國 立 交 通 大 學

電信工程研究所

碩 士 論 文

新的用於低密度同位檢查碼之重組解碼方法及其收斂性分析

New Shuffled Decoding of LDPC Codes and their Convergence Analysis

研究生：許晏誠

指導教授：蘇育德 教授

中 華 民 國 一百零一年 九月

新的用於低密度同位檢查碼之重組解碼方法及其收斂性分析

New Shuffled Decoding of LDPC Codes and their Convergence Analysis

研 究 生：許晏誠　　Student：Yen-Cheng Hsu

指導教授：蘇育德　教授　　Advisor：Prof. Yu T. Su

國 立 交 通 大 學

電信工程研究所

碩 士 論 文

A Thesis

Submitted to the Institute of Communications Engineering

in partial fulfillment of the requirements

for the Degree of

Master of Science

in

Communications Engineering

at the

National Chiao Tung University

September 2012

Hsinchu, Taiwan

公元2012年九月

新的用於低密度同位檢查碼之重組解碼方法及其收斂性分析

學生：許晏誠　　指導教授：蘇育德　教授

國立交通大學

電信工程研究所碩士班

摘　　要

在本論文中，我們提出了兩種新的用於低密度同位檢查碼（LDPC codes）的重組信度傳遞解碼演算法(Shuffled Belief-Propagation Decoding)；為了加速解碼收斂速度以及降低運算複雜度，我們提出了一種將檢查節點分成有交集的組別的重組信度傳遞解碼方法。此外，我們亦提出了將水平重組(針對檢查節點分組)和垂直重組(針對變數節點做分組)混合交互使用的解碼方法。

我們使用高斯近似（Gaussian Approximation）的方法來分析不同信度傳遞演算法的效能與解碼行為；理論分析與實驗模擬結果皆一致地顯示出我們所提之方法在相同的複雜度下可以達到較好的解碼成果。經由蒙特卡羅（Monte-Carlo）模擬的結果可以發現我們所提出的兩種演算法比起傳統的重組信度傳遞解碼演算法皆有較好的錯誤率效能。

New Shuffled Decoding of LDPC Codes and their Convergence Analysis

Student : Yen-Cheng Hsu Advisor : Prof. Yu Ted Su

Institute of Communication Engineering

National Chiao Tung University

Abstract

Two new shuffled belief propagation decoding algorithms for low-density parity-check (LDPC) codes are proposed in this thesis. To accelerate the decoding convergence rate and lower the implementation complexity, we propose a group shuffled decoding schedule which divides check nodes into non-disjoint groups to perform group-by-group message-passing decoding. A hybrid shuffled schedule which combines horizontal (partitioning check nodes into groups) and vertical (partitioning variable nodes into groups) shuffled schemes is also presented.

Performance of the proposed algorithms are analyzed by a Gaussian approximation based approach. Both analysis and numerical experiments verify that the new algorithms do yield a convergence performance better than that of existing conventional shuffled BP decoder with the same computing complexity constraint. In terms of error-rate performance, Monte-Carlo simulations show that the proposed approaches yield improved results in comparison with the conventional shuffled decoding schedules.

誌謝

能完成我的碩士班研究，首先要感謝的是這兩年來辛苦提攜與指導我們的蘇育德老師，因為有老師的細心教導與提供完善的實驗環境，才能夠完成今天的這篇論文。在學術的這座殿堂當中，雖然還好深、好廣。但是感謝老師提供了如此多的機會，才讓我們能夠更加的貼近了一些些、更了解它一點點。老師不僅僅在研究上給予我們許多建議和幫助，更在為人處事方面給了很多受用的指導，讓我受益良多。

感謝這兩年來曾經一起在實驗室追求自己學位的學長姐及學弟妹們，感謝你們和我一起經歷彼此生命中這一段珍貴且不會重來的時光。其中特別要感謝的是張致遠學長，從大學時期做專題起就一直盡心盡力地帶著我和殷呈做研究，不管研究上遇到甚麼問題，你總是會跳出來幫我們一起想辦法解決，真的很謝謝你對我們的付出。另外還要謝謝翁健家學長，雖然不是我們實驗室的，但是對於我們的研究也是不餘遺力的幫忙和討論。

再來要感謝的是系上所有的好朋友們，還有一起住過的室友們，有你們的陪伴才能讓我的休閒生活多采多姿。最後當然要感謝我的父母，能讓我在求學時期完全不用考慮經濟問題，並不斷的給予我支持和鼓勵，讓我能夠完成我的學業。

Contents

List of Figures

Chapter 1

Introduction

Low-density parity-check (LDPC) codes with belief propagation (BP) or so-called sum-product algorithm (SPA) based decoder can offer near-capacity performance. The SPA decoder, however, suffers from low convergence rate and high implementation complexity. To improve the rate of convergence and reduce implementation cost, serialized BP decoding algorithms which partition either the variable nodes (VNs) [5] or the check nodes (CNs) [6] of the corresponding bipartite graph into multiple groups were introduced. These two classes of serial SPA algorithms are called vertical and horizontal shuffled BP (SBP) decoding algorithms, respectively. More recent related works can be found in [7] -[10]. These practical alternatives use serial-parallel decoding schedules that perform sequential group-wise message-passings and have the advantage of obtaining more reliable extrinsic messages for subsequent decoding within an iteration.

We focus on the horizontal shuffled BP (HSBP) decoding algorithms as they provide more advantages in hardware implementation [5] [10] than vertical shuffled BP (VSBP). For conventional HSBP schedules, the CNs are divided into a number of groups such that each CN belongs to just one group. A decoding iteration consists of several sub-iterations. Each sub-iteration updates in parallel the log-likelihood ratios (LLR) associated with the VNs connecting to the CNs in the same group. Hence within a sub-iteration, message-passing is performed on the bipartite subgraph that consists of the CNs of a group and all the VNs connecting to these CNs. Unlike conventional shuffled

schedules which partition either VNs or CNs into disjoint groups, we propose a shuffled decoding schedule which divides CNs into non-disjoint CN groups. Such a CN grouping results in larger connectivity of consecutive subgraphs (CoCSG) associated with two neighboring CN groups, where the CoCSG refers to the the average number of VNs connecting the CNs of, say, the kth group and the VNs which are also linked to the CNs of the previous, i.e., $(k-1)$th, CN group. A larger CoCSG means more information will be forwarded from the previous sub-iteration and thus provides opportunities for improved decoding performance. We demonstrate by using both simulation and analysis that the proposed SBP is indeed capable of offering performance improved and additional performance-complexity-decoding delay tradeoffs. Since our division on the CNs yields CN groups with a nonempty intersection for any two neighboring groups, we refer to the resulting decoding schedule as non-disjoint group-shuffled belief propagation (NDGSBP) in subsequent discourse.

Shuffled BP decoding is a sequential approach, and the conventional method is based on a natural increasing order according to the node indexes. In vertical shuffled BP decoding, the later a bit is processed, the more information it may get. Therefore, as the index increases, the reliability of the bit increases and the corresponding error rate decreases. This may result in the unequal error-correcting capability of the coded bits and yield bad convergence performance. Randomly adjusting the updating order of VNs or CNs in each iteration is a simple and effective way to overcome this drawback, i.e., updating messages in a random node-by-node order helps that each bit could obtain equivalent amount of new updated messages. However, this random-ordering manner is impractical due to the high hardware implementation complexity. We propose a new decoding schedule which alternately performs VSBP and HSBP decoding to achieve pseudo-random decoding schedule and name it hybrid-shuffled belief propagation (H-SBP) decoding. The H-SBP algorithm provides excellent trade-offs between error-rate performance and implementation complexity.

To analyze the performance of iterative LDPC decoding algorithms in binary-input additive white Gaussian noise (BI-AWGN) channels, approaches such as density evolution (DE), Gaussian approximation (GA), and extrinsic information transfer (EXIT) charts have been proposed [11]-[15]. We adopt the GA approach [12] [15] as it requires just the tracking of the first two moments which are sufficient to completely characterize the probability densities. Moreover, if a consistency condition is met [15], we need to track only the means of related likelihood parameters.

The rest of this thesis is organized as follows. In chapter 2, we review the basic definition, some decoding algorithms of LDPC code and the GA approach. We explain the basic idea of the new grouping methods, provide relevant parameter definitions and present the NDGSBP decoding algorithm and hybrid SBP in Chapter 3. The corresponding GA-based performance analysis is given in Chapter 4. Chapter 5 provides numerical performance examples of the our algorithms, estimated by both computer simulations and analysis. Finally, concluding remarks are drawn in Chapter 6.

Chapter 2

Review of Low-Density Parity-Check Codes

Low-density parity-check codes form a class of linear block codes which provide the near-Shannon-limit performance with practical complexity if the code length is long enough. It was originally invented by Gallager [1]. The algorithm Gallager proposed was too complex to implement at that time thus it was ignored by researchers for almost 35 years. In the meanwhile Tanner provided a graphical interpretation of LDPC codes, which are called as Tanner graphs [2]. LDPC codes were "rediscovered" again until the mid 1990s with the works of MacKay and Neal [3]. They noticed the advantage between linear block codes which generated by sparse matrix and iterative decoding based on belief propagation. And by that time the decoding complexity has become practically achievable and extensive efforts on various related issues then followed.

