Beiträge aus der Informationstechnik

Mobile Nachrichtenübertragung
Nr. 81

Najeeb Ul Hassan

On Decoding of Spatially Coupled LDPC Codes under Latency Constraints

VOGT

Dresden 2016

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.dnb.de abrufbar.

Bibliographic Information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available on the Internet at http://dnb.dnb.de.

Zugl.: Dresden, Techn. Univ., Diss., 2016

Die vorliegende Arbeit stimmt mit dem Original der Dissertation „On Decoding of Spatially Coupled LDPC Codes under Latency Constraints“ von Najeeb Ul Hassan überein.

Gesetzt vom Autor

ISBN 978-3-95947-003-2

Jörg Vogt Verlag
Niederwaldstr. 36
01277 Dresden
Germany

Phone: +49-(0)351-31403921
Telefax: +49-(0)351-31403918
e-mail: info@vogtverlag.de
Internet : www.vogtverlag.de

Technische Universität Dresden

On Decoding of Spatially Coupled LDPC Codes under Latency Constraints

Najeeb Ul Hassan

von der Fakultät Elektrotechnik und Informationstechnik
der Technischen Universität Dresden

zur Erlangung des akademischen Grades

Doktoringenieur

(Dr.-Ing.)

genehmigte Dissertation

Vorsitzender:	Prof. Dr. sc. techn. habil.
	Dipl. Betriebswissenschaften Frank Ellinger
Gutachter:	Prof. Dr.-Ing. Dr. h.c. Gerhard Fettweis
	Prof. Dr.-Ing. Stephan ten Brink

Tag der Einreichung: 03.09.2015
Tag der Verteidigung: 21.03.2016

Abstract

This dissertation focuses on the decoding of low-density parity-check (LDPC) convolutional codes, also known as spatially coupled LDPC (SC-LDPC) codes. These codes combine the properties of both LDPC block codes (good performance for long block lengths) and convolutional codes (good performance for short block lengths), and are shown to be suitable for applications that allow medium to large block lengths. The first part of the dissertation deals with efficient decoding of SC-LDPC codes under practical constraints like decoding latency and decoding complexity while keeping their good performance advantage over LDPC block codes.

We divide the decoders in two classes based on their resulting decoding latency and complexity; (i) *block decoding* runs belief propagation algorithm over complete chain of coupled codeword resulting in a large decoding latency and decoding complexity, (ii) *windowed decoding*, on the other hand, exploits the convolutional structure of the coupled code making decoding latency and complexity independent of the length of the code. We consider *protograph* based codes, since this allow us to assess the performance of code ensemble, rather than the performance of a single code. Both asymptotic and finite length analysis are performed to show the superiority of SC-LDPC codes over LDPC block and convolutional codes for medium to large latency range. For very short latency requirements, convolutional codes decoded using Viterbi decoder are found to be suitable.

In order to reduce the decoding complexity, traditionally used uniform serial decoding schedules are applied within the window. However, our results show that this only gains 18% reduction in decoding complexity compared to parallel decoding schedule. We propose *non-uniform window schedules* which are based on the observed decoding convergence behavior within a window. These result up to 50% reduction in decoding complexity compared to uniform window schedules without any loss in performance.

Non-uniform schedules require estimates of error probability during the iterative process and hence the resulting schedule is time variant. However, based on conclusions drawn using the asymptotic analysis, we propose a pragmatic decoding schedule that does not require any additional calculation within the decoding process and with little loss in performance reduces the decoding complexity by 45%

compared to the uniform schedule. Finally, taking into account the non-uniform nature of the update rule, we propose an *implementation/sequencing strategy* such that the decoding throughout is doubled without significantly increasing the hardware requirements.

The second part of the dissertation deals with the application of SC-LDPC codes for *block-fading* channel. Block-fading channel is a suitable model for *mobile-radio channel*, where the channel state stays constant for multiple symbol durations (N_c). Hence a codeword of length N is divided into F equal parts where $F = N/N_c$. Codes on block-fading channel are characterized by their (i) outage probability, P_{out}, and (ii) diversity order, d. For block codes, a special structure is required to guarantee a required d. However in convolutional codes, it largely depends on the constraint length (or memory) of the code.

We present bounds on the maximum achievable diversity for SC-LDPC codes decoded using maximum likelihood (ML) decoder and a sub-optimal iterative decoder. For a code decoded by an ML decoder, it turns out that d is related to the *blockwise minimum Hamming distance* $d_{\min}$ of the code. However, since SC-LDPC (in general LDPC) codes are decoded using a sub-optimal iterative decoder, the maximum diversity under iterative decoding is calculated by the *blockwise stopping distance* $s_{\min}$ of the code, where $s_{\min} \leq d_{\min}$. Here the main contribution is an algorithm to design a protograph for which $s_{\min} = d_{\min}$.

The *root-LDPC* code is one example of block codes that have a special structure to achieve $d = F$. However, these codes require $R = 1/F$, i.e., to achieve high diversity order, code rate has to be decreased. The major advantage of the proposed SC-LDPC codes is that these do not require a special structure to achieve the required diversity. Furthermore, diversity order can be increased by increasing the memory of the code and without decreasing the code rate. Another advantage of SC-LDPC codes is robustness against synchronization offset, where the loss in performance due to synchronization offset can be compensated by increasing decoding latency i.e., W for window decoder. On the other hand, root-LDPC codes have to be designed specifically for a given F and their performance drastically degrades in the presence of synchronization offset.

Acknowledgment

This thesis comprises of the results generated during my work at the Vodafone Chair Mobile Communications Systems at Technische Universität Dresden between 2011 and 2015.

