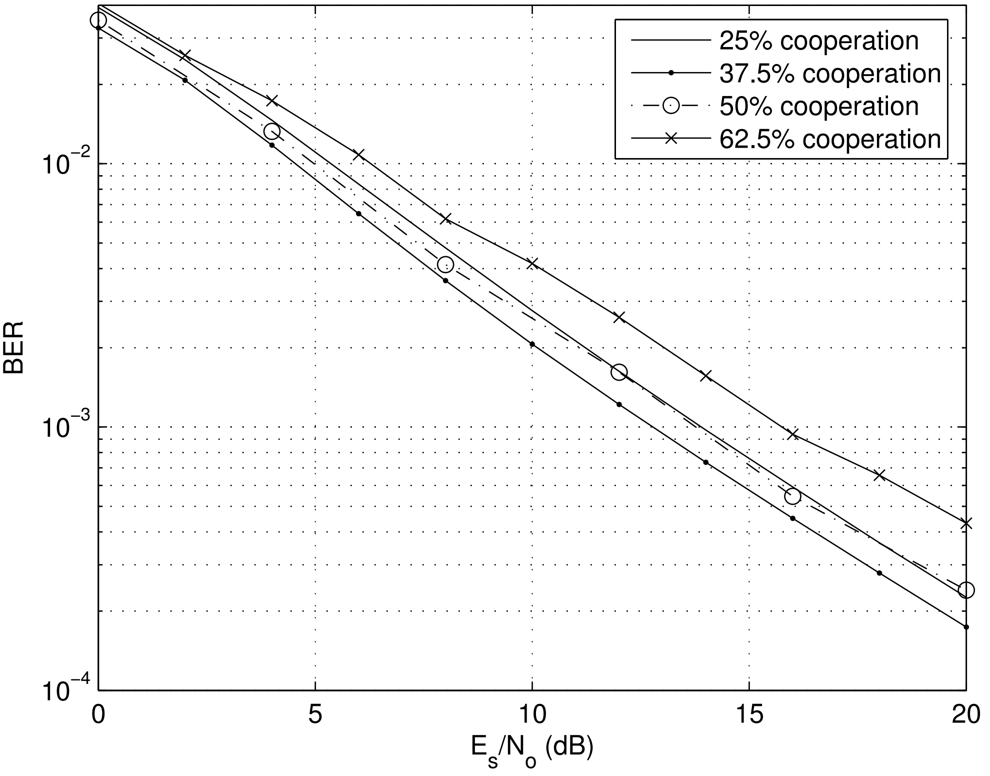


Figure 5.4: Performance of coded cooperation for a 2-user cluster for the case of the Rayleigh interuser channel with average SNR of 5 dB and varying levels of cooperation.

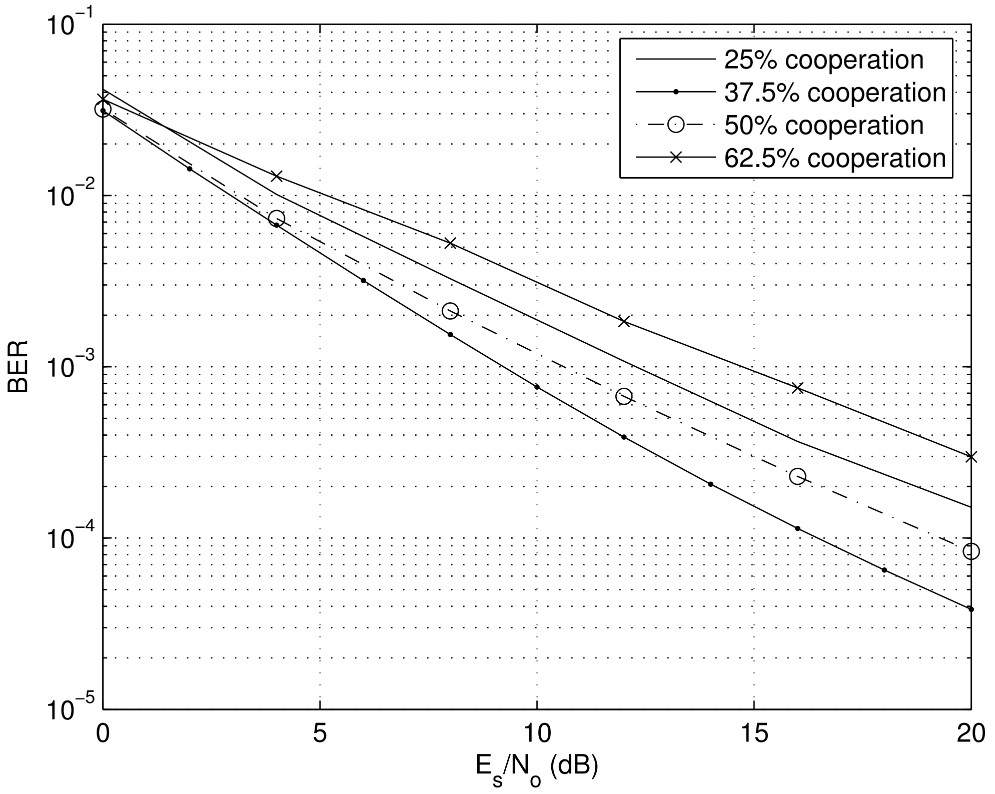


Figure 5.5: Performance of coded cooperation for a 2-user cluster for the case of the Nakagami (m=3) interuser channel with average SNR of 5 dB and varying levels of cooperation.