For simplicity, we only consider binary LDPC codes. This chapter starts with the fundamental representations of LDPC codes via parity-check matrix and Tanner graphs. We then introduce the sum-product (or belief propagation, BP) algorithm and shuffled iterative decoding algorithms for the binary-input additive white Gaussian noise (BI-AWGN) channels. Finally, we investigate the iterative decoding performance of LDPC code ensembles using the Gaussian approximation (GA) approach [12].

2.1 Representations of LDPC codes

2.1.1 Matrix representation

As its name implies, an LDPC code is a linear block code defined by the null space of a parity-check matrix **H** that has a low density of 1s. An LDPC code with a parity-check matrix **H** which has constant row and column weights d_c and d_v is called a (d_c, d_v) regular LDPC code. It said to be irregular if all the rows or all the columns of the parity-check matrix **H** do not have the same weight.

2.1.2 Tanner graphs

A Tanner graph is a bipartite graph used to illustrate constraints or parity check equations which characterizes an error correcting code. The graph is partitioned into check nodes (CNs) and variable nodes (VNs) which denote the rows of the parity-check matrix **H** and the columns of the parity-check matrix **H**, respectively. An edge connects the CN i to the VN j whenever the element h_{ij} in parity-check matrix **H** is a 1. The Tanner graph of a LDPC code is a graphical model as the trellis of a convolutional code. It not only provides another representation of the code but helps to describe and develop decoding algorithms. Each of nodes is like a locally operating processor and each edge is likes a bridge that conveys the messages from a given node to its neighbors.

Example 1 *Consider a* $N = 6$ *linear block code with* $d_c = 3$ *and* $d_v = 2$ *with the following* **H** *matrix, the Tanner graph corresponding to* **H** *is depicted in Figure 2.1:*

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

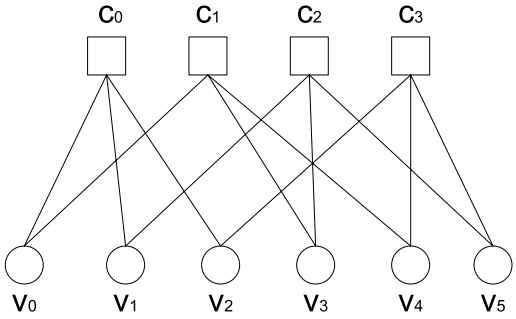


Figure 2.1: The Tanner graph for the code given in the example.

2.2 The sum-product algorithm

Gallager also proposed a near-optimal iterative decoding algorithm which is now called the sum-product algorithm (SPA) besides introducing LDPC codes in his doctoral dissertation. It also known as belief propagation algorithm which is used in describing inference in Bayesian networks and was originally invented by Pearl for developing probabilistic approaches for artificial intelligence applications. The SPA can be viewed as two kind of decoders work cooperatively, one is a repetition (REP) decoder (VN decoder) and the other is a single parity check (SPC) decoder (CN decoder). Figs. 2.2 and 2.3 depict the VN and CN decoder situations. For simplicity, we show the updating equations of these two decoders directly with the extrinsic information to be sent from VN n to CN m

$$L_{n\to m} = L_n + \sum_{m'\in\mathcal{M}(n)-\{n\}} L_{m'\to n}. \tag{2.1}$$

The extrinsic information to be sent from CN m to VN n is

$$L_{m\to n} = 2\tanh^{-1}\left(\prod_{n'\in\mathcal{N}(m)-\{m\}} \tanh\left(\frac{1}{2}L_{j'\to i}\right)\right). \tag{2.2}$$

A binary (N, K) LDPC code $\mathcal{C}$ is a linear block code whose $M \times N$ parity check matrix $\mathbf{H} = [H_{mn}]$ has sparse nonzero elements. And thus $\mathcal{C}$ can be viewed as a bipartite graph with N VNs corresponding to the encoded bits and M CNs corresponding to the

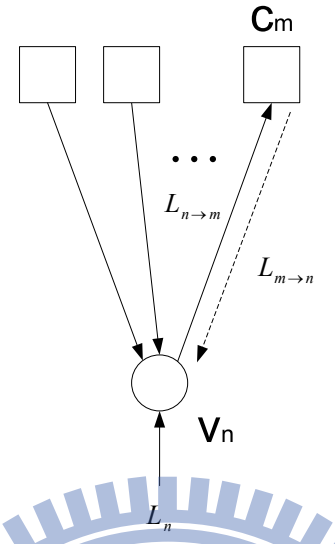


Figure 2.2: A VN decoder (REP decoder). VN n receives LLR information from the channel and from all of its neighbors, excluding message $L_{m\to n}$ from CN m, from which VN n composes message $L_{n\to m}$ that is sent to CN m.

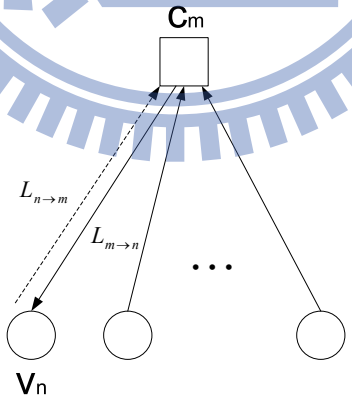


Figure 2.3: A CN decoder (SPC decoder). CN m receives LLR information from all of its neighbors, excluding message $L_{n\to m}$ from VN n, from which CN m composes message $L_{m\to n}$ that is sent to VN n.

parity-check functions represented by the rows of $\mathbf{H}$.

Let $\mathcal{N}(m)$ be the set of variable nodes that participate in check node m and $\mathcal{M}(n)$ be the set of check nodes that are connected to variable node n in the code graph. $\mathcal{N}(m)\backslash n$ is defined as the set $\mathcal{N}(m)$ with the variable node n excluded while $\mathcal{M}(n)\backslash m$ is the set $\mathcal{M}(n)$ with the check node m excluded. Let $L_{n\to m}$ be the message sent from VN n to CN m and $L_{m\to n}$ be the message sent from CN m to VN n.

Assume a codeword $\mathbf{c} = (c_0, c_1, ..., c_{N-1})$ is BPSK-modulated and transmitted over an AWGN channel with noise variance σ^2. Let $\mathbf{y} = (y_0, y_1, ..., y_{N-1})$ be the corresponding received sequence and L_n be the log-likelihood ratio (LLR) of the variable node n. Let l be the iteration counter and I_{Max} be the maximum number of iterations. The SPA is given as follows:

Initialization

Set $l = 1$, $L_n = \frac{2}{\sigma^2} y_n$, for $0 \le n \le N-1$.

Step 1: Message passing

a) CN update: $\forall m, 0 \le m \le M-1$, and $n \in \mathcal{N}(m)$

$$L_{m\to n} = 2\tanh^{-1}\left(\prod_{n'\in\mathcal{N}(m)\backslash n} \tanh\left(\frac{1}{2}L_{n'\to m}\right)\right) \tag{2.3}$$

b) VN update: $\forall n, 0 \le n \le N-1$, and $m \in \mathcal{M}(n)$

$$L_{n\to m} = L_n + \sum_{m'\in\mathcal{M}(n)\backslash m} L_{m'\to n} \tag{2.4}$$

Step 2: Total LLR computation

$\forall n, 0 \le n \le N-1$,

$$L_n^{total,(l)} = L_n + \sum_{m'\in N(n)} L_{m'\to n} \tag{2.5}$$

Step 3: Hard decision and stopping criterion test

a) Create $\mathbf{D}^{(l)} = [d_0^{(l)}, d_1^{(l)}, ..., d_{N-1}^{(l)}]$ such that $d_n^{(l)} = 0$ if $L_n^{total,(l)} \geq 0$ and $d_n^{(l)} = 1$ if $L_n^{total,(l)} < 0$.

b) If $\mathbf{D}^{(l)}\mathbf{H}^T = \mathbf{0}$ or I_{Max} is reached, stop decoding and output $\mathbf{D}^{(l)}$ as the decoded codeword. Otherwise, set $l = l + 1$ and go to **Step 1**.

2.3 Shuffled belief propagation decoding

It is well known that long LDPC codes decoded by SPA or a BP-based algorithm can yield capacity-approaching performance. However, for long LDPC codes, implementing the fully parallel algorithm require a large number of processing units, high computation complexity, large memory space, and complicated network connecting. A more practical alternative is to partition either the variable nodes or the check nodes of the corresponding bipartite code graph into several groups and perform group-wise parallel decoding in a serial manner. That is, the parallel-serial architecture divides a single iteration into several sub-iterations so that one needs only a reasonable hardware complexity to perform each sub-iteration. There are two different types of shuffled schedules called vertical shuffled BP (VSBP) and horizontal shuffled BP (HSBP), depending on whether variable nodes or check nodes are partitioned. Once a group is processed, the subsequent groups will have a chance to obtain the corresponding updated messages. As a result, a shuffled BP algorithm often converge faster than its standard BP counterpart does.

2.3.1 Vertical shuffled belief propagation algorithm

The vertical shuffled BP algorithm divides VNs into several groups and each group updates sequentially in a iteration. Different grouping methods result in different convergence. How to group the VNs so that the messages spread faster is another topic, we assume that VNs are partitioned into groups based on their index order. Let G be the number of VN groups, $\mathcal{G}_g$ be the gth VN group, l be the iteration counter and I_{Max}

be the maximum number of iterations. We can then describe the VSBP algorithm as follows:

Initialization

Set $l = 1$, $L_n = \frac{2}{\sigma^2} y_n$, for $0 \leq n \leq N-1$.