First of all i would like to express my sincere gratitude towards my supervisor Prof. Gerhard Fettweis for giving me the opportunity to join his group. It is due to his invaluable guidance and motivation that kept me going in the right direction. I cannot be any less grateful to have Prof. Michael Lentmaier as my co-supervisor whose constant guidance and mentoring had made this journey possible. He stimulated my interest in the field of coding theory and helped improve my writing abilities. I also thank Michael for reviewing the first draft of the thesis that helped shaped it to its current state. Additionally, i am grateful to my collaborators Prof. Daniel J. Costello Jr., Prof. Ali E. Pusane, and Prof. Iryna Andriyanova for the fruitful discussions and suggestions in last years.

I also would like to thank all colleagues from the chair. Just to name few, Lukas, Björn, Walter, Nicola, Ines, Steffen, Rohit, Meik, Jan, and Patrick. It is you that made these years memorable.

I am also grateful to my parents for their faith in me and for allowing me to follow my ambitious dreams (Shukrya ami aur abu). I also like to thank my wife, Mariam. Her support, encouragement and unwavering love made me survive these last years. I am also grateful to have my son, Rayyan, whose magical smile always make my work tiredness go away.

Contents

List of Symbols

$\mathbf{A}$	Adjacent matrix
α	Fading coefficient
$\mathbf{B}$	Protograph base matrix
$\mathbf{B}_i$	i-th component matrix
$\mathbf{B}_{[1,L]}$	Base matrix of a protograph based terminated SC-LDPC code
$\mathbf{B}_W$	Part of the base matrix within a window decoder of size W
$\mathcal{C}$	Set of check nodes in a Tanner graph
d	Code diversity
d_{IT}	Iterative diversity
$d_{\min}$	Blockwise minimum Hamming distance of the code
$d_{\min}^{\mathbf{B}}$	Blockwise minimum Hamming distance of the protograph $\mathbf{B}$
$d_{\min}^{\mathbf{H}}$	Blockwise minimum Hamming distance of the code with parity-check matrix $\mathbf{H}$
$\mathbb{E}[\cdot]$	Expectation
E_b/N_0	Ratio of energy per information bit and single sided power spectral density of the noise
ϵ	Erasure probability
F	Number of fading coefficients in a code block
$\mathcal{G}$	Tanner graph
γ	SNR
$\mathbf{H}$	Parity-check matrix
$\mathbf{H}_{[1,L]}$	Parity-check matrix of a terminated SC-LDPC code
$I(\cdot)$	Mutual information
$\mathcal{I}_0$	Induced graph
$I_{\max}$	Maximum number of iterations
i, j, k	Index variables
J	Number of 1's in every column of matrix $\mathbf{H}$
(J, K)	Regular LDPC block code
$(J, K, L, m_{\mathrm{cc}})$	Regular SC-LDPC code

$J_{\max}$	Maximum allowed number of edges in a column of $\mathbf{H}$
$J_{\min}$	Maximum allowed number of edges in a column of $\mathbf{H}$
K	Number of 1's in every row of matrix $\mathbf{H}$
$K_{\max}$	Minimum required number of edges in a row of $\mathbf{H}$
$K_{\min}$	Minimum required number of edges in a row of $\mathbf{H}$
L	Termination length
λ_i	Fraction of edges incident on degree-i variable node
$L_{\mathrm{ch}}(\cdot)$	Channel LLR value
L_{v_k,c_j}	Message from v_k to c_j
L_{c_j,v_k}	Message from c_j to v_k
$L_{\mathrm{out}}(\cdot)$	Output LLR
M	Rows of matrix $\mathbf{H}$ or number of check nodes in a Tanner graph
m_{cc}	Coupling memory of SC-LDPC code
m_{WD}	Memory of the window decoder
N	Columns of matrix $\mathbf{H}$ or number of variable nodes in a Tanner graph or Length of a code block
$\mathbb{N}$	Set of natural numbers
$\mathcal{N}(\cdot)$	Neighborhood of a node
n_c	Number of check nodes in a protograph
N_c	Number of bits in a coherence time
n_v	Number of variable nodes in a protograph
$\mathbb{P}[\cdot]$	Probability
$P_{\mathrm{b}}(\cdot)$	Probability of error
$\hat{P}_{\mathrm{b}}(\cdot)$	Estimate of probability of error
P_{b}^{I}	Probability of error at Ith iteration
$P_{\mathrm{b}}^{\max}$	Target error probability
P_{DEO}	Density evolution outage probability
P_{R}	Periodic repeat parameter
P_{we}	Probability of word error under ML decoding
R	Code rate/ transmission rate
ρ_j	Fraction of edges incident on degree-j check node
$\mathbf{s}$	Syndrome vector
σ^2	Noise variance
σ_{BP}^{*}	BP threshold of an ensemble
σ^{*}	Threshold of an ensemble
σ_{Sh}	Shannon limit
σ_{W}^{*}	Window decoding threshold of an ensemble
$s_{\min}$	Blockwise minimum stopping distance

t	Index for time instant
θ	Fractional improvement parameter
T_{b}	Latency in terms of information bits for the block decoder
T_{v}	Latency in terms of information bits for the Viterbi decoder
T_{WD}	Latency in terms of information bits for the window decoder
$\mathbf{u}$	Information word
$\hat{\mathbf{u}}$	Estimate of the information word
U_{avg}	Average decoding complexity
U_t	Number of variable node updates at time t
$\mathcal{V}$	Set of variable nodes in a Tanner graph
$\mathbf{v}$	Codeword
$\mathbf{v}_{[1,L]}$	Codeword of a terminated SC-LDPC code
$\mathbf{v}_{[i,j]}$	Vector consisting of code blocks $\mathbf{v}$ starting from index i until index j
w	Code block index within a window
W	Window Size
$W(\cdot)$	Weight enumerator polynomial
$W_{\max}$	Minimum possible window size
$W_{\min}$	Minimum allowed window size
$\mathbf{x}$	Transmitted block
$\mathbf{y}$	Received noisy block
$\mathbf{y}_{[i,j]}$	Vector consisting of received blocks $\mathbf{y}$ starting from index i until index j
Z	Lifting factor
$(\cdot)^T$	Transpose operation