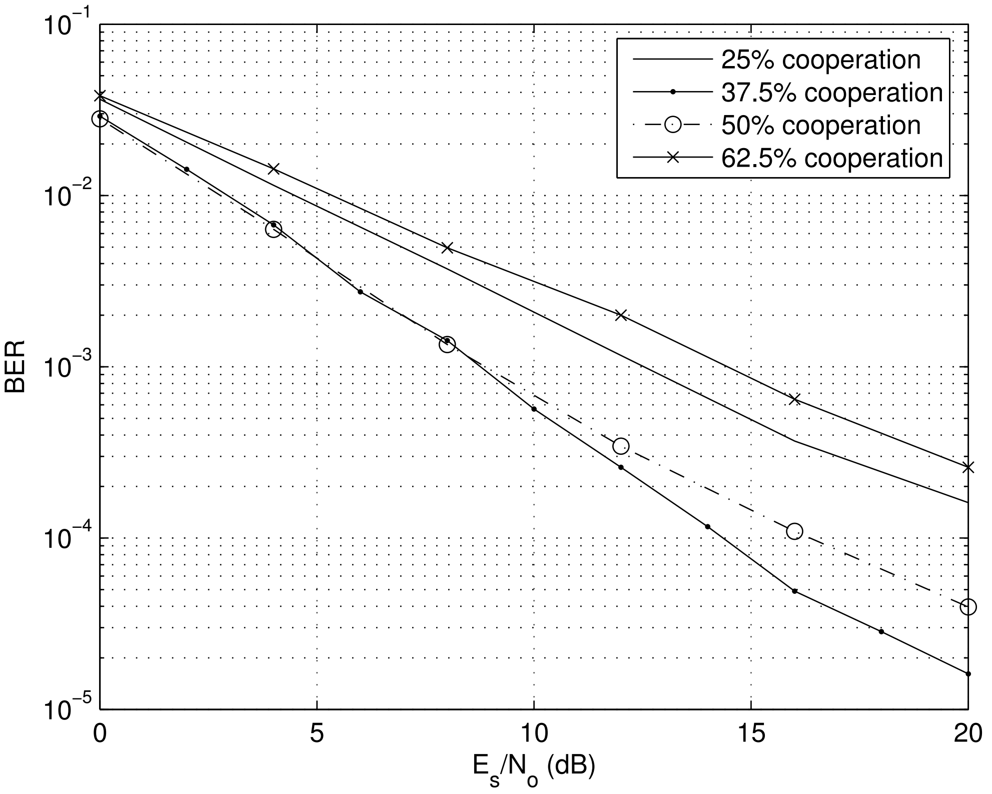


Figure 5.6: Performance of coded cooperation for a 2-user cluster for the case of the Rician ($\kappa = 10$ dB) interuser channel with average SNR of 5 dB and varying levels of cooperation.

Chapter 6

Loop removal from Semi-Random Codes

Removal of small loops from a LDPC code improves the performance in terms of lower BER and reduced error floor [56]. This chapter is concerned with the design of an algorithm for removing small loops from semi-random LDPC codes. Most random constructions of the parity-check matrix contain loops of small lengths [56]. During decoding, small loops cause the values of incorrect bits propagate back around to themselves, rapidly reinforcing their belief and resist the efforts of the algorithm to correct them. In [56] an algorithm to remove loops from a randomly constructed parity-check matrix was proposed. This algorithm is based on the adjacency matrices [71] and has its roots in Graph theory.

For the case of the semi-random parity-check matrix, the deterministic part of the parity-check matrix is constructed such that it does not contain loops of any size. However, it is the random part of the parity-check matrix which may be the source of loops. Furthermore, the concatenation of the random part with the deterministic part may lead to the creation of new loops. When the algorithm in [56] is used to remove loops in a semi-random code, it will destroy the structure of the deterministic part of the parity-check matrix which is needed for low-complexity encoding. Hence the algorithm given in [56] needs to be modified before it can be applied to semi-random LDPC codes. In the following the original loop-removal algorithm in [56] is described, followed by the modified loop removal algorithm for semi-random LDPC codes.

6.1 The Loop Removal Algorithm

The loop removal algorithm given in [56] uses the code's adjacency matrix to locate unwanted loops, and then certain edges within the graph are exchanged to eliminate those loops (without simultaneously creating any others). The algorithm is composed of two phases: namely the loop detection and loop removal phases. During the first phase the existing loops in a parity-check matrix are detected, whereas during the second phase the loops detected during the first phase are removed.

6.1.1 Loop Detection

The adjacency matrix of the code can be used to locate loops of any length. Denoting all the nodes of the codes' graph as $v_1, v_2, ..., v_p$, and define the *adjacency matrix* $A = [a_{ij}]$ to be the $p \times p$ symmetric binary matrix

$$a_{ij} = \begin{cases} 1 & \text{if an edge connects } v_i \text{ with } v_j \\ 0 & \text{otherwise} \end{cases}$$

The natural ordering of the nodes for an LDPC graph results in the relationship

$$A = \begin{pmatrix} \mathbf{0} & H \\ H^T & \mathbf{0} \end{pmatrix}, \tag{6.1}$$

which shows that A can be easily constructed from H. Consider the square of A, it is given by :

$$A^2 = \begin{pmatrix} HH^T & \mathbf{0} \\ \mathbf{0} & H^TH \end{pmatrix}, \tag{6.2}$$

where the elements of A^2 can be calculated as

$$A^2_{ij} = \sum_{k=1}^{p} a_{ik}a_{kj} \ . \tag{6.3}$$

Note that this sum represents the number of paths of length 2 between v_i and v_j, because whenever $a_{ik} = a_{kj} = l$ we have two edges joining v_i to v_k to v_j. This observation can be extended using induction to yield the following theorem [56].

Theorem 1 [56] The (i, j)-entry of A^n equals the number of paths of length n from v_i to v_j.

proof: see [56].

The diagonal elements A^n_{ii} represent the number of paths of length n starting and ending at node v_i. These paths include the n-loops that pass through v_i, but also other degenerate loops that repeat nodes and/or backtrack along edges. The following theorem is used to avoid these unwanted cases when locating loops [56].

Theorem 2 [56] In a graph with girth n, the nodes v_i and v_j are said to be directly opposite each other in an n-loop iff

$$A^{n/2}_{ij} \geq 2 \quad \text{and} \tag{6.4}$$

$$A^{(n/2)-2}_{ij} = 0\,. \tag{6.5}$$

proof: see [56].