Step 1: Message passing

For $0 \leq g \leq G-1$

a) CN update: $\forall\ m \in \mathcal{M}(n), n \in \mathcal{G}_g$

$$L_{m \to n} = 2 \tanh^{-1} \left(\prod_{n' \in \mathcal{N}(m) \backslash n} \tanh \left(\frac{1}{2} L_{n' \to m} \right) \right) \tag{2.6}$$

b) VN update: $\forall\ n \in \mathcal{G}_g, m \in \mathcal{M}(n)$

$$L_{n \to m} = L_n + \sum_{m' \in \mathcal{M}(n) \backslash m} L_{m' \to n} \tag{2.7}$$

Step 2: Total LLR computation

$\forall n, 0 \leq n \leq N-1$,

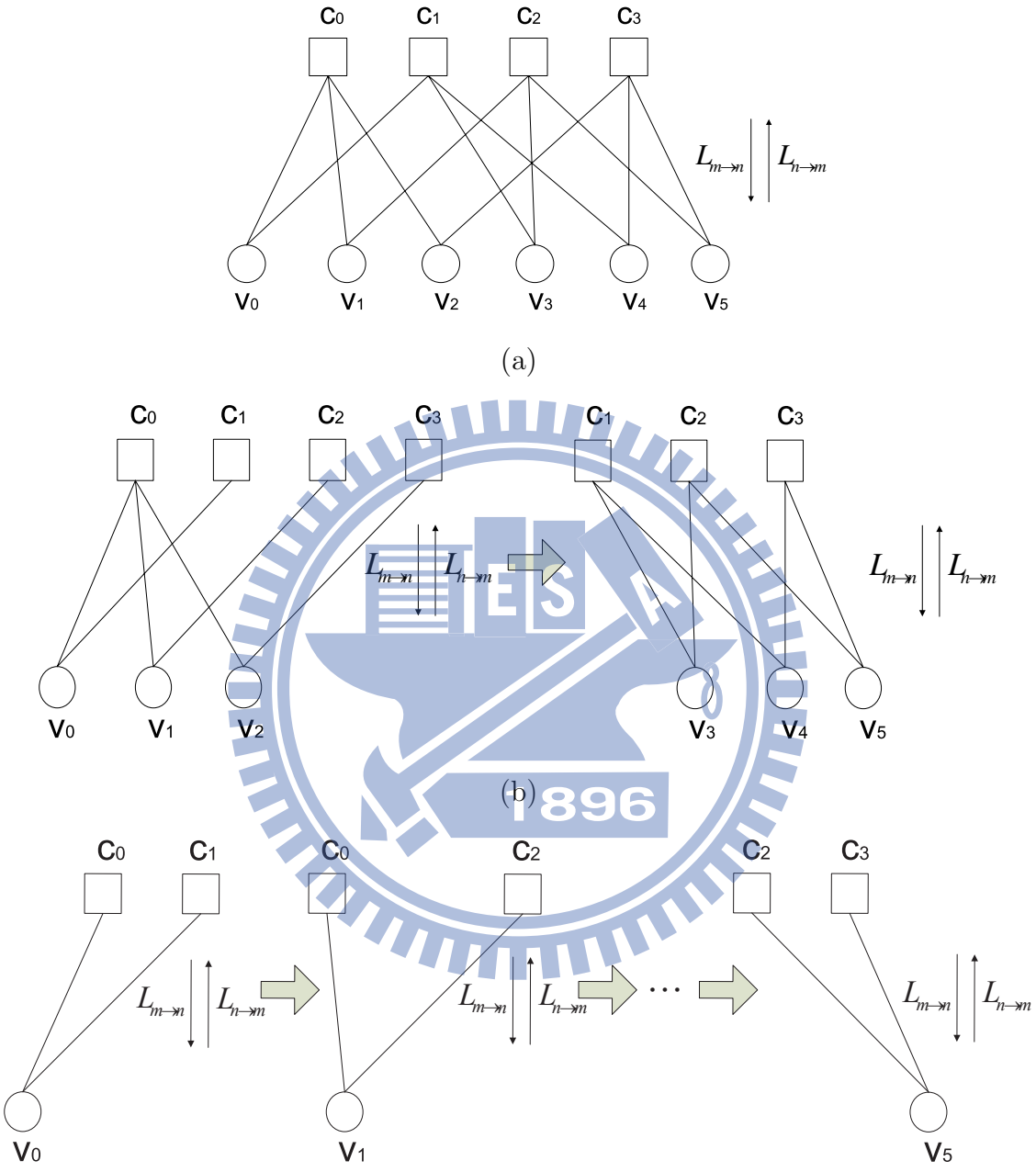
$$L_n^{total,(l)} = L_n + \sum_{m' \in \mathcal{N}(n)} L_{m' \to n} \tag{2.8}$$

Step 3: Hard decision and stopping criterion test

a) Create $\mathbf{D}^{(l)} = [d_0^{(l)}, d_1^{(l)}, ..., d_{N-1}^{(l)}]$ such that $d_n^{(l)} = 0$ if $L_n^{total,(l)} \geq 0$ and $d_n^{(l)} = 1$ if $L_n^{total,(l)} < 0$.

b) If $\mathbf{D}^{(l)}\mathbf{H}^T = \mathbf{0}$ or I_{Max} is reached, stop decoding and output $\mathbf{D}^{(l)}$ as the decoded codeword. Otherwise, set $l = l + 1$ and go to **Step 1**.

As an example, let use consider the code with parity-check matrix $\mathbf{H}$ of Example 1. The decoding processes for one iteration of the vertical shuffled BP is illustrated in Fig. 2.4 with $G = 1$ (SPA), 2, and 6.



(a)

(b)

(c)

Figure 2.4: Vertical shuffled BP with $G = 1, 2, 6$ for decoding a code given in Example 1. (a) $G = 1$ (standard BP). (b) $G = 2$. (c) $G = 6$.

2.3.2 Horizontal shuffled belief propagation algorithm

Similar to VSBP algorithm, HSBP algorithm separate all check nodes into several groups and decoding is carried out in a group-by-group manner. Relative to VSBP algorithm, HSBP provides more advantages in hardware implementation. As VSBP, different grouping methods could result in the difference of the decoding performance and here we assume that the index-order-based partition is used. Let G be the number of VN groups, $\mathcal{G}_g$ be the gth VN group, l be the iteration counter and I_{Max} be the maximum number of iterations. We can then describe the HSBP algorithm as follows:

Initialization

Set $l = 1$, $L_n = \frac{2}{\sigma^2} y_n$, for $0 \leq n \leq N-1$.

Step 1: Message passing

For $0 \leq g \leq G-1$

a) CN update: $\forall\, m \in \mathcal{G}_g, n \in \mathcal{N}(m)$

$$L_{m \to n} = 2 \tanh^{-1} \left(\prod_{n' \in \mathcal{N}(m) \backslash n} \tanh \left(\frac{1}{2} L_{n' \to m} \right) \right) \tag{2.9}$$

b) VN update: $\forall\, n \in \bigcup_{m' \in \mathcal{G}_g} \mathcal{N}(m'), m \in \mathcal{M}(n)$

$$L_{n \to m} = L_n + \sum_{m' \in \mathcal{M}(n) \backslash m} L_{m' \to n} \tag{2.10}$$

Step 2: Total LLR computation

$\forall n, 0 \leq n \leq N-1$,

$$L_n^{total,(l)} = L_n + \sum_{m' \in N(n)} L_{m' \to n} \tag{2.11}$$

Step 3: Hard decision and stopping criterion test

a) Create $\mathbf{D}^{(l)} = [d_0^{(l)}, d_1^{(l)}, ..., d_{N-1}^{(l)}]$ such that $d_n^{(l)} = 0$ if $L_n^{total,(l)} \geq 0$ and $d_n^{(l)} = 1$ if $L_n^{total,(l)} < 0$.

b) If $\mathbf{D}^{(l)}\mathbf{H}^T = \mathbf{0}$ or I_{Max} is reached, stop decoding and output $\mathbf{D}^{(l)}$ as the decoded codeword. Otherwise, set $l = l + 1$ and go to **Step 1**.

As an example, consider the code with parity-check matrix $\mathbf{H}$ in Example 1. The decoding processes for one iteration of the horizontal shuffled BP is illustrated in Fig. 2.5 with $G = 1$ (SPA), 2, and 4.