List of Abbreviations

5G	Fifth generation
AWGN	Additive white Gaussian noise
BEC	Binary erasure channel
BER	Bit error rate
BP	Belief propagation
BPSK	Binary phase shift keying
CC	Convolutional codes
CSI	Channel state information
DE	Density evolution
DEO	Density evolution outage
DVB	Digital video broadcasting
FEC	Forward error correction
HAEC	Highly adaptive energy-efficient computing
LDPC	Low-density parity-check
LLR	Log likelihood ratio
MAP	Maximum a posteriori
ML	Maximum likilihood
NU	Non-uniform
OFDM	Orthogonal division frequency multiplexing
PEG	Progressive edge growth
RCA	Reciprocal channel approximation
SC-LDPC	Spatially coupled LDPC
SNR	signal-to-noise ratio
U	Uniform
WER	Word error rate
WiGig	Wireles Gigabit alliance
WiMAX	Worldwide Interoperability for Microwave Access

List of Figures

List of Tables

Chapter 1

Introduction

Forward error correction (FEC) codes add redundant bits to information bits such that a reliable communication over a noisy channel is viable. The roots of FEC coding date back to Shannon's pioneer work [Sha48] in 1948, where he proved that a coded transmission with rates close to capacity is possible with arbitrarily low error rate given long codes and an optimal decoder. However, the questions related to the practical coding scheme and the length of the codeword required to achieve good performance was not answered by Shannon. Note that the length of the codeword determines the information delay and is a critical parameter for the choice of coding scheme in any system. It has been therefore the main focus of the researchers to find codes with very good performance while fulfilling the latency requirements imposed by different applications.

In general, it is possible to construct a code that operates very close to the limits of the channel (also known as Shannon limit) by assuming very long codewords. However, in practice one has to use a codeword of finite length, hence introducing a gap between the Shannon limit and code performance. Naturally one way to reduce this gap is to increase the codeword length. The codeword length has a direct impact on both decoding latency and decoding complexity and hence different applications require codes with different lengths depending on the available resources and latency requirements. Hence a system designer is not only required to provide codes with different rates but also with different lengths to get a trade-off between latency (length of a codeword) and performance. As an example, in WiMAX for each rate, blocks ranging from 576 to 2304 bits are proposed.

Encoding information bits to generate code bits can be performed either *blockwise* (block codes) or in *continuous* (convolutional codes) manner. In case of blockwise encoding, a unique code is required to define blocks of different lengths. However,

for continuous encoding, one can simply run the encoder longer to obtain a codeword of different lengths. Hence in convolutional codes, the problem of designing codes of different length can be simplified compared to block codes.

Scope and Outline of this Work

Gallager introduced a class of linear codes which are described by a sparse parity-check matrix [Gal63] referred as low-density parity-check (LDPC) codes. Since these codes are constructed from sparse matrices, these can be decoded using a simple and practical iterative decoder (see Section 2.2.2). Hence as a result, in general, the decoding complexity per decoded bit is independent of the length of the code when the number of decoding iterations is fixed. Thanks to this property of LDPC codes, various standards that deals with high information transfer rate, e.g., DVB, WiMAX, WiGig, propose LDPC codes which allow them to operate close to the Shannon limit using long block lengths. In contrast to this, several applications such as, wireless sensors, machine-to-machine communications, etc., because of their low information transfer rate, propose convolutional codes since they are known to be suitable when small block lengths are inevitable.

In this work, we combine the properties of both LDPC (good performance for long block lengths) and convolutional (good performance for short block lengths) codes, and propose to use a convolutional version of LDPC codes, otherwise known as spatially coupled LDPC (SC-LDPC) codes for applications that allow medium to large block lengths. The dissertation is structured as follows.

- In Chapter 2, we define LDPC and SC-LDPC codes. A structured way to construct codes using protograph is also described together with the iterative decoding algorithm used throughout the dissertation. Finally we show by using an example, the *threshold saturation* phenomenon of SC-LDPC codes.

- Chapter 3 starts by describing the decoding latency associated with the two possibilities to decode SC-LDPC codes namely; block decoding and window decoding. It turns out that using window decoding is efficient in terms of latency. However, this also results in some constraints on the protograph and are discussed using density evolution. Finally, computer simulation results of finite length codes, generated using a progressive edge growth algorithm (PEG), are presented that show the performance of the window decoder and a comparison with Viterbi decoded convolutional codes and LDPC block codes.

- In Chapter 4, decoding complexity of the window decoder is discussed. We propose non-uniform window schedules that utilize the structure of code to reduce the decoding complexity. The definition of the decoding complexity and a comparison between the window decoder and the block decoder in terms of decoding complexity shows that block decoding is not efficient for SC-LDPC codes in terms of complexity. Next non-uniform decoding schedules are proposed together with their density evolution and simulation results. Finally a practical pragmatic decoding schedule is introduced. The pragmatic schedule, with little loss in performance, is shown to be able to reduce decoding complexity by $\approx 50\%$ compared to the traditional decoding schedules. We also present an implementation of the proposed pragmatic schedule that allows to double the decoding throughput without doubling the requirement on hardware components.

- In Chapter 5, we demonstrate the suitability of SC-LDPC codes for mobile-radio (block-fading) channel. We start by defining the channel model and parameters used to characterize the channel. Next the existing solution for block-fading channel is described together with its limitations. We motivate the use of SC-LDPC codes for block-fading channel and provide an algorithm to design protographs suitable for block-fading channel. The performance evaluation is presented for the designed codes using both density evolution and computer simulations of finite length codes. Furthermore, using simulations we show that the proposed codes are robust against the synchronization offset between the transmitted bits and code bits.