Consider a graph of girth 4, as most random graphs are. For two nodes v_i and v_j to be opposite on a 4-loop (refer to Figure 6.3: nodes i and j or k_1 and k_2 are opposite each other) we require that there are at least two paths of length 2 between them, and so (6.4) is satisfied. The only situation where (6.4) does not automatically result in v_i and v_j being on a 4-loop is when the paths are able to backtrack along themselves, i.e., when $v_i = v_j$ or $i = j$. Thus, all non-diagonal elements of A^2 with a value of at least 2 will correspond to nodes on a 4-loop. This is equivalent to (6.4), because when $n = 4$ we have

$$A^{4/2-2}_{ij} \geq A^0_{ij} = I_{ij} = 0\,. \tag{6.6}$$

Once all 4-loops are found, they can be removed using the algorithm explained below, resulting in a graph with a girth of 6. By induction we can now assume that we have a graph with girth n, and wish to locate the n-loops in the graph. Again we need to find at least two paths of length $n/2$ between v_i and v_j, and so Theorem 1 immediately gives us the condition in (6.4). Any v_i and v_j satisfying (6.4) are either on a genuine n-loop or lie on smaller loops intersecting in a figure-of-eight shape (which cannot be the case here as the graph has girth n), unless there are paths between v_i and v_j with length less than $n/2$. The next shortest possible length is $(n/2)$ - 2, and so (6.5) ensures that no such shorter paths occur.

A loop located in this way can be removed, and the search is repeated until the girth increases from n to $n + 2$. After that, larger loops can be removed if a larger girth is required. Obviously, each loop can be defined in terms of multiple pairs (an n-loop has $n/2$ opposite pairs), but any pair is sufficient for the removal algorithm.

6.1.2 Loop Removal

Once an unwanted loop is detected, the next task is to remove it from the graph. It is always possible to destroy the loop by swapping around any edges composing the loop. However we need to do this in a way such that no new loops are formed. First, we need an edge from the loop. The above detection technique will give us two nodes v_i and v_j. A node v_k, of the loop adjacent to v_j will be at distance $n/2$ - 1 from v_i, and so we take any v_k (the two directions around the loop give two possibilities) that satisfies

$$A_{ik}^{(n/2)-1} > 0 \quad \text{and} \quad a_{jk} = 1 \,. \tag{6.7}$$

This gives a loop-edge $e = \overline{v_j v_k}$. We now find C_e, the set of all nodes at a distance greater than n - 1 from e. This consists of the set of nodes $\{v_c\}$, for which $A_{cj}^{n-1} = A_{ck}^{n-1} = 0$.

An edge e' with both end-points in C_e is randomly chosen. If there are no edges with this property then e is not removable from the loop, and a different loop edge will need to be selected. Let the end-nodes of e' be v_l and v_m. Delete edges e and e' from the graph, and replace them with $e_2 = \overline{v_j v_m}$ and $e'_2 = \overline{v_l v_k}$, as in Fig. 6.1.

Theorem 3 [56] Replacing e and e' with the new e_2 and e'_2 will remove the original loop that e was part of and create no new loops of size n or less.

proof: see [56].

We know from the definition of C_e that the edges e_2 and e'_2 did not exist before applying the loop removal algorithm, and hence it is possible to perform the exchange. The result is that the old loop is definitely removed, since one of its edges has been deleted. In order to confirm that no new n-loops could have been produced, we examine the three possible cases:

1. A new loop comprising of e_2 and other pre-existing edges would require a path from v_j to v_m, of length n - 1 or less. But $v_m \in C_e$, and by the definition of C_e, this is not possible.

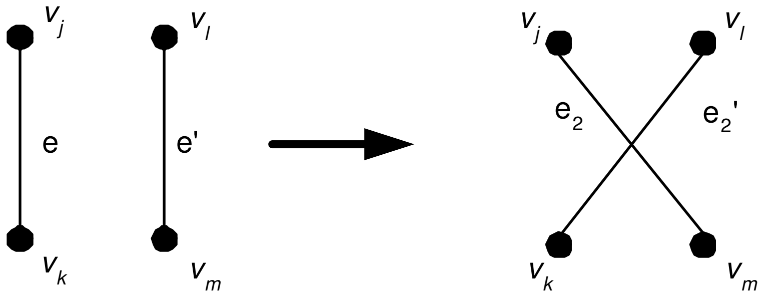


Figure 6.1: Replacing the old edges e and e' with the new e_2 and e_2'

2. Similarly $v_l \in C_e$, so there can be no loop containing e_2' with other old edges.

3. The other potential way to form a new loop would be to include both e_2 and e_2'. There is nothing in our selection criteria preventing v_l and v_m being connected by a path of length 3. However we know that v_i and v_j were previously on an n-loop, and so with e removed the shortest path between them must have length n - 1. Therefore no new loops can be created with a length less than (n - 1) + 1 + 3 + 1 = n + 4.

6.2 The Modified Algorithm

The loop-removal algorithm described in the previous section removes loops by breaking *any* edge in a particular loop (while ensuring that no new loops are created). However for the case of semi-random LDPC codes the respective loops have to be broken while ensuring that the structure of the semi-random parity-check matrix is maintained.

$$H = \left(\begin{array}{c|c} \underbrace{\begin{matrix} 1 & & & 0 \\ 1 & 1 & & \\ & 1 & 1 & \\ 0 & & 1 & 1 \end{matrix}}_{Deterministic} & \underbrace{\begin{matrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{matrix}}_{Random} \end{array} \right).$$

Figure 6.2: A Semi-Random parity-check matrix

Figure 6.2 shows a parity-check matrix for a semi-random LDPC code. It is clear that to maintain the structure (which enables efficient low-complexity encoding), the deterministic part of the parity-check matrix should not be altered. It should be noted that *all check nodes are shared by both the random and the deterministic parts of the matrix, while variable nodes are split between the two parts.* There are two types of loops which can be present in a semi-random LDPC code:

1. Loops formed only by variable nodes of the random sub-matrix.

2. Loops that involve variable nodes of both the random and the deterministic sub-matrices.

Loops of the first kind do not place any restriction on the loop-removal algorithm, since they do not affect the deterministic part. However loops of the second kind must by removed by *swapping edges only in the random part.*

6.2.1 Loop Detection

The method for detecting loops is the same as that defined in Section 6.1.1 which uses the adjacency matrix of the code and powers of the adjacency matrix. This implies that a loop is found if (6.4) and (6.5) are satisfied.