2.4 The Gaussian approximation

To analyze the iterative decoding performance of LDPC code ensembles in the water-fall region, several methods have been proposed. These include density evolution (DE), Gaussian approximation (GA), and extrinsic information transfer (EXIT) charts [11]-[15]. The DE method models the decoding process as messages being passed around as random variables and tracing the associated message probability density functions (pdfs). GA is an approach aimed to simplifies and stabilizes the numerical computations for the BI-AWGN channel by approximate density evolution based on a Gaussian approximation. The pdfs of the messages is approximated to Gaussian densities that fully specified by two parameters, the mean and variance. It allows tracking the message means only, under a consistency assumption. EXIT chart technique is a graphical tool based on the mutual information. The method relies on the Gaussian approximation, but provides some intuition regarding the dynamics and convergence properties. We adopt the GA approach as it requires just the tracking of the first moment which are sufficient to completely characterize the probability densities. In this section, we introduce the GA algorithm for LDPC codes. A message m satisfies the consistency condition

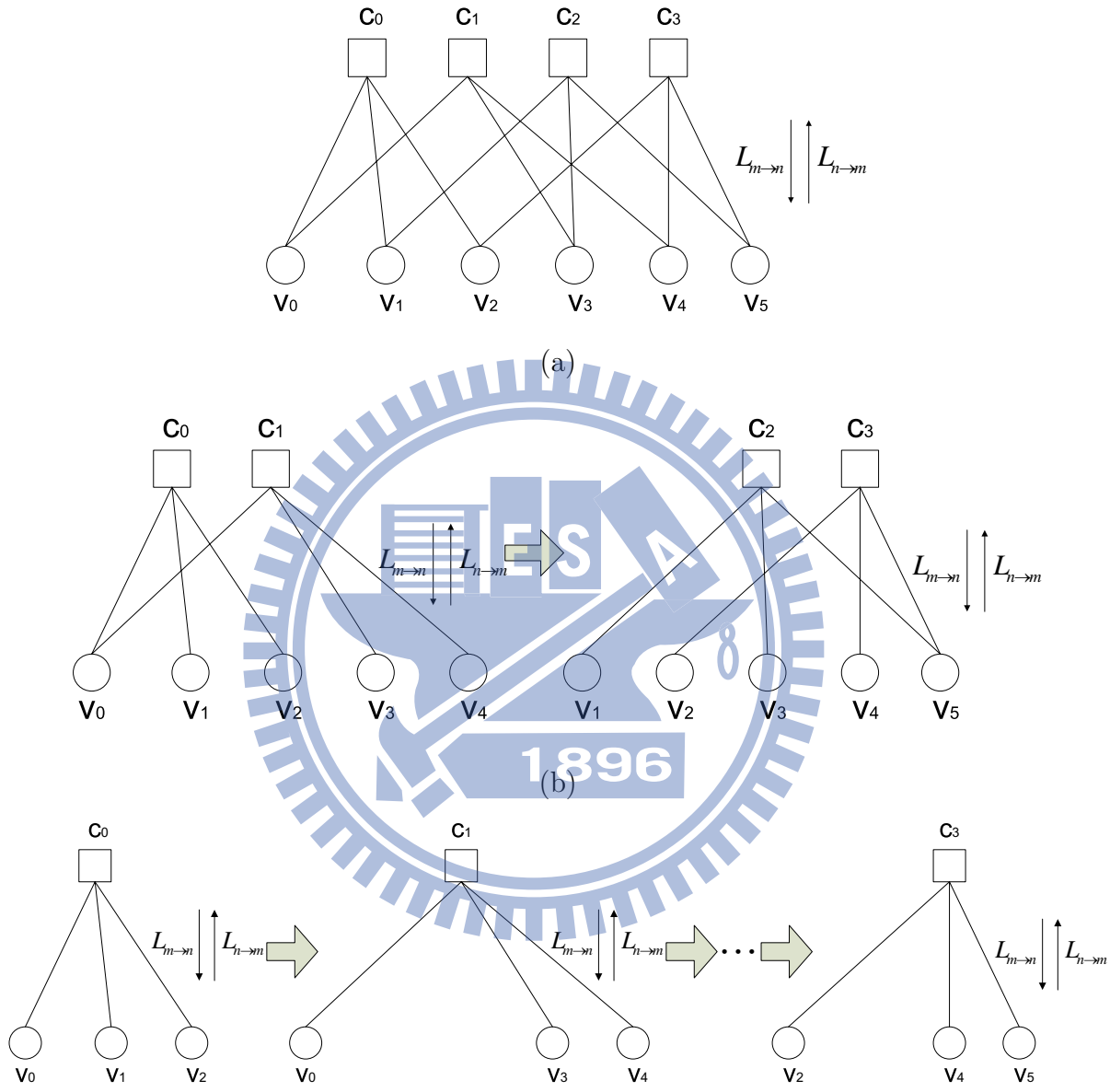


Figure 2.5: Horizontal shuffled BP with $G = 1, 2, 4$ for decoding a code given in Example 1. (a) $G = 1$ (standard BP). (b) $G = 2$. (c) $G = 4$.

if its pdf p_m satisfies

$$p_m(\tau) = p_m(-\tau)e^{\tau}. \tag{2.12}$$

For all Gaussian pdfs that satisfy the consistency condition

$$\frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{1}{2\sigma^2}(\tau-\mu)^2\right] = \frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{1}{2\sigma^2}(-\tau-\mu)^2\right]e^{\tau}, \tag{2.13}$$

it reduces to

$$\sigma^2 = 2\mu. \tag{2.14}$$

One need only monitor the message means when performing density evolution using a Gaussian approximation with the consistency condition.

2.4.1 GA for regular LDPC codes

We start by recalling that an outgoing message form CN c at l iteration may be rewritten as

$$\tanh\left(\frac{m_c^{(l)}}{2}\right) = \prod_{n=1}^{d_c-1}\tanh\left(\frac{m_{v_n}^{(l-1)}}{2}\right), \tag{2.15}$$

where $m_{v_1}^{(l-1)}, \ldots, m_{v_{d_c-1}}^{(l-1)}$ are the messages received from the d_c-1 neighboring VNs. Examining now the propagation of means for regular code ensembles, we take the expected value of this equation under an i.i.d. assumption for the messages $m_{v_n}^{(l-1)}$, we have

$$E\left\{\tanh\left(\frac{m_c^{(l)}}{2}\right)\right\} = \left[E\left\{\tanh\left(\frac{m_v^{(l-1)}}{2}\right)\right\}\right]^{d_c-1}. \tag{2.16}$$

We can therefore write the above quuation as

$$1-\Phi(\mu_{c^{(l)}}) = \left[1-\Phi(\mu_{v^{(l-1)}})\right]^{d_c-1}, \tag{2.17}$$

where, for $\mu > 0$, we define

$$\Phi(\mu) \triangleq 1 - \frac{1}{\sqrt{4\pi\mu}}\int_{-\infty}^{\infty}\tanh(\tau/2)\exp\left[-(\tau-\mu)^2/4\mu\right]d\tau. \tag{2.18}$$

It can be shown that $\Phi(\mu)$ is continuous and decreasing for $\mu \geq 0$, so

$$\mu_{c^{(l)}} = \Phi^{-1}\left(1-\left[1-\Phi\left(\mu_{v^{(l-1)}}\right)\right]^{d_c-1}\right). \tag{2.19}$$

Recalling that an outgoing message form VN v at l iteration may be written as

$$m_v^{(l)} = m_0 + \sum_{n=1}^{d_v-1} m_{c_n}^{(l)}, \tag{2.20}$$

where m_0 is the message from the channel and $m_{c1}^{(l)}, \ldots, m_{c_{d_v-1}}^{(l)}$ are the messages received from the $d_v - 1$ neighboring CNs. Taking the expected value of (2.20), the update equation for the VN-to-CN messages $m_v^{(l)}$, to obtain

$$\begin{aligned} \mu_{v^{(l)}} &= \mu_0 + \sum_{n=1}^{d_v-1} \mu_{c_n^{(l)}} \\ &= \mu_0 + (d_v - 1)\mu_{c^{(l)}}, \end{aligned} \tag{2.21}$$

where the second line follows from the fact that the $d_v - 1$ messages are assumed to be i.i.d.

2.4.2 GA for irregular LDPC codes

Analogously to the regular case, the mean of an output message $m_v^{(l)}$ of a degree-i VN is given by

$$\mu_{v_i^{(l)}} = \mu_0 + (i-1)\mu_{c^{(l)}} \tag{2.22}$$

Because a randomly chosen edge is connected to a degree-i variable node with probability λ_i, averaging over all degrees i yields the following Gaussian mixture pdf for the variable node messages:

$$p_{v_i^{(l)}}(\tau) = \sum_{i=2}^{d_{vMax}} \lambda_i \cdot \mathcal{N}_\tau \left(\mu_{v_i^{(l)}}, 2\mu_{v_i^{(l)}} \right), \tag{2.23}$$

where $\mathcal{N}_\tau(\mu, \sigma^2)$ represents the pdf for a Gaussian r.v. with mean μ and variance σ^2. The expected value within brackets $[\cdot]$ on the right-hand side of (2.16) is