- Finally we draw conclusions and provide some future research directions in Chapter 6.

The results and analysis is limited to regular LDPC codes, however, without loss of generality, these can be extended to irregular LDPC codes. Furthermore, the finite length codes used for simulations are not optimized and we use the PEG algorithm only to avoid cycles of length 4.

Chapter 2

Preliminaries

In the following, we describe the system model considered throughout the dissertation. LDPC block codes and their iterative decoding algorithm is detailed in Section 2.2. Section 2.3 introduces the convolutional version of LDPC codes, also known as spatially coupled LDPC codes, which is the main focus of this dissertation.

2.1 System Model

In order to make the nomenclature clear, we start with the definition of system model which will be considered throughout the dissertation. A simplified system model consisting of a source, channel encoder, modulator, physical channel, demodulator, channel decoder and a sink is considered (see Fig. 2.1). The blocks which are not directly associated to channel coding problems, e.g., channel estimation, channel equalization, analog-to-digital conversion etc, are not considered here for simplicity of the model. The source generates an information word $\mathbf{u} = [u_1, \ldots, u_k]$ of k information bits. The channel encoder maps the information word $\mathbf{u}$ to a codeword $\mathbf{v} = [v_1, \ldots, v_n]$ of length n, hence adding $n - k$ redundant bits to the information word. The ratio of k and n gives rate R of the code. The coded bit stream is then mapped to $\mathbf{x} = [x_1, \ldots, x_n]$[1] and transmitted over a noisy channel. We consider binary phase shift keying (BPSK) modulation where the BPSK symbols x_i are obtained from input symbols v_i as $x_i = 2v_i - 1 \in \{+1, -1\}$. Considering an additive while Gaussian noise (AWGN) channel[2], the received symbols have the

[1] we consider here only binary channel encoders and binary bit mapping, and hence the input and output length of the mapping block is of length n.

[2] we consider mainly AWGN channel, however, a slowly-varying fading channel together with AWGN is considered and detailed in Chapter 5.

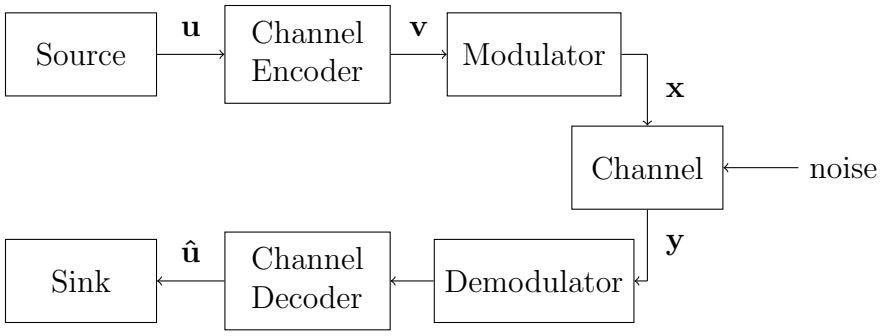


Figure 2.1: A communication system.

form,

$$y_i = x_i + n_i \ , \quad \forall \ \ i = 1, \ldots, n \ , \tag{2.1}$$

where n_i are independent and identically distributed Gaussian random variables with zero mean and variance $\sigma^2 = N_0/2$, and N_0 is the single sided power spectral density of the noise. The signal-to-noise ratio (SNR) is defined in terms of E_b/N_0, where E_b is the average energy per information bit. On the receiver side, log-likelihood ratios (LLRs) corresponding to the noisy received word $\mathbf{y} = [y_1, \ldots, y_n]$ are passed to the channel decoder that removes the redundancy added by the channel encoder and produces the estimates $\hat{\mathbf{u}}$ of the information word $\mathbf{u}$. The bit error rate (BER) is calculated using a Monte Carlo method based on the information word $\mathbf{u}$ and its estimate produced by the channel decoder $\hat{\mathbf{u}}$.

In general, any coding scheme can be compared against the limit of the channel, otherwise known as *Shannon limit*. In terms of the channel parameter σ, we denote the Shannon limit as σ_{Sh}. Furthermore, we define σ^* as a parameter for a code that represents a unique channel parameter such that for channels with $\sigma \leq \sigma^*$, decoding succeeds with high probability and with channels $\sigma > \sigma^*$ decoding fails with high probability. The σ^* denotes the *threshold* of the code. The code is referred as *capacity approaching* when σ^* is close to σ_{Sh}.

2.2 LDPC Codes

Low-density parity-check (LDPC) codes were first invented by Gallager in 1963 [Gal63] but had long been forgotten until they resurface in late 90's. Gallager's LDPC codes were defined by sparse parity-check matrices $\mathbf{H}$ (of dimension $M \times N$) that contained a fixed number of K and J non-zero values in every row and column, respectively, known as *regular* LDPC codes. It is due to the sparsity of the parity-check matrix that these codes are called low-density codes. Alternatively, a regular

LDPC code can be represented by using a bipartite graph (Tanner graph) consisting of M check and N variable nodes with each of degree K and J, respectively. The check nodes correspond to the set of parity-check constraints and the variable nodes correspond to the code bits. Referring to the parity-check matrix, check nodes and variable nodes represent the rows and columns of the parity-check matrix, respectively. Each 1 in $\mathbf{H}$ represents an edge connecting the corresponding variable and check node in the Tanner graph.