6.2.2 Loop Removal

Once a loop is found, it has to be destroyed while ensuring that no new loops are formed and the structure of the semi-random code is maintained.

Proposition 1 The loop removal procedure involves the breaking of two edges and the formation of two new edges (see Figure 6.1). To maintain the structure of the code, the main restriction on the loop removal procedure for semi-random LDPC codes is that both the edges being broken (one of these is a part of the loop to be removed, while the other is not) *must involve variable nodes in the random part of the semi-random code.*

The loop detection technique will give us two nodes v_i and v_j. A node v_k, of the loop adjacent to v_j will be at distance $n/2$ - 1 from v_i. There are two possibilities for v_k that satisfy (6.7) (the two directions around the loop give two possibilities), but the v_k chosen for the purpose of loop removal has to be a variable node belonging to the random part of the semi-random code. The following example will further help to clarify the concept.

Example: A semi-random LDPC code of rate 1/2 and block size 128 has a girth of 4. Therefore, there are a total of 64 check nodes and 128 variable nodes. Suppose during the loop detection process, a loop of length 4 is encountered as shown in Fig. 6.3. It is comprised of two check nodes (numbered 14 and 15) and two variable nodes (numbered 15 and 67). There are two possibilities for k that satisfy (6.7), namely $k_1 = 15$ and $k_2 = 67$. However, to remove this loop we will chose $k_2 = 67$ (which lies in the random part of the matrix) as v_k so that the structure of the semi-random code is maintained. The edge $\overline{v_l v_m}$ (that will be swapped with an edge of the detected loop) must also be chosen such that the variable node v_l lies in the random part of the matrix. This ensures that the loop is removed while the structure of the semi-random LDPC code is maintained.

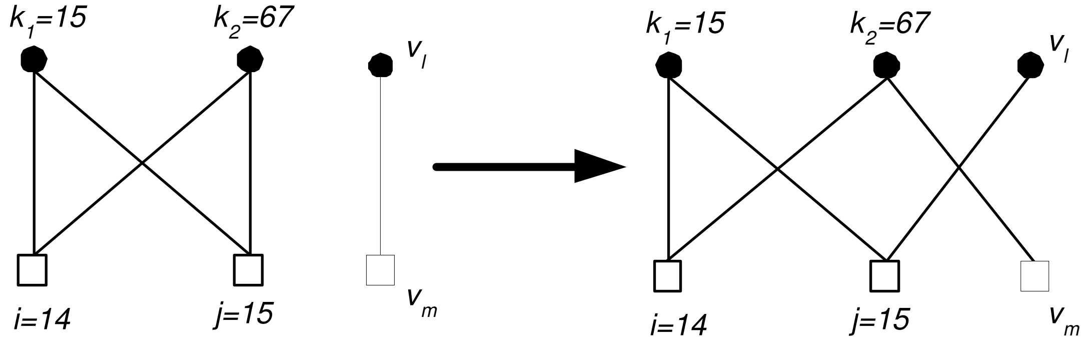


Figure 6.3: A 4-loop before and after removal

An explicit statement of the algorithm suggested by the above is given in the Appendix.

6.3 Simulation Results

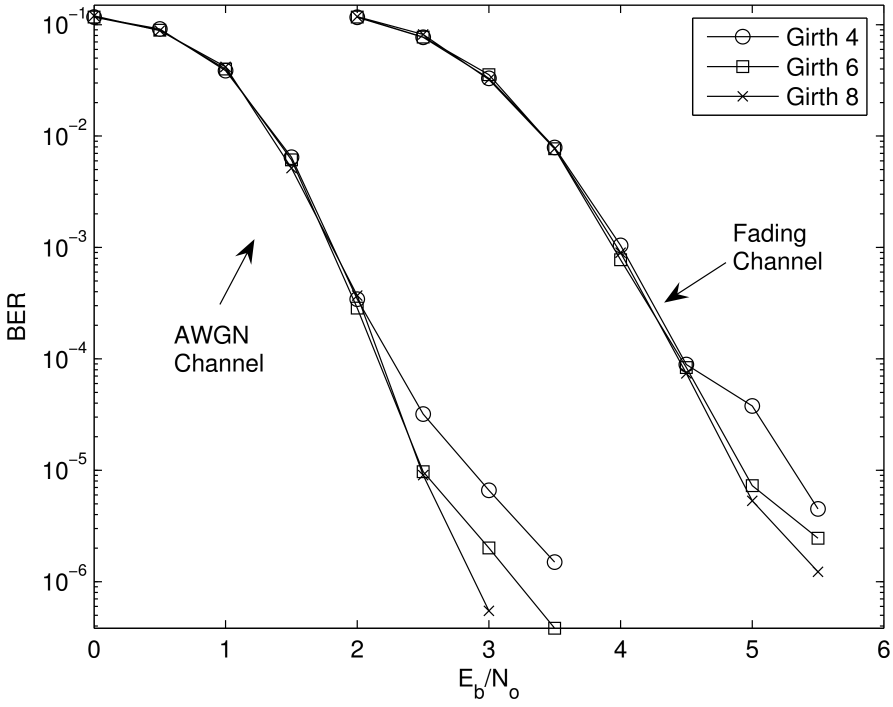
The effect of loop-removal from a LDPC code can by observed by obtaining the histogram of the distribution of loops in the code, before and after removing loops. However, to compare the performance of codes of different girth, Monte Carlo simulation has to be used. Figure 6.4 shows the performance results obtained by simulating rate-half semi-random LDPC codes over the AWGN and uncorrelated Rayleigh fading channels. The block size (n=1000) and the code rate used correspond to those used in [56]. For each SNR point, enough codewords are simulated to generate at least 50 codeword errors. The coded bits are modulated using BPSK. For the Rayleigh fading channel, coherent detection is employed at the receiver using perfect channel side informtion. Ten different semi-random codes of girth 4 were generated, and then the modified loop removal algorithm was applied to obtain two new sets of codes of girths 6 and 8. The average BER and FER (frame error rate) over all the codes are shown in the figure.