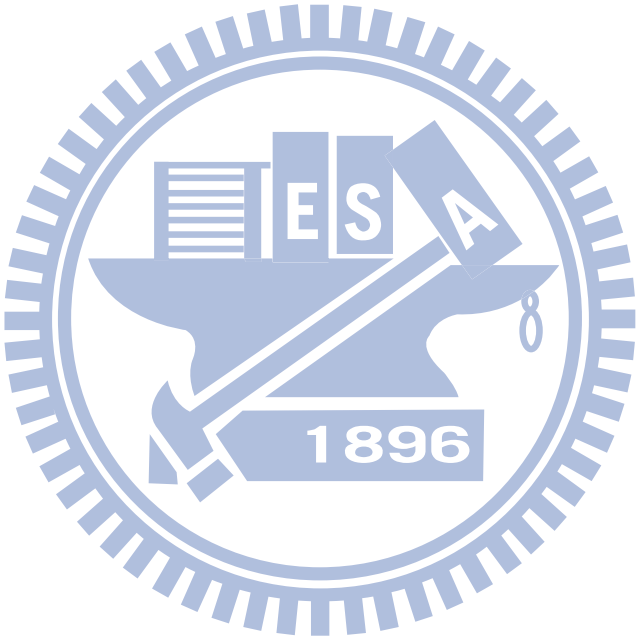
$$\begin{aligned} E\left\{ \tanh\left(\frac{m_v^{(l)}}{2} \right) \right\} &= \int_{-\infty}^{\infty} \tanh\left(\frac{m_v^{(l-1)}}{2} \right) \sum_{i=2}^{d_{vMax}} \lambda_i \cdot \mathcal{N}_\tau \left(\mu_{v_i^{(l-1)}}, 2\mu_{v_i^{(l-1)}} \right) d\tau \\ &= 1 - \sum_{i=2}^{d_{vMax}} \lambda_i \Phi\left(\mu_{v_i^{(l-1)}} \right). \end{aligned}$$

Reffering to the development of (2.18), we have that the mean of a degree-j check node output is

$$\mu_{c_j^{(l)}} = \Phi^{-1}\left(1 - \left[1 - \sum_{i=2}^{d_{vMax}} \lambda_i \Phi\left(\mu_{v_i^{(l)}}\right)\right]^{j-1}\right). \tag{2.24}$$

Finally, if we average over all check-node degrees j, we have

$$\begin{aligned}\mu_{c^{(l)}} &= \sum_{j=2}^{d_{cMax}} \rho_j \mu_{c_j^{(l)}} \\ &= \sum_{j=2}^{d_{cMax}} \rho_j \Phi^{-1}\left(1 - \left[1 - \sum_{i=2}^{d_{vMax}} \lambda_i \Phi\left(\mu_0 + (i-1)\mu_{c^{(l-1)}}\right)\right]^{j-1}\right).\end{aligned} \tag{2.25}$$



Chapter 3

Proposed Algorithms

In this chapter, we present two new shuffled decoding algorithms named Hybrid-shuffled BP and non-disjoint group shuffled BP, respectively. The ideas and advantages of these proposed algorithms will be shown in the following sections.

3.1 Hybrid-shuffled belief propagation algorithm

Shuffled BP decoding is a bit-based sequential algorithm, the later a bit is processed, the more information it may get. This may result in the unequal error-correcting capability of the coded bits and yield bad convergence performance. Fig. 3.1 depicts the number of bit errors using standard BP, vertical shuffled BP with increasing, decreasing and random order for the Mackay's (504,252) regular LDPC code with $d_c = 6$, $d_v = 3$ at the SNR of 3.0dB. Randomly adjusting the updating order of VNs or CNs in each iteration is a simple and effective way to overcome this drawback, i.e., updating messages in a random node-by-node order helps that each bit could obtain equivalent amount of new updated messages. However, this random-ordering manner is impractical due to the high hardware implementation complexity. We propose a new decoding schedule which alternately performs VSBP and HSBP decoding to achieve pseudo-random decoding schedule and name it hybrid-shuffled BP (H-SBP) decoding.

Let G^v, G^c be the number of VN groups and the number of CN groups respectively. Define $\mathcal{G}_g^v$ as the gth VN group and $\mathcal{G}_g^c$ as the gth CN group. Let l be the iteration

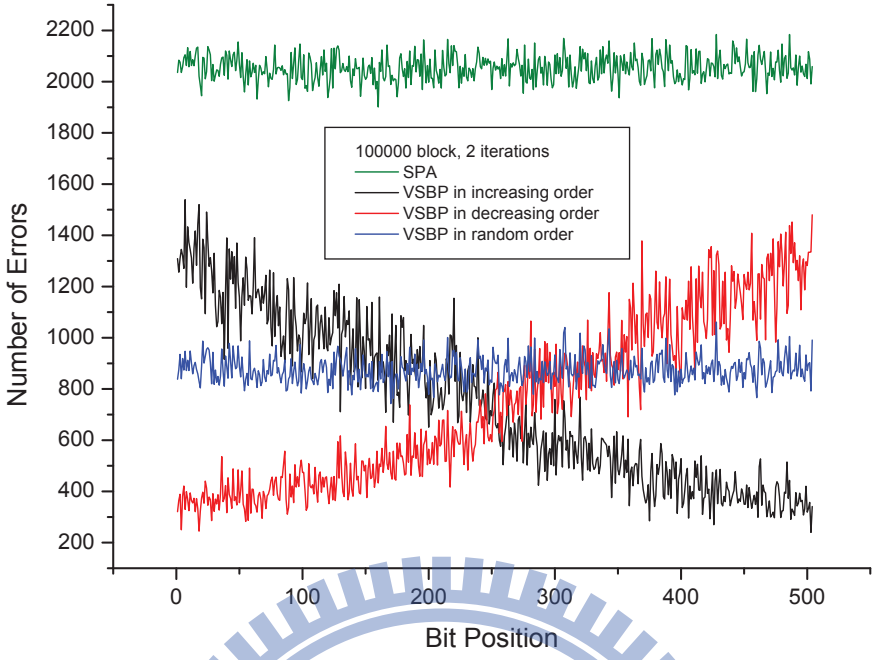


Figure 3.1: Number of bit errors versus bit position in the (504, 252) LDPC code at SNR of 3.0 dB.

counter and I_{Max} be the maximum number of iterations. Then the H-SBP algorithm can be described as follows:

Initialization

Set $l = 1$, $L_n = \frac{2}{\sigma^2} y_n$, for $0 \leq n \leq N-1$.

Step 1: If L is even then go to **Step 2**; otherwise, go to **Step 3**.

Step 2: VSBP

For $0 \leq g \leq G^v - 1$

a) CN update: $\forall\ m \in \mathcal{M}(n), n \in \mathcal{G}_g^v$

$$L_{m \to n} = 2 \tanh^{-1} \left(\prod_{n' \in \mathcal{N}(m) \backslash n} \tanh \left(\frac{1}{2} L_{n' \to m} \right) \right) \tag{3.1}$$

b) VN update: $\forall\ n \in \mathcal{G}_g^v, m \in \mathcal{M}(n)$

$$L_{n \to m} = L_n + \sum_{m' \in \mathcal{M}(n) \backslash m} L_{m' \to n} \tag{3.2}$$

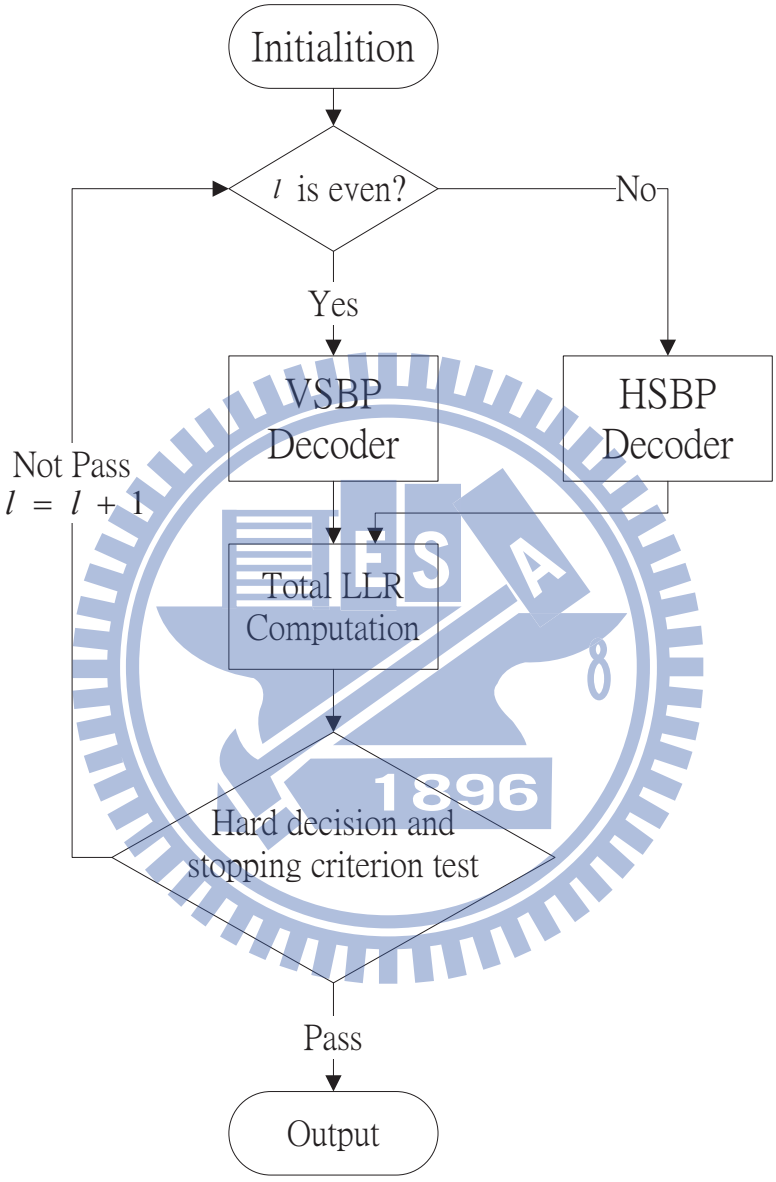


Figure 3.2: Flow chart of H-SBP algorithm.