Definition 2.1. [Regular LDPC Code] A regular LDPC code is determined by the condition that every code bit (variable node) participates in exactly J parity-check constraints (check nodes) and every parity-check constraint (check node) involves exactly K code bits (variable nodes). The resultant code is called (J, K)-regular LDPC code. ■

Since LDPC codes belong to a class of linear codes, every valid codeword must satisfy the condition,

$$\mathbf{H}\mathbf{v}^T = \mathbf{0} \ , \tag{2.2}$$

where T denotes transpose operation. By making use of (2.2), an encoder for an LDPC code can be realized using the matrix $\mathbf{H}$ (see, e.g., [JFZ99]). The rate of the resultant regular LDPC code can be given as,

$$R \geq 1 - \frac{J}{K} \ , \tag{2.3}$$

where the equality holds if and only if all the rows of matrix $\mathbf{H}$ are linearly independent. An example of a $(2, 3)$-regular LDPC code represented by its parity-check matrix $\mathbf{H}$ and the corresponding Tanner graph $\mathcal{G} = (\mathcal{C}, \mathcal{V}, \mathcal{E})$ is shown in Fig. 2.2. Here, $\mathcal{C} = [c_1, \ldots, c_M]$ and $\mathcal{V} = [v_1, \ldots, v_N]$[3] denote the set of check and variable nodes, respectively, and $\mathcal{E}$ is the set of edges in the Tanner graph. A check node is represented by a square and a variable node by a circle.

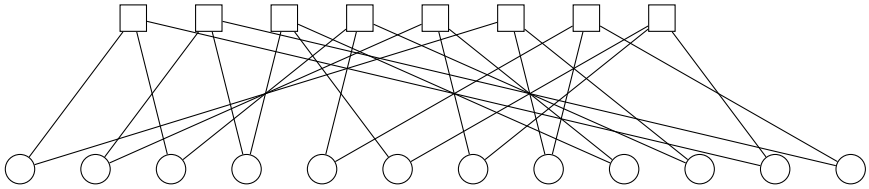
Irregular LDPC Codes

LDPC codes have been adopted in various wireless communication standards, e.g., WiMAX, IEEE 802.11n, etc. Many of these standards use LDPC codes as an optional choice together with turbo codes. However, recently in IEEE 802.11ad standard [IEE14] LDPC codes have been considered as the only choice due to

[3] since bits in a codeword are represented by variable nodes in a Tanner graph, we use the symbol v_i to denote the ith code bit and ith variable node.

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

(a) Parity-check matrix.



(b) Tanner graph.

Figure 2.2: An example of a $(2, 3)$-regular LDPC code.

their good performance and low complexity decoding over turbo codes, which is of importance particularly when considering applications with high data rates. The LDPC codes considered in the above mentioned standards are all *irregular LDPC codes*, i.e., the nodes in the Tanner graph are allowed to have different node degrees.

Definition 2.2. [Irregular LDPC Code] An irregular LDPC code can have any number of edges incident on a check or a variable node, i.e., the row or column weight of the parity-check matrix is not constant. The resultant ensemble of an irregular LDPC code can be characterized by edge perspective degree distribution polynomials,

$$\lambda(x) = \sum_i \lambda_i x^{i-1} \quad \text{and} \quad \rho(x) = \sum_j \rho_j x^{j-1} \ , \tag{2.4}$$

where λ_i denotes the fraction of edges that are incident on degree-i variable nodes and ρ_j denotes the fraction of edges incident on degree-j check nodes. ■

For regular LDPC codes, it can be shown that there exists a noticeable gap between the σ^* and the σ_{Sh}. However, this gap can be reduced when the node degrees are optimized, i.e., the degree distribution for an irregular code is optimized using a differential evolution algorithm [RSU01]. In general, such an optimization results in a large fraction of degree-2 variable nodes (λ_2). Although the degree-2 variable nodes in the graph result in small structures which consequently help the decoding

process, these also lead to low-weight codewords and hence higher error floor at large SNRs. Furthermore, the minimum distance of these optimized irregular codes grows only sub-linearly with the code length because of the large fraction of low degree variable nodes. As an example taken from [RSU01], the authors in [TJVW03] observed an error floor slightly below BER = 10^{-6} for an irregular LDPC code. Note that the irregular code designed in [RSU01] is 0.6 dB better in terms of asymptotic performance (threshold) compared to their counterpart regular LDPC code.

On the other hand, the regular codes constructed by Gallager enjoy the property that their minimum distance grows linearly with the code length and thus asymptotically have very low error floors. In Section 2.3, using regular LDPC codes and a technique known as *spatial coupling*, we show that performance can be pushed close to the Shannon limit without compromising the linear minimum distance growth. It is also important to mention here that with further optimization in the process of code construction, the error floor of irregular LDPC codes can be lowered. However, code construction is not a main theme of this dissertation and hence we do not consider designing and optimizing irregular LDPC codes. The techniques and algorithms devised here are demonstrated on regular codes. However, without loss of generality, these are applicable to irregular codes.

2.2.1 Structured LDPC Ensembles

In order to construct the parity-check matrix of an LDPC code, Thorpe in [Tho03] introduced a template graph, otherwise known as *protograph*. The protograph serves as a blue print for the parity-check matrix. A protograph is a small bipartite graph consisting of n_c check and n_v variable nodes and is represented by its bi-adjacency matrix $\mathbf{B}$, called *base matrix*. The base matrix consists of integer values and the matrix $\mathbf{H}$ is obtained by using a *graph lifting* operation where each 1 in $\mathbf{B}$ is replaced by a $Z \times Z$ permutation matrix and each 0 by a $Z \times Z$ zero matrix. The integer entries larger than 1 represent multiple edges between a pair of nodes and are replaced by a sum of permutation matrices. The resultant parity-check matrix consists of $M = Zn_c$ check and $N = Zn_v$ variable nodes. The permutation matrices can be generated using the progressive edge growth (PEG) algorithm introduced in [HEA05] such that short cycles are avoided, wherever possible.