We observe that removing short loops gives significant improvements in the performance of the semi-random LDPC codes. For low SNR values, the loop removal has little effect on performance. The subtleties of the loops' effects on belief propagation are irrelevant when the noise level is so high [56]. As the noise level decreases a noticeable difference between the three lines emerges, and this difference increases as the SNR increases.

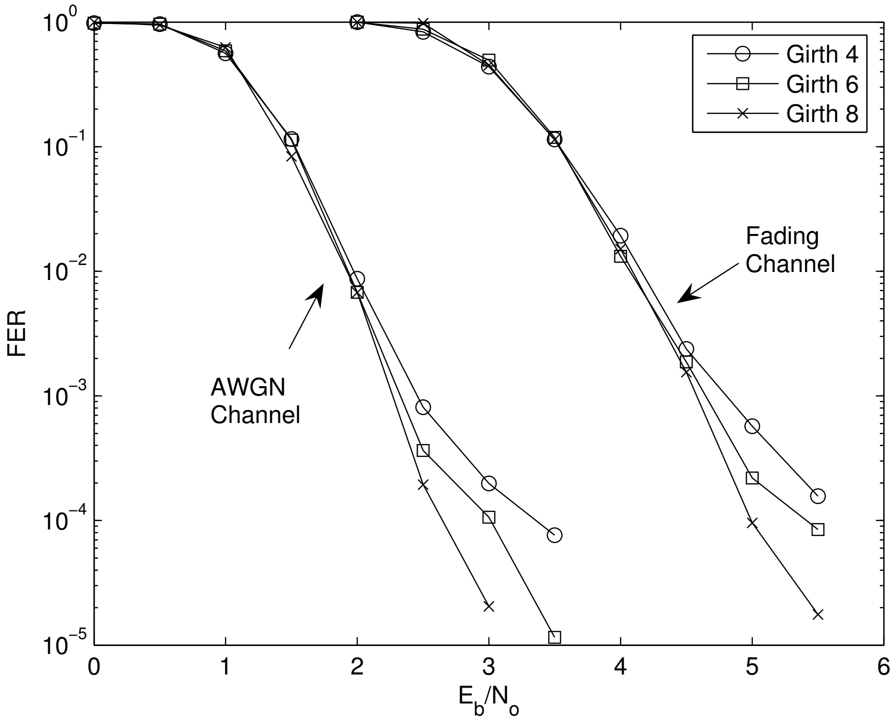
It can be seen that the codes with larger girth have lower error floors. This can be explained by the notion of stopping sets: a *Stopping set* S is a subset of V, the set of variable nodes, such that all neighbors of S are connected to S at least twice [60]. Stopping sets have special significance for the Binary Erasure Channel (BEC) as the erased variable nodes which form a stopping set cannot be 'estimated' by the iterative decoder (their APP LLR values remain at zero) [60]. The role of stopping sets is easily translated to non-erasure scenarios where variables with poor observation reliability are analogous to erasures [69]. The *stopping number* of a code is the size of its smallest stopping set, and the stopping number lower bounds the minimum distance of the code [61]. The stopping number of a code can be increased by increasing its girth, and hence codes with larger girth have lower error floors [61].

Chapter Summary

In this chapter, an algorithm for the removal of small loops from semi-random LDPC codes has been proposed. The removal of small loops leads to an increase in the girth of the code, which consequently increases the d_{min} of the code. Hence, the performance of the code improves, specially in the error floor (high SNR) region. Simulation results verify that codes of high girth (hence larger d_{min}) perform better in the error floor region as compared to codes of smaller girth.



(a)



(b)

Figure 6.4: The performance of Semi-Random Codes, n=1000, R=1/2, max-iterations = 50, (a) BER (b) FER

Chapter 7

Conclusion

In this chapter we first briefly review the main contributions of the thesis and then propose some directions for future research.

7.1 Summary of Contributions

This thesis has contributed to the field of LDPC coding in a number of ways. In particular, we have produced results applicable to the design of RC-LDPC codes and their applications in wireless networks. The results apply to LDPC codes that are decoded using the iterative belief-propagation algorithm.

First, we have proposed an algorithm for the design of good puncturing patterns for regular LDPC codes that results in a punctured code with large girth. A major factor that affects the performance of finite-length LDPC codes is the code's girth.

In the presence of loops, the sum-product algorithm becomes suboptimal even if the block length is large. Increasing the girth by removing small loops improves performance. We have shown that the punctured codes designed using this algorithm outperform randomly punctured codes. In this direction, we have proposed an algorithm for removing small loops from semi-random LDPC codes in order to increase the girth of the code. It was shown that the application of the algorithm leads to semi-random LDPC codes with low error floors.

As compared to regular LDPC codes, semi-random LDPC codes offer the major advantage of low encoding complexity. Furthermore, the structure of the parity-check matrix enables the design of rate-compatible codes that outperfrom rate-compatible regular codes. We have proposed puncturing patterns for semi-random LDPC codes which offer good performance both in the waterfall and the error floor regions. In addition, we have also proposed two efficient methods for extending semi-random LDPC codes. One of the methods proposed for extending offers good performance, while the other method performs even better, albeit at a higher code-design complexity. A Type-II Hybrid ARQ system utilizing rate-compatible semi-random LDPC codes designed using the proposed puncturing and extending techniques was shown to outperform an existing system based on regular LDPC codes.

Coded cooperation diversity achieves impressive gains over the non-cooperative scheme by the sharing of antennas of users to achieve uplink transmit diversity. In this work, we have investigated coded cooperation schemes based on rate-compatible semi-random LDPC codes and have shown that cooperation leads to improved performance even when the interuser channel quality is worse than the quality of the uplink channel. It was shown that the performance gains achieved via cooperation improve as the quality of the interuser channel improves. Furthermore, the effect of varying the cooperation level has been investigated, and a cooperation level has been proposed for semi-random LDPC codes that leads to the best performance over a variety of interchannel conditions.