Step 3: HSBP

For $0 \leq g \leq G^c - 1$

a) CN update: $\forall\ m \in \mathcal{G}_g^c, n \in \mathcal{N}(m)$

$$L_{m \to n} = 2 \tanh^{-1} \left(\prod_{n' \in \mathcal{N}(m) \backslash n} \tanh \left(\frac{1}{2} L_{n' \to m} \right) \right) \tag{3.3}$$

b) VN update: $\forall\ n \in \bigcup_{m' \in \mathcal{G}_g^c} \mathcal{N}(m'), m \in \mathcal{M}(n)$

$$L_{n \to m} = L_n + \sum_{m' \in \mathcal{M}(n) \backslash m} L_{m' \to n} \tag{3.4}$$

Step 4: Total LLR computation

$\forall n, 0 \leq n \leq N - 1,$

$$L_n^{total,(l)} = L_n + \sum_{m' \in \mathcal{N}(n)} L_{m' \to n} \tag{3.5}$$

Step 5: Hard decision and stopping criterion test

a) Create $\mathbf{D}^{(l)} = [d_0^{(l)}, d_1^{(l)}, ..., d_{N-1}^{(l)}]$ such that $d_n^{(l)} = 0$ if $L_n^{total,(l)} \geq 0$ and $d_n^{(l)} = 1$ if $L_n^{total,(l)} < 0$.

b) If $\mathbf{D}^{(l)}\mathbf{H}^T = \mathbf{0}$ or I_{Max} is reached, stop decoding and output $\mathbf{D}^{(l)}$ as the decoded codeword. Otherwise, set $l = l + 1$ and go to **Step 1**.

Fig. 3.2 depict the flow chart of H-SBP algorithm.

3.2 Non-disjoint group shuffled belief propagation algorithm

The so-called horizontal shuffled BP algorithm partitions the check nodes of the code graph into groups to perform group-by-group message-passing decoding. We propose a

new grouping technique to accelerate the message-passing rate by dividing CNs into non-disjoint CN groups and named it non-disjoint group shuffled belief propagation algorithm (NDGSBP). Fig. 3.3 is the schematic diagram of NDGSBP. In this section, we first explain why GS decoding with non-disjoint groups and give a easy example to demonstrate it. Then we define some notations and show the detail of the NDGSBP algorithm.

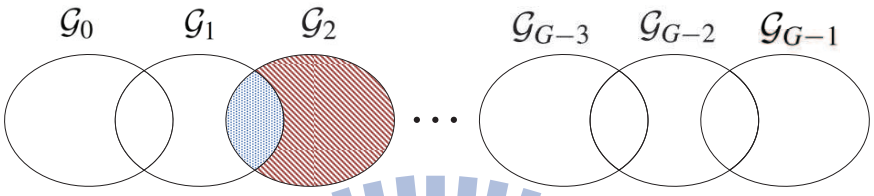


Figure 3.3: Grouping method of NDGSBP algorithm.

3.2.1 Why GS decoding with non-disjoint groups?

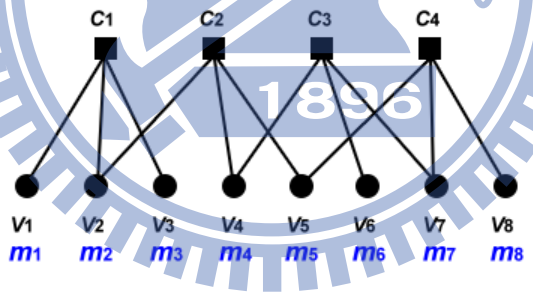


Figure 3.4: The Tanner Graph of A Linear Block Code.

Consider the decoding sub-iteration which performs VN-to-CN and then CN-to-VN message passing for the CNs of the kth group and all connecting VNs. If (at least) one of the VNs is linked to some CNs in other (CN) groups which have been processed in the same decoding iteration before (i.e., whose group indices are smaller than k), then other connecting VNs which have no such links will benefit from receiving more newly updated messages. We use a simple linear code and its associating Tanner graph shown in Fig.

3.4, where there are four CNs $\{c_1, c_2, c_3, c_4\}$ and eight VNs $\{v_1, v_2, \ldots, v_7, v_8\}$, to explain this effect. Let the messages the VNs carry be denoted by $m_1, m_2, \ldots, m_7, m_8$. In a conventional BP decoding iteration, each VN receives the messages from its neighboring VNs which are linked through some VNs. For instance, v_4 and v_6 are updated by the messages $\{m_2, m_5, m_6, m_7\}$ and $\{m_4, m_7\}$, respectively. For the GSBP decoding with two CN groups $\{c_1, c_2\}$ and $\{c_3, c_4\}$, v_4 receives $\{m_2, m_5\}$ in the first sub-iteration and $\{m_2, m_5, m_6, m_7\}$ in the second sub-iteration while v_6 is updated by $\{m_2, m_4, m_5, m_7\}$ in which m_2 and m_5 are the messages forwarded by v_4 because of its connection to the second CN group and will help improving the convergence. Obviously, the amount of messages the CNs in the k group receive from VNs connected to CNs belonging to the jth group, $j < k$ depends on the code structure and the grouping of CNs. If we limit our attention to the case $j = k - 1$, the CoCSG defined in the introductory section can be used to quantify the average amount of messages received from the previous sub-iteration and the grouping should try to maximize this number.

To simplify our systematic non-disjoint grouping method, we assume identical group cardinality, N_G, and denote the number of CN groups by G so that $G \times N_G = M$ is the number of CNs. We define the overlapping ratio r as the ratio between the size of the intersection between two neighboring CN groups and G. Then, we have, $GN_G - (G-1)N_G r = M$.

We arbitrary select N_G CNs to form the first CN group. The kth ($k > 1$) group includes $r \cdot N_G$ CNs randomly chosen from the $(k-1)$th group and $(1-r) \cdot N_G$ CNs from the CNs which do not belong to any of the earlier groups. Therefore, a CN does not necessarily belong to only one group anymore. As an illustration, we consider the grouping $(r, G.N_G) = (0.5, 3, 2)$ on the Tanner graph of Figure 3.4 again. Let the first group be $\{c_1, c_2\}$, the second one be $\{c_2, c_3\}$ and the third one be $\{c_3, c_4\}$. In the first sub-iteration, v_2 and v_4 receive $\{m_1, m_3, m_4, m_5\}$ and $\{m_2, m_5\}$, respectively. v_4 and v_6 receive $\{m_1, m_2, m_3, m_5, m_6, m_7\}$ and $\{m_2, m_4, m_5, m_7\}$ in the second sub-iteration, in

the final sub-iteration, v_6 will be updated by $\{m_1, m_2, m_3, m_4, m_5, m_7\}$. In short, for conventional BP, a VN can just collect information from VNs which are two-edge away in one iteration; for GSBP decoding, a VN has the opportunity to obtain the messages from four-edge-apart VNs; and for the proposed NDGSBP decoding algorithm, it is possible that a VN obtains the messages from VNs which are more than six-edge away if we select the overlapping ratio and CNs carefully. With fixed degree of parallelism N_G and CN number M, the larger r becomes, the longer the per-iteration delay is while the less the required iteration number becomes as a VN can update its LLR using information from more VNs. The product of the required iteration number and the per-iteration delay equals the total decoding delay to achieve a predetermined error rate performance. Section IV shows that the NDGSBP algorithm does give improved error rate performance for the same decoding delay.

3.2.2 Basic definitions and notations

A binary (N, K) LDPC code $\mathcal{C}$ is a linear block code whose $M \times N$ parity check matrix $\mathbf{H} = [H_{mn}]$ has sparse nonzero elements. And thus $\mathcal{C}$ can be viewed as a bipartite graph with N VNs corresponding to the encoded bits, and M CNs corresponding to the parity-check functions represented by the rows of $\mathbf{H}$. To track the statistical property variations of the message-passing sequence between VNs and CNs in an iterative decoding schedule, we also need to know the VN and CN degree-distribution polynomials $\lambda(x) = \sum_{i=2}^{d_v} \lambda_i x^{i-1}$ and $\rho(x) = \sum_{j=2}^{d_c} \rho_j x^{j-1}$, where λ_i and ρ_j denote the fraction of all edges connected to degree-i VNs and degree-j CNs, d_v and d_c denotes the maximum VN and CN degree.

Let $\mathcal{N}(m)$ be the set of variable nodes that participate in check node m and $\mathcal{M}(n)$ be the set of check nodes that are connected to variable node n in the code graph. $\mathcal{N}(m)\backslash n$ is defined as the set $\mathcal{N}(m)$ with the variable node n excluded while $\mathcal{M}(n)\backslash m$ is the set $\mathcal{M}(n)$ with the check node m excluded. Let $L_{n\to m}$ be the message sent from VN n to

CN m and $L_{m\to n}$ be the message sent from CN m to VN n.