The protograph defines a code ensemble instead of a particular LDPC code. This allow us to assess the performance of code ensemble, rather than the performance of a single code. The graph lifting operation guarantees that the final code inherits

properties of the protograph and thus protographs are very useful for analysis purposes. The structured codes have many practical advantages including, but not limited to, efficient hardware architectures for the encoders and decoders and easy realization of puncturing for high rate codes [MLPCJ12]. Let us illustrate the process of obtaining a parity-check matrix of an LDPC code from a protograph with a help of an example.

Example 2.1. In this example, we start from a protograph with base matrix $\mathbf{B}$ and demonstrate the graph lifting operation to obtain a parity-check matrix $\mathbf{H}$. Let us consider a protograph for a $(2, 3)$-regular LDPC code with $n_c = 4$ check nodes and $n_v = 6$ variable nodes as,

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix} . \tag{2.5}$$

The base matrix $\mathbf{B}$, similar to parity-check matrix $\mathbf{H}$, can be represented by a Tanner graph. Now considering a lifting factor $Z = 2$, i.e., each 1 in $\mathbf{B}$ is replaced by a 2×2 permutation matrix and each 0 by a 2×2 zero matrix. An example of a resultant parity-check matrix with $M = 8$ check and $N = 12$ variable nodes is given as,

$$\mathbf{H} = \left[\begin{array}{cc:cc:cc:cc:cc:cc} 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hdashline 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ \hdashline 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ \hdashline 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \end{array}\right] \tag{2.6}$$

■

Note that the matrix in (2.6) only gives one possible realization of an ensemble defined by protograph in (2.5). Another LDPC code using the same protograph can be seen in Fig. 2.2(a). The graph lifting operation can also be visualized using Z copies of the protograph and then permuting the edges between the nodes of the same type. For more detail, interested readers are referred to [Tho03].

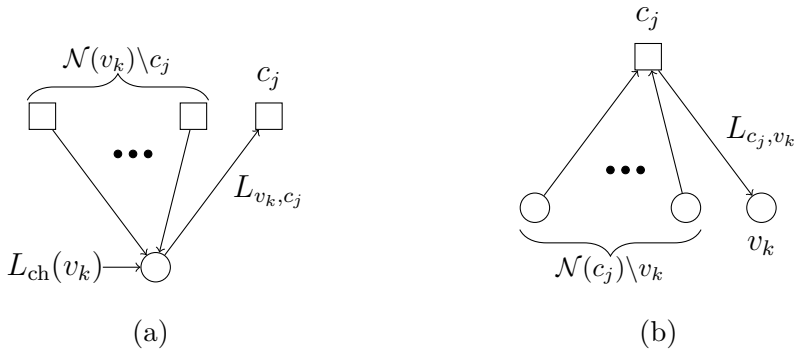


Figure 2.3: Calculation of the LLRs along the edge (a) variable node v_k to check node c_j and (b) check node c_j to variable node v_k.

2.2.2 Belief Propagation

As mentioned above, one of the benefits of having *low density* of ones in parity-check matrix is that LDPC codes can be decoded using an iterative message passing decoder. The decoding complexity increases with density of ones in $\mathbf{H}$, hence a low density is desirable. In case of the AWGN channel, messages in form of LLR values are exchanged between check and variable nodes in every iteration. Since the absolute value of an LLR represents the strength of our belief that its associated variable node assumes a particular (binary) value, a decoder applying this message passing algorithm is known as a *belief propagation* (BP) decoder. The calculation of LLR values at each step of the iterative decoding is detailed below.

Let $L_{\mathrm{ch}}(v_k)$ denote the input channel LLR for variable node v_k and $\mathcal{N}(v_k)$ denotes the set of check nodes connected to the variable node v_k. Similarly $\mathcal{N}(c_j)$ represents the set of variable nodes connected to the check node c_j. The extrinsic LLRs from the variable node v_k to the check node c_j and from the check node c_j to the variable node v_k are denoted as L_{v_k,c_j} and L_{c_j,v_k}, respectively, (see Fig. 2.3) and are calculated as follows,

$$L_{v_k,c_j} = L_{\mathrm{ch}}(v_k) + \sum_{l \in \mathcal{N}(v_k)\backslash c_j} L_{c_l,v_k} \quad , \tag{2.7}$$

$$L_{c_j,v_k} = 2\tanh^{-1}\left(\prod_{l \in \mathcal{N}(c_j)\backslash v_k} \tanh\left(\frac{L_{v_l,c_j}}{2}\right)\right) \quad . \tag{2.8}$$

Here $\mathcal{N}(v_k)\backslash c_j$ represents the set of check nodes connecting to v_k excluding c_j. The exclusion of c_j here precludes the information received by v_k from c_j to be reused to calculate the message L_{v_k,c_j}. Similarly, in (2.8) the variable node v_k is excluded while calculating the output message L_{c_j,v_k}.

The iterative process continues until the symbols are decoded or the maximum number of iterations $I_{\max}$ is reached. The output LLR $L_{\text{out}}(v_k)$ for variable node v_k is then computed as

$$L_{\text{out}}(v_k) = L_{\text{ch}}(v_k) + \sum_{l \in \mathcal{N}(v_k)} L_{c_l, v_k} \ . \tag{2.9}$$

In many practical scenarios it is advantageous to use an iteration stopping rule. A stopping rule, using L_{out}, determines if the received word is correctly decoded and stops the iterative process before the maximum number of iterations $I_{\max}$ is performed. The most commonly used stopping rule for LDPC codes is to calculate the syndrome $\mathbf{s}$ of the estimated codeword $\hat{\mathbf{v}}$ ($\mathbf{s} = \mathbf{H}\hat{\mathbf{v}}^T$) [4]. The iterative process stops when $\mathbf{s} = \mathbf{0}$, which indicates that a valid codeword is estimated. Note that $\mathbf{s} = \mathbf{0}$ only assures that $\hat{\mathbf{v}}$ is a valid codeword and it does not guarantee $\hat{\mathbf{v}} = \mathbf{v}$. Additionally, there exists also an *early stopping rule* [LYL06], which stops the iterative process earlier than $I_{\max}$ when it determines that the decoding will not be successful. In this work we will only consider an iteration stopping rule and a rule is proposed in Chapter 3 for the spatially coupled LDPC codes.