7.2 Future Work

In continuation of this work, there are a number of problems that can be the subject of future research. Below is a short list of some of the possible directions of research.

In Chapter 2 a *stopping set* was defined. The significance of stopping sets is that if a set of variable nodes that form a stopping set are punctured (e.g if variable nodes v_1, v_6, v_{10} in a particular parity-check matrix form a stopping set and *all* these nodes are punctured), then the iterative decoder cannot correct them even after an infinite number of decoder iterations [60]. Therefore, care must be taken while puncturing to ensure that any stopping set is not punctured completely. This requires the identification of unique stopping sets. However, there is no algorithm in the literature that is able to identify all stopping sets in a parity-check matrix. The development and application of an algorithm that is able to identify unique stopping sets would lead to a significant improvement in the performance of punctured LDPC codes.

It was shown in [48] that increasing the cluster size for a system employing coded cooperation diversity based on RCPC codes does not necessarily lead to an improvement in the performance. The analysis showed that among the different cooperation scenarios, the two extreme scenarios of *no cooperation* (no user is able to decode any of his partners) and *full cooperation* (each user is able to decode all of his partners) have the largest probabilities. It was shown that increasing the cluster size leads to an increase in the probability of no cooperation, and a decrease in the probability of full cooperation. However, the system utilized convolutional codes and Maximum Likelihood decoding, and therefore the same results do not necessarily carry over for the case of LDPC codes with iterative decoding. Therefore, the effect of increasing the cluster size to more than two needs to be investigated for the case of systems employing LDPC codes. Furthermore, coded cooperation employing RC-LDPC codes can be compared with other cooperation strategies such as decode-and-forward and amplify-and-forward [77].

Appendix

Algorithm 1 Removing all possible loops from a SR graph H of size $(m \times n)$

```
$n \Leftarrow 4$
repeat
  $A \Leftarrow$ the Adjacency Matrix of H (from (6.1))
  for all elements $(i, j)$ of $A$ do
    if $A_{ij}^{n/2} \geq 2$ and $A_{ij}^{(n/2)-2} = 0$ then
      choose a $k$ such that $A_{ik}^{n/2-1} > 0$ and $A_{kj} = 1$ and $k > 2m + 1$
      $C_e \Leftarrow$ all columns $\{x : A_{jx}^{n-1} + A_{kx}^{n-1} = 0\}$
      $E_e \Leftarrow$ all edges of $H_d$ (the random part of $H$) with both endpoints in $C_e$
      if $E_e \neq \phi$ then
        Choose an edge $e'$ which belong to $E_e$
        Swap the check nodes of $e$ and $e'$ in $H$
        Return to repeat
      end if
    end if
  end for
  if no $n$-loops were found then
    $n \Leftarrow n + 2$
  end if
until no edges swapped since last repeat
```

Note: The ‘underlined’ portions are modifications to the algorithm given in [56].

Bibliography

[1] J. Li and K. Narayanan, "Rate-Compatible Low-Density Parity-Check Codes for Capacity-Approaching ARQ Scheme in Packet Data Communications," *Int. Conf. on Comm., Internet, and Info. Tech. (CIIT)*, November 2002.

[2] S. Lin and D. Costello, *Error Control Coding.* Prentice Hall, 2nd ed., 2004.

[3] R. G. Gallager, "Low-Density Parity-Check Codes," *IRE Transactions on Information Theory*, vol. IT-8, pp. 21–28, January 1962.

[4] S. Chung, G. D. F. Jr., T. Richardson, and R. Urbanke, "On the Design of Low-Density Parity-Check Codes Within 0.0045 dB of the Shannon Limit," *IEEE Communication Letters*, vol. 5, pp. 58–60, February 2001.

[5] M. Yazdani and A. Banihashemi, "On Construction of Rate-Compatible Low-Density Parity-Check Codes," *IEEE Communications Letters*, vol. 8, March 2004.

[6] "Flarion Vector-LDPC," (Available [Online]: www.flarion.com/products/whitepapers/Vector-LDPC.pdf).

[7] T. S. Rappaport, *Wireless Communications: Principles and Applications.* Pearson Education, Inc., 2nd ed., 2002.

[8] J. Proakis, *Digital Communications.* McGraw-Hill, 4th ed., 2001.

[9] S. Rice, "Statistical Properties of a Sine Wave Plus Random Noise," *Bell Systems Technical Journal*, vol. 27, pp. 109–157, January 1948.

[10] M. Nakagami, *"The m-Distribution- A General Formula of Intensity Distribution of Fading".* in Statistical Methods in Radio Wave Propagation, NY, USA: W. C. Hoffman (ed.), Pergamon Press, 1960.

[11] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp. 397–423 and 623–656, July/Oct 1948.

[12] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding," *Proc.ICC '93, Geneva, Switzerland*, pp. 1064–1070, May 1993.

[13] S. A. Zummo, *Coding and Channel Estimation for Block Fading Channels.* PhD thesis, University of Michigan, Ann Arbor, USA, June 2003.

[14] M. Ardakani, *Efficient Analysis, Design and Decoding of Low-Density Parity-Check Codes.* PhD thesis, University of Toronto, 2004.

[15] R. M. Tanner, “A Recursive Approach to Low Complexity Codes,” *IEEE Transactions on Information Theory*, vol. 27, pp. 533–547, Sept 1981.

[16] N. Wiberg, *Codes and Decoding on General Graphs.* PhD thesis, Linkoping University, Sweden, 1996.

[17] F. R. Kschischang and B. J. Frey, “Iterative Decoding of Compound Codes by Probability Propagation in Graphical Models,” *IEEE J. Select. Areas Commun.*, vol. 16, pp. 219–230, Feb 1998.

[18] R. J. McEliece, D. J. C. MacKay, and J.-F. Cheng, “Turbo Decoding as an Instance of Pearls“Belief Propagation” Algorithm,” *IEEE J. Select. Areas Commun.*, vol. 16, pp. 140–152, Feb 1998.