3.2.3 System model and decoding schedule

Assume a codeword $\mathbf{C} = (c_1, c_2, ..., c_N)$ is BPSK-modulated and transmitted over an AWGN channel with noise variance σ^2. Let $\mathbf{Y} = (y_1, y_2, ..., y_N)$ be the corresponding received sequence and L_n be the log-likelihood ratio (LLR) of the variable node n with the initial value given by $L_n = \frac{2}{\sigma^2} y_n$.

Let $\mathcal{G}_g$ be the gth CN group, $1 \leq g \leq G$ and $\mathcal{U}$ be a set of CNs, l as the iteration counter and I_{Max} as the maximum number of iterations. We can then describe the NDGSBP algorithm as follows:

Initialization

Set $l = 1, \mathcal{U} = \{x | 1 \leq x \leq M\}$, and $\mathcal{G}_g = \emptyset$ for $1 \leq g \leq G$.

Step 1: Grouping check nodes

Collect N_G elements randomly from the set $\mathcal{U}$ to form $\mathcal{G}_1$, let $\mathcal{U} = \mathcal{U} \backslash \mathcal{G}_1$. Collect $N_G - N_G \cdot r$ element randomly from the set $\mathcal{U}$ and $N_G \cdot r$ elements from $\mathcal{G}_1$ to create $\mathcal{G}_2$. For $3 \leq g \leq G$, collect $N_G - N_G \cdot r$ element randomly from the set $\mathcal{U}$ and $N_G \cdot r$ elements from $\mathcal{G}_{g-1} \backslash \mathcal{G}_{g-2}$ to create $\mathcal{G}_g$ and let $\mathcal{U} = \mathcal{U} \backslash \mathcal{G}_g$.

Step 2: Message passing

For $1 \leq g \leq G$

a) CN update: $\forall\ m \in \mathcal{G}_g, n \in \mathcal{N}(m)$

$$L_{m\to n} = 2 \tanh^{-1} \left(\prod_{n' \in \mathcal{N}(m) \backslash n} \tanh \left(\frac{1}{2} L_{n'\to m} \right) \right) \tag{3.6}$$

b) VN update: $\forall\ n \in \bigcup_{m' \in \mathcal{G}_g} \mathcal{N}(m'), m \in \mathcal{M}(n)$

$$L_{n\to m} = L_n + \sum_{m' \in \mathcal{M}(n) \backslash m} L_{m'\to n} \tag{3.7}$$

Step 3: Total LLR computation

$\forall n, 1 \leq n \leq N$,

$$L_n^{total,(l)} = L_n + \sum_{m' \in N(n)} L_{m' \to n} \tag{3.8}$$

Step 4: Hard decision and stopping criterion test

a) Create $\mathbf{D}^{(l)} = [d_1^{(l)}, d_2^{(l)}, ..., d_N^{(l)}]$ such that $d_n^{(l)} = 0$ if $L_n^{total,(l)} \geq 0$ and $d_n^{(l)} = 1$ if $L_n^{total,(l)} < 0$.

b) If $\mathbf{D}^{(l)}\mathbf{H}^T = \mathbf{0}$ or I_{Max} is reached, stop decoding and output $\mathbf{D}^{(l)}$ as the decoded codeword. Otherwise, set $l = l + 1$ and $\mathcal{U} = \{x | 1 \leq x \leq M\}$, go to **Step 1**.

Chapter 4

Convergence Analysis

In this chapter, we adopt the Gaussian approximation approach to analyze the convergence behavior of the proposed algorithms. The basic analytic approach follows that presented in Section 2.4 but takes the scheduling into account.

4.1 GA for shuffled belief propagation algorithm

We first consider two basic types of shuffled BP decoding schedules.

4.1.1 GA for VSBP algorithm

Consider a degree-j CN m and suppose it is connected with k $(k = 0, 1, \ldots, j-1)$ VNs which belongs to the group V_1 and $j-k$ VNs which are in group V_2. For such a CN we obtain

$$E\left\{\tanh\left(\frac{c_{j,k}^{(l)}}{2}\right)\right\} = \left[E\left\{\tanh\left(\frac{v^{(l),V_1}}{2}\right)\right\}\right]^k \cdot \left[E\left\{\tanh\left(\frac{v^{(l-1)}}{2}\right)\right\}\right]^{j-k-1}. \quad (4.1)$$

Let $v_i^{(l),V_1}$ be the message sent by degree-i VN which belongs to V_1, and $v_i^{(l),g}$ be the message sent by degree-i VN in group g at the lth iteration. Then (4.1) can be written as

$$\mu_{c,j,k}^{(l)} = \Phi^{-1}\left(1 - \left(1 - \sum_{i=2}^{d_{vMax}} \lambda_i \Phi\left(\mu_{v,i}^{(l-1)}\right)\right)^{j-k-1} \cdot \left(1 - \sum_{i=2}^{d_{vMax}} \lambda_i \Phi\left(\mu_{v,i}^{(l),V_1}\right)\right)^k\right), \quad (4.2)$$

where

$$\mu_{v,i}^{(l),V_1} = \frac{1}{g}\sum_{g'=0}^{g-1}\mu_{v,i}^{(l),g'}, \tag{4.3}$$

and $\Phi(\cdot)$ is defined in (2.18). The mean of degree-j CN messages $\mu_{c,j}^{(l)}$ is obtained by accumulating all possible values of $\mu_{c,j,k}^{(l)}$ with their corresponding coefficients $P_{j,k}$:

$$\mu_{c,j}^{(l)} = \sum_{k=0}^{j-1} P_{j,k}\cdot\mu_{c,j,k}^{(l)}, \tag{4.4}$$

where $P_{j,k}$ is the proportion of degree-j CNs which have k neighboring VNs belonging to V_1 in all degree-j CNs. Thus $P_{j,k}$ is given by

$$P_{j,k} = \begin{cases} 1, & g=0, k=0 \\ C_k^{j-1}\cdot\left(\frac{g}{G}\right)^k\cdot\left(1-\frac{g}{G}\right)^{j-1-k}, & otherwise \end{cases}. \tag{4.5}$$

By linearly combining the means of degree-$2,\ldots,d_{cMax}$ CN messages with weights $\{\rho_j, 2\le j\le d_c\}$, the mean of the CN messages $\mu_c^{(l)}$ is given by

$$\mu_c^{(l)} = \sum_{j=2}^{d_c}\rho_j\cdot\mu_{c,j}^{(l)}. \tag{4.6}$$

A VN with degree-i collects the messages from $i-1$ connected CNs as well as the channel initial message u_0, hence we have

$$\mu_{v,i}^{(l),g} = \mu_{c_0} + (i-1)\mu_c^{(l)}. \tag{4.7}$$

After linearly combining $\mu_{v,i}^{(l),g}$ $(i=2,3,\ldots,d_{vMax})$ with λ_i, $\mu_v^{(l),g}$ becomes

$$\mu_v^{(l),g} = \sum_{i=2}^{d_{vMax}}\lambda_i\cdot\mu_v^{(l),g}. \tag{4.8}$$

4.1.2 GA for HSBP algorithm

Consider a degree-j CN m. For such CNs we obtain

$$E\left\{\tanh\left(\frac{c_j^{(l),g}}{2}\right)\right\} = \left[E\left\{\tanh\left(\frac{v^{(l)}}{2}\right)\right\}\right]^{j-1}, \tag{4.9}$$

where $v^{(l)}$ is the message sent by VNs at the lth iteration, $c_j^{(l),g}$ is the message sent by a degree-j CN in group g at the lth iteration. Then (4.9) can be rewritten as

$$\mu_{c,j}^{(l),g} = \Phi^{-1}\left(1-\left(1-\sum_{i=2}^{d_{vMax}}\lambda_i\Phi\left(\mu_{v,i}^{(l)}\right)\right)^{j-1}\right). \tag{4.10}$$

By linearly combining the means of degree-$2,\ldots,d_c$ CN messages with weights $\{\rho_j, 2\leq j\leq d_{cMax}\}$, the mean of CN messages $\mu_c^{(l),g}$ is obtained from

$$\mu_c^{(l),g} = \sum_{j=2}^{d_{cMax}}\rho_j\cdot\mu_{c,j}^{(l),g}. \tag{4.11}$$

Consider a degree-i VN n which is connected to k $(k=0,1,\ldots,i-1)$ CNs in group C_1 and $i-k$ CNs in group C_2. For such a VN, we have

$$\mu_{v,i,k}^{(l)} = \mu_{c_0} + k\mu_c^{(l),C_1} + (i-k-1)\mu_c^{(l-1)}, \tag{4.12}$$

where

$$\mu_c^{(l),C_1} = \frac{1}{g}\sum_{g'=0}^{g-1}\mu_c^{(l),g'}, \tag{4.13}$$

$$\mu_c^{(l-1)} = \frac{1}{G}\sum_{g'=0}^{G-1}\mu_c^{(l-1),g'}. \tag{4.14}$$

The mean of degree-i VN messages $\mu_{v,i}^{(l)}$ is obtained by accumulating all possible values of $\mu_{v,i,k}^{(l)}$ with their corresponding coefficients $P_{i,k}$:

$$\mu_{v,i}^{(l)} = \sum_{k=0}^{i-1}P_{i,k}\cdot\mu_{v,i,k}^{(l)}, \tag{4.15}$$

where $P_{i,k}$ is the ratio of degree-i VNs which have k neighboring CNs belonging to C_1 among all degree-i CNs. Thus $P_{i,k}$ is given by

$$P_{i,k} = \begin{cases} 1, & g=0, k=0 \\ C_k^{i-1}\cdot\left(\frac{g}{G}\right)^k\cdot\left(1-\frac{g}{G}\right)^{i-1-k}, & otherwise \end{cases}. \tag{4.16}$$

By linearly combining $\mu_{v,i}^{(l)}$ $(i=2,3,\ldots,d_{vMax})$ with λ_i, $\mu_v^{(l)}$ is given by

$$\mu_v^{(l)} = \sum_{i=2}^{d_{vMax}}\lambda_i\cdot\mu_{v,i}^{(l)}. \tag{4.17}$$

4.2 GA for hybrid-shuffled belief propagation algorithm

Since H-SBP perform VSBP and HSBP alternatively, the GA for H-SBP execute GA for VSBP and HSBP by turns.