2.2.3 Density Evolution

In order to analyze the asymptotic performance of an iterative decoder, a technique known as *density evolution* is used [5]. Density evolution tracks the probability distribution of messages that are exchanged between nodes in the Tanner graph [RSU01] in every iteration. The worst channel parameter for which the decoding error probability converges to zero is refereed as the *BP threshold* of an ensemble. For the case of the AWGN channel, we denote the BP threshold of an ensemble as σ^*_{BP}. Furthermore, we always consider protograph based LDPC codes and hence apply density evolution equations on the considered protograph. The main advantage of using protograph based LDPC codes is that it allows us to analyze the performance of ensemble of codes instead of a particular LDPC code. The density evolution equations are omitted here but interested readers are referred to [CRU01] [RU08].

We make use of density evolution to calculate the BP thresholds of ensembles in Chapter 3. Also density evolution is used to analyze the decoding process in Chapter 4. In Chapter 5, density evolution is used to calculate the outage probability

[4] $\hat{\mathbf{v}}$ is calculated by hard decision based on the output of (2.9).

[5] here the term asymptotic refers to having infinite codeword length N, i.e., infinite lifting factor Z.

of a coded transmission over a slow time varying channel. The density evolution results are always supported by simulation results where a PEG algorithm is used to generate an LDPC code.

2.3 Spatially Coupled LDPC Codes

In order to push the BP threshold of an LDPC code close to the Shannon limit, irregularity in the Tanner graph plays an important role. However, this is only at the expense of high error floor. In this section we use a technique known as *spatial coupling* to push the BP threshold of a regular LDPC code close to the Shannon limit while guaranteeing the linear growth of the minimum distance.

Spatially coupled LDPC (SC-LDPC) codes were first invented by Jimenez Felström and Zigangirov in [JFZ99]. Since these are characterized by a semi-infinite diagonal type parity-check matrix, these were referred to as LDPC convolutional codes. Analogous to LDPC codes, the parity-check matrix of an SC-LDPC code can also be derived by protograph expansion (see Example 2.1).

In order to describe spatial coupling, let us consider a transmission of a sequence of L codewords $\mathbf{v}_t$, $t = 1, \ldots, L$, using a protograph based LDPC code. An essential feature of SC-LDPC codes is that the blocks at different time instants are interconnected. Instead of encoding all codewords independently, the blocks $\mathbf{v}_t$ are *coupled* by the encoder to other time instants. The largest distance between a pair of coupled blocks defines the memory m_{cc} of the coupled code. The coupling of consecutive blocks can be achieved by distributing the edges from variable nodes at time t among equivalent check nodes at times $t + i$, $i = 0, \ldots, m_{\text{cc}}$, called an *edge spreading* procedure [LFZCJ09]. The resultant regular SC-LDPC ensemble is represented as (J, K, L, m_{cc}).

In order to maintain the degree distribution and structure of the original protograph, a valid edge spreading should satisfy the condition

$$\sum_{i=0}^{m_{\text{cc}}} \mathbf{B}_i = \mathbf{B} \ . \tag{2.10}$$

Example 2.2. Consider a protograph of a $(3, 6)$-regular LDPC code with base matrix $\mathbf{B} = [3\ 3]$ as shown in Fig. 2.4(a) and an edge spreading using the component base matrices as

$$\mathbf{B}_0 = \mathbf{B}_1 = \mathbf{B}_2 = [1\ 1]. \tag{2.11}$$

Here the ensemble has memory $m_{\text{cc}} = 2$. The component matrix $\mathbf{B}_i$ represents the connections between the variable nodes at time instant t and the check nodes at