[19] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* San Mateo, CA:: Morgan Kaufmann Publishers, 1998.

[20] D. J. C. MacKay and R. M. Neal, “Near Shannon Limit Performance of Low Density Parity Check Cpdes,” *IEEE Electronics Letters*, vol. 32, pp. 1645–1646, Aug 1996.

[21] M. Sipser and D. A. Spielman, “Expander Codes,” *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1710–1722, 1996.

[22] T. J. Richardson and R. Urbanke, “The Capacity of Low-Density Parity-Check Codes Under Message-Passing Decoding,” *IEEE Transactions on Information Theory*, vol. 47, pp. 599–618, February 2001.

[23] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, “Improved Low-Density Parity-Check Codes Using Irregular Graphs,” *IEEE Transactions on Information Theory*, vol. 47, pp. 585–598, Feb 2001.

[24] A. Shokrollahi, “New Sequence of Linear Time Erasure Codes Approaching the Channel Capacity,” *Proc. the International Symposium on Applied Algebra, Algebraic Algorithms*, no. 1719, pp. 65–67, 1999.

[25] A. Shokrollahi, “Capacity-achieving Sequences,” *IMA Volumes in Mathematics and its Applications*, vol. 123, pp. 153–166, 2000.

[26] H. E. Gamal and A. R. Hammons, “Analyzing the Turbo Decoder Using the Gaussian Approximation,” *IEEE Transactions on Information Theory*, vol. 47, pp. 671–686, Feb 2001.

[27] S. Chung, T. Richardson, and R. Urbanke, “Analysis of Sum-Product Decoding of Low-Density Parity-Check Codes using a Gaussian Approximation,” *IEEE Transactions on Information Theory*, vol. 47, pp. 657–670, February 2001.

[28] S. ten Brink, “Convergence of Iterative Decoding,” *IEEE Electronics Letters*, vol. 35, pp. 806–808, May 1999.

[29] S. ten Brink, “Convergence Behavior of Iteratively Decoded Parallel Concatenated Codes,” *IEEE TRANSACTIONS ON COMMUNICATIONS*, vol. 49, pp. 1727–1737, October 2001.

[30] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Prentice Hall, 1995.

[31] D. M. Mandelbaum, “An Adaptive-Feedback Coding Scheme Using Incremental Redundancy,” *IEEE Transactions on Information Theory*, vol. IT-20, pp. 388–389, May 1974.

[32] J. Hagenauer, “Rate-Compatible Punctured Convolutional Codes (RCPC codes) and Their Applications,” *IEEE Transactions on Information Theory*, vol. 36, pp. 389–400, April 1988.

[33] G. Clark and J. Cain, *Error-Correcting Coding for Digital Communications*. Plenum Press, 1981.

[34] E. Telatar, “Capacity of Multi-antenna Gaussian Channels,” *European Transactions Telecommun*, vol. 10, pp. 585–595, November 1995.

[35] V. Tarokh, N. Seshadri, and A. Calderbank, “Space-Time Codes for High Data Rate Wireless Communication: Performance Criterion and Code Construction,” *IEEE Transactions on Information Theory*, vol. 44, pp. 744–765, March 1998.

[36] A. Sendonaris, E. Erkip, and B. Aazhang, “User Cooperation Diversity - part I: System Description,” *IEEE Transactions on Communications*, vol. 51, pp. 1927–1938, January 2003.

[37] A. Sendonaris, E. Erkip, and B. Aazhang, “User Cooperation Diversity - Part II: Implementation Aspects and Performance Analysis,” *IEEE Transactions on Communications*, vol. 51, pp. 1939–1948, January 2003.

[38] T. Hunter and A. Nosratinia, “Coded cooperation under slow fading, fast fading, and power control,” *Asilomar Conference on Signals, Systems and Computers*, pp. 118–122, November 2002.

[39] T. Hunter and A. Nosratinia, “Performance analysis of coded cooperation diversity,” *IEEE International Conference on Communication, ICC*, pp. 2688–2692, June 2003.

[40] M. Janani, A. Hedayat, T. Hunter, and A. Nosratinia, “Coded Cooperation in Wireless Communications: Space-Time Transmission and Iterative Decoding,” *IEEE Transactions on Communications*, vol. 52, pp. 79–81, February 2004.

[41] J. B. Cain, G. C. Clark, and J. M. Geist, “Punctured Convolutional Codes of rate (n - l)/n and Simplified Maximum Likelihood Decoding,” *IEEE Transactions on Information Theory*, vol. IT-25, pp. 97–100, January 1979.

[42] D. N. Rowitch and L. B. Milstein, "On the Performance of Hybrid FEC/ARQ Systems Using Rate-Compatible Punctured Turbo (RCPT) Codes," *IEEE Transactions on Communications*, pp. 948–959, June 2000.

[43] M. Kousa and A. Mugaibel, "Puncturing Effects on Turbo Codes," *IEE Proceedings on Communication*, vol. 149, June 2002.

[44] A. Mugaibel and M. Kousa, "Evaluation of Transfer Functions for Punctured Turbo Codes," *IEE Electronics Letters*, vol. 36, April 2000.

[45] F. Babich, G. Montorsi, and F. Vatta, "Some Notes on Rate-Compatible Punctured Turbo Codes (RCPTC) Design," *IEEE Transactions on Communications*, vol. 52, May 2004.

[46] J. Ha, J. Kim, and S. McLaughlin, "Rate-Compatible Puncturing of Low-Density Parity-Check Codes," *IEEE Transactions on Information Theory*, vol. 50, November 2004.

[47] J. Ha, J. Kim, D. Klinc, and S. McLaughlin, "Rate-Compatible Punctured Low-Density Parity-Check Codes With Short Block Lengths," *IEEE Transactions on Information Theory*, vol. 52, pp. 728–738, February 2006.

[48] S. Zummo, "Performance Analysis of Coded Cooperation Diversity in Wireless Networks," *Journal on Wireless Communications and Mobile Computing (WCMC)*, To appear, 2006.