4.3 GA for non-disjoint group shuffled belief propagation algorithm

As can be seen from the above description of the proposed algorithm, the messages $L_{n\to m}$ and $L_{m\to n}$ are real random variables that depend on the received channel values y_n, the code structure and the decoding schedule. The GA approach assumes that they can be approximated by Gaussian random variables. With this approach, we need only to monitor the message means as the consistency condition holds in our case [11]. We further assume that the all-zero codeword $\mathbf{C} = (0, 0, \ldots, 0)$, which is mapped into the BPSK modulated vector $\mathbf{X} = (1, 1, \ldots, 1)$, is transmitted. The following analysis is based on the ideas of [12] and [15] with two distinct considerations. First, the analysis presented in [15] deals with vertical GSBP while we are dealing with horizontal GSBP. Second, the intersection among groups can be nonempty in our schedule. For GSBP decoding, we divide CNs into two types, one is updated CNs and the other is non-updated CNs. As depicted in Fig.4.1. To analyze the effect of nonempty intersections, we divide CNs into four classes in a given, say the gth sub-iteration of the lth iteration. Class-**a** includes the CNs that will be updated at the g'th ($g' > g$) sub-iteration, Class-**b** includes the CNs which are also members of the previous $(g-1)$th group, Class-**c** contains the CNs which are not members of the previous $(g-1)$th group and the Class-**d** are all CNs exclude Class-**a** and Class-**b**. Figs.4.2 and 4.3 depict the situations after three sub-iterations for overlapping ratio $r < 0.5$ and $0.5 \leq r \leq 1$ respectively. We now track the average values of all updated parameters at the lth iteration for the proposed NDGSBP algorithm. We first define $\mu_{c_{\mathbf{x}}^{g,(l)}}$ as the mean of the message sent by a Class-**x**

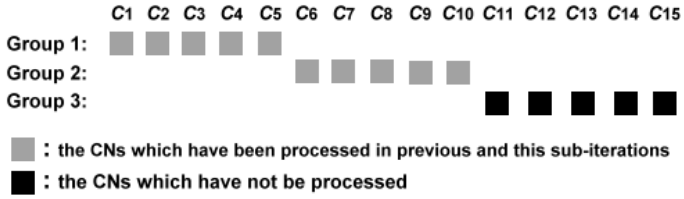


Figure 4.1: A example for GSBP after two sub-iterations.

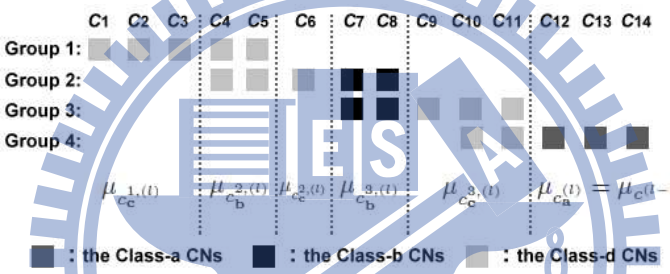


Figure 4.2: A example for NDGSBP after three sub-iterations when $r < 0.5$.

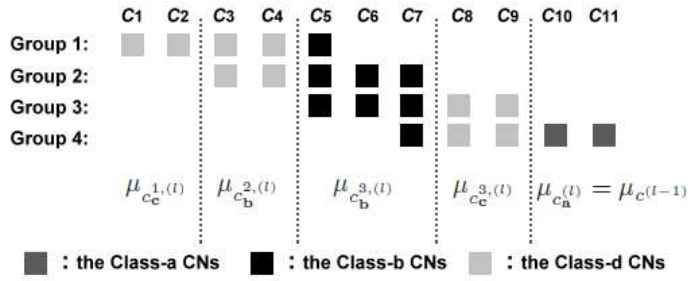


Figure 4.3: A example for NDGSBP after three sub-iterations when $0.5 \leq r \leq 1$.

CN, that is, $\mu_{c_{\mathbf{x}}^{g,(l)}} = E\{L_{m\to n}^{g,(l)}\}$, where m belong to Class-**x** CNs, n is a VN connecting to m in the gth sub-iteration of the lth iteration. We start with the VN update equation. Consider the degree-i VN n which is connected to p Class-**d** CNs, q Class-**b** CNs and $i-p-q$ Class-**a** CNs. For the gth sub-iteration of the lth iteration, we have, for $g = 1$,

$$\begin{aligned} \mu_{v_{i,p,q}^{(l)}} &= \mu_0 + p\mu_{c_{\mathbf{d}}^{(l)}} + q\mu_{c_{\mathbf{b}}^{g,(l)}} \\ &\quad +(i-p-q-1)\mu_{c_{\mathbf{a}}^{(l)}} \end{aligned} \tag{4.18}$$

$$\begin{aligned} &= \mu_0 + p\mu_{c_{\mathbf{d}}^{(l)}} + q\mu_{c_{\mathbf{b}}^{g,(l)}} \\ &\quad +(i-p-q-1)\mu_{c^{(l-1)}} \end{aligned} \tag{4.19}$$

where $\mu_{c_{\mathbf{d}}^{(l)}} = \mu_{c_{\mathbf{c}}^{1,(l)}}$ and $\mu_0 \triangleq E\{L_n\} = E\{\frac{2y_n}{\sigma^2}\}$ is the mean of the channel value. For $g > 1$, we obtain

$$\mu_{c_{\mathbf{d}}^{(l)}} = \frac{1}{g}\left(\mu_{c_{\mathbf{c}}^{1,(l)}} + \mu_{c_{\mathbf{c}}^{g,(l)}} + \sum_{g'=2}^{g-1}\left(\frac{r}{1-r}\mu_{c_{\mathbf{b}}^{g',(l)}} + \frac{1-2r}{1-r}\mu_{c_{\mathbf{c}}^{g',(l)}}\right)\right), \tag{4.20}$$

for $r < 0.5$ and

$$\mu_{c_{\mathbf{d}}^{(l)}} = \frac{1}{g}\left(\mu_{c_{\mathbf{c}}^{1,(l)}} + \mu_{c_{\mathbf{c}}^{g,(l)}} + \sum_{g'=2}^{g-1}\mu_{c_{\mathbf{b}}^{g',(l)}}\right), \tag{4.21}$$

for $0.5 \le r \le 1$.

When the CNs in the g-th group are processed in l-th iteration, the mean of message for degree-i VNs $\mu_{v_i^{(l)}}$ can be obtained by accumulating all possible values of $\mu_{v_{i,p,q}^{(l)}}$ with their corresponding coefficients $\omega(i,p,q)$:

$$\mu_{v_i^{(l)}} = \sum_{p=0}^{i-1}\sum_{q=0}^{i-1-p}\omega(i,p,q)\cdot\mu_{v_{i,p,q}^{(l)}}, \tag{4.22}$$

where $\omega(i,p,q)$ is the proportion of degree-i VNs which have p neighboring Class-**d** CNs, q neighboring Class-**b** CNs in all degree-i CNs. Thus $\omega(i,p,q)$ is given by

$$\omega(i,p,q) = \begin{cases} \binom{i-1}{p}x^p(1-x)^{i-1-p}, & g=1 \\ \binom{i-1}{p}\binom{i-1-p}{q}y^pz^q(1-y-z)^{i-1-p-q}, & g\neq 1 \end{cases} \tag{4.23}$$

where x is the fraction of Class-**d** CNs for $g = 1$, y is the fraction of Class-**d** CNs and z is the fraction of Class-**b** CNs.

Thus

$$x = \frac{1}{G-(G-1)r}, \tag{4.24}$$

$$y = \frac{g(1-r)}{G-(G-1)r}, \tag{4.25}$$

$$z = \frac{r}{G-(G-1)r}. \tag{4.26}$$

From Class-**c** CNs updating formula, we can obtain

$$E\left\{\tanh\left(\frac{c_{\mathbf{c},j}^{g,(l)}}{2}\right)\right\} = \left[E\left\{\tanh\left(\frac{v^{(l)}}{2}\right)\right\}\right]^{j-1}. \tag{4.27}$$

Under the Gaussian approximation and for $\mu \geq 0$, define

$$\Phi(\mu) \triangleq 1 - \frac{1}{\sqrt{4\pi\mu}}\int_{-\infty}^{\infty}\tanh(\frac{\tau}{2})\exp\left[\frac{-(\tau-\mu)^2}{(4\mu)