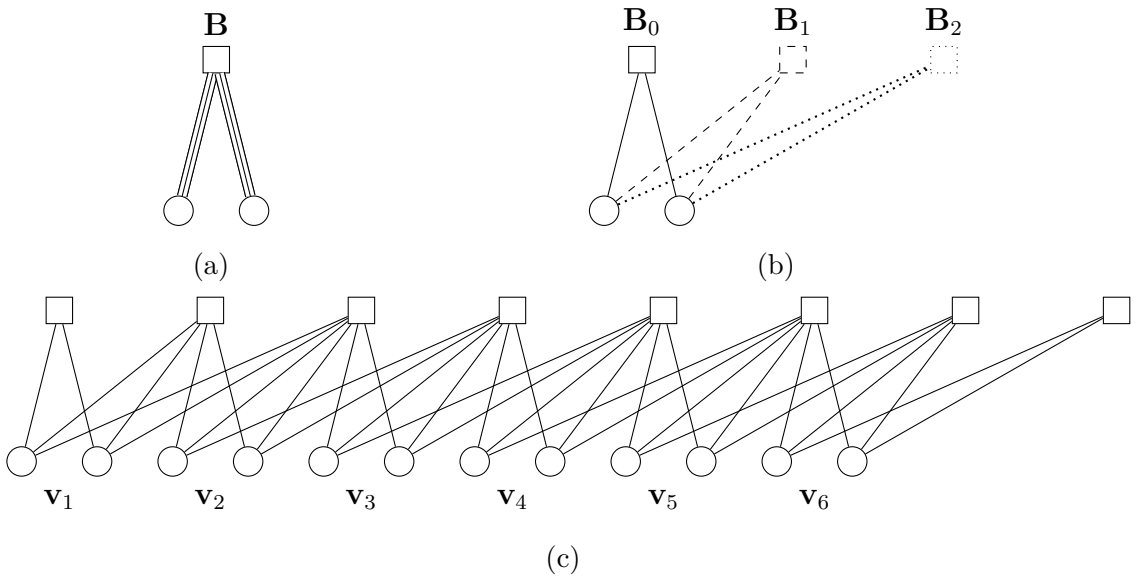


Figure 2.4: An illustration of edge spreading for a protograph based $(3, 6)$-regular LDPC code with base matrix $\mathbf{B}$. The protograph is repeated $L = 6$ times and the edges are spread over time according to the component base matrices $\mathbf{B}_0$, $\mathbf{B}_1$, and $\mathbf{B}_2$, resulting in a $(3, 6, 6, 2)$ SC-LDPC code.

time instant $t+i$, as shown in Fig. 2.4(b). Finally, considering a termination length of $L = 6$, the Tanner graph corresponding to the terminated spatially coupled code is obtained as shown in Fig. 2.4(c).

■

A terminated SC-LDPC code can be described by means of a *convolutional protograph* base matrix

$$\mathbf{B}_{[1,L]} = \begin{bmatrix} \mathbf{B}_0(1) & & \\ \vdots & \ddots & \\ \mathbf{B}_{m_{\text{cc}}}(1) & & \mathbf{B}_0(L) \\ & \ddots & \vdots \\ & & \mathbf{B}_{m_{\text{cc}}}(L) \end{bmatrix}_{(L+m_{\text{cc}})n_c \times Ln_v} . \tag{2.12}$$

The corresponding sequence of coupled code blocks forms a codeword $\mathbf{v}_{[1,L]} = [\mathbf{v}_1, \ldots, \mathbf{v}_t, \ldots, \mathbf{v}_L]$ of a terminated spatially coupled code. Here we always consider time invariant protograph, i.e., the edge spreading does not change over time. Note that $m_{\text{cc}} = 0$, i.e., $\mathbf{B}_0 = \mathbf{B}$ in (2.12) corresponds to L LDPC block codes with disconnected protographs.

Based on the parity-check matrix of a terminated SC-LDPC code in (2.12), the following two observations are made,

- Starting from a (J, K)-regular LDPC code, an asymptotically regular SC-LDPC code is obtained with slight irregularity at both ends of the Tanner graph. This results into low degree check nodes at both ends of the Tanner graph. Since the check node update in (2.8) corresponds to a parity-check code, a low degree check protects the connected code bits more strongly than a high degree check node. This property of the SC-LDPC code is beneficial for the decoding and will be further discussed in Section 2.3.1.

- The rate of a (J, K, L, m_{cc}) regular coupled code with termination length L and memory m_{cc} is given by,

$$R_L = 1 - \left(\frac{L + m_{\text{cc}}}{L}\right)\frac{J}{K} \ . \tag{2.13}$$

  The $m_{\text{cc}}n_c$ additional check nodes result in a rate loss due to termination. However, the rate loss diminishes with increasing termination length L.

2.3.1 Threshold Saturation

In this section, making use of density evolution, we calculate the BP thresholds of protograph based LDPC and SC-LDPC codes. In general, it can be shown that the maximum a posteriori (MAP) threshold of regular LDPC ensembles converges to the Shannon limit when the node degrees are increased. On the other hand, the BP threshold moves away from the Shannon limit with increasing degrees. Table. 2.1 shows the BP thresholds of the LDPC codes when the variable node degrees are increased from 3 to 5[6] for regular LDPC codes with rate $R = 1/2$. For the case of LDPC block codes, the BP threshold degrades with increasing degrees. As mentioned above, in order to close the gap between the BP threshold of regular LDPC ensembles and the Shannon limit, the inclusion of degree-2 variable nodes is indispensable, which induces an error floor.

In contrast to this, spatial coupling improves the performance of the iterative decoder significantly and it has been shown in [LSCJZ10] that the BP threshold of the coupled code asymptotically approaches the MAP threshold of the underlying LDPC code when the termination length L tends to infinity (see Fig. 2.5). For small values of L, increasing the node degree has a negative effect on the BP

[6] the corresponding check node degrees are increased from 6 to 10.

Table 2.1: BP thresholds (σ^*_{BP}) for LDPC and SC-LDPC codes with increasing node degrees.

Code	LDPC	SC-LDPC ($L \to \infty$)
$(3, 6)$	0.875	0.9477
$(4, 8)$	0.829	0.972
$(5, 10)$	0.783	0.9778

threshold, whereas, as L tends to infinity, the BP threshold of regular SC-LDPC codes converge to the Shannon limit. The thresholds of the LDPC codes can also be visualized in terms of E_b/N_0 in Fig. 2.5(b). Here the rate of the code is taken into account and the small value of L results in larger noise variance. However, the behavior is similar to Fig. 2.5(a) when the termination length tends to infinity. The Table 2.1 provides the BP threshold of the coupled LDPC code in the asymptotic case. In contrast to uncoupled case, BP threshold of coupled code moves closer to the Shannon limit with increasing node degrees.

This *threshold saturation* phenomenon was first observed by Lentmaier et.al. in [LSCJZ10]. Later an analytical proof of threshold saturation has been given by Kudekar, Richardson and Urbanke in [KRU10][KRU12]. A similar proof is extended in [KYMP12] for protograph based codes using potential function.

Note that, with increasing variable node degrees from 3 to 5, memory has also been increased from 2 to 4, respectively. This increase in memory is required to increase the effect of coupling and hence increases the strength of the iterative decoder. However, increasing memory of the code has consequences on the decoding latency in multiple ways and shall be discussed in detail in Chapter 3.