[49] T. J. Richardson and R. Urbanke, “Design of Capacity Approaching Irregular Low-Density Parity-Check codes,” *IEEE Transactions on Information Theory*, vol. 47, Feb 2001.

[50] D. J. C. MacKay, “Good Error-Correcting Codes Based on Very Sparse Matrices,” *IEEE Transactions on Information Theoy*, vol. 45, pp. 399–431, March 1999.

[51] M. A. Landolsi, “Semi-random LDPC Codes for CDMA Communication over Nonlinear Band-limited Satellite Channels,” *International Journal of Satellite Communications and Networking (IJSCN)*, To appear, July 2006.

[52] E. Yeo, B. Nikolic, and V. Anantharam, “Iterative Decoder Architectures,” *IEEE Communications Magazine*, August 2003.

[53] D. J. C. MacKay and R. M. Neal, “Good Error-Correcting Codes Based on Very Sparse Matrices,” *Cryptography and Coding. 5th IMA Conf. C. Boyd, Ed., LNCS, no. 1025,Berlin: Springer*, pp. 100–111, 1995.

[54] *Digital Video Broadcasting (DVB).* European Standard (Telecommunications series) Std. ETSI EN 302 307 V1.1.1, 2004.

[55] *Optional B-LDPC coding for OFDMA PHY.* IEEE Std. IEEE 802.16e-04/78, 2004.

[56] J. McGowan and R. Williamson, “Loop Removal from LDPC Codes,” *Information Theory Workshop, Paris, France*, March 31 -April 4 2003.

[57] L. Ping, W. K. Leung, and N. Phamdo, “Low-Density Parity-Check codes with Semi-Random Parity Check Matrix,” *IEE Electronics Letters*, vol. 35, pp. 38–39, January 1999.

[58] J. Barry, *Digital Communication.* Springer, 3rd ed., 2003.

[59] W. Ryan, “An Introduction to LDPC Codes,” tech. rep., University of Arizona, 2001.

[60] C. Di, D. Proietti, E. Telatar, T. Richardson, and R. Urbanke, “Finite Length Analysis of Low-Density Parity-Check Codes on the Binary Errasure Channel,” *IEEE Transactions on Information Theory*, vol. 48, pp. 1570–1579., June 2002.

[61] K. A.Orlitsky, R.Urbanke and J.Zhang, “Stopping Sets and the Girth of Tanner Graphs,” *International Symposium on lnformation Theory*, June 2002.

[62] T. Etzion, A. Trachtenberg, and A. Vardy, “Which Codes Have Cycle-Free Tanner Graphs?,” *IEEE Transactions on Information Theory*, vol. 45, pp. 2173–2181, September 1999.

[63] T. J. Richardson and R. Urbanke, “Efficient Encoding of Low-Density Parity-Check Codes,” *IEEE Transactions on Information Theory*, vol. 47, pp. 638–656, February 2001.

[64] S. Johnson and S. Weller, “A Family of Irregular LDPC Codes With Low Encoding Complexity,” *IEEE Communications Letters*, vol. 7, pp. 79–81, February 2003.

[65] Y. Kou, S. Lin, and M. Fossorier, “Low-Density Parity-Check Codes Based on Finite Geometries: A Rediscovery and New Results,” *IEEE Transactions on Information Theory*, vol. Volume 47, pp. 2711 – 2736, Nov 2001.

[66] D. Divsalar, H. Jin, and R. J.McEliece, “Coding Theorems for Turbo-Like Codes,” *Proceedings of the 36th Annual Allterton Conference on Communications, Control and Computing*, 1998.

[67] H. Jin, A. Khandekar, and R. J.McEliece, “Irregular Repeat Accumulate Codes,” *Proceeding of the 2nd International Symposium on Turbo Codes and Related Topics*, pp. 1–8, 2000.

[68] M. Yang, W. E. Ryan, and Y. Li, “Design of Efficiently Encodable Moderate-Length High-Rate Irregular LDPC Codes,” *IEEE Transactions on Communications*, vol. 52, pp. 564–571, April 2004.

[69] T. Tian, C. Jones, J. Villasenor, and R. Wesel, “Construction of Irregular LDPC Codes with Low Error Floors,” *IEEE International Conference on Communications*, vol. 5, pp. 3125–3129, 2003.

[70] Y. Mao and A. H. Banihashemi, “A Heuristic Search for Good Low-Density Parity-Check Codes at Short Block Lengths,” *IEEE International Conference on Communications*, June 2001.

[71] M. H. Alsuwaiyel, *Algorithms: Design Techniques and Analysis.* World Scientific, 1999.

[72] R. Morelos-Zaragoza, *The Art of Error Correcting Coding.* John Wiley and Sons, 2002.

[73] G. L. Stuber, *Principles of Mobile Communication.* Boston, MA, USA: Kluwer Academic Publisher, 2nd ed., 2001.

[74] E. Al-Hussaini and A. Al-Bassiouni, “Performance of MRC Diversity Systems for the Detection of Signals with Nakagami Fading,” *IEEE Transactions on Communications*, vol. 33, pp. 1315–1319, December 1985.

[75] M. Nakagami, “The m-Distribution - A General Formula of Intensity Distribution of Rapid Fading,” in *Statistical Models of Radio Wave Propagation* (W. C. Hoffman, ed.), pp. 3–36, 1960.

[76] W. C. Jakes, *Microwave Mobile Communications.* New Jersy, USA: IEEE Press, 1974.

[77] J. Laneman, G. Wornell, and D. Tse, "An Efficient Protocol for Realizing Cooperative Diversity in Wireless Networks," *IEEE International Symposium on Information Theory, ISIT*, 2001.

Vitae

- Shaikh Faisal Zaheer.
- Born in Lahore, Pakistan on January 1, 1978.
- Received Bachelor of Science (B.Sc) degree in Electronics Engineering from Ghulam Ishaq Institute of Engineering Sciences and Technology, Topi, Pakistan in 2000.
- Joined King Fahd University of Petroleum and Minerals in September 2003.
- Email: sfzaheer@kfupm.edu.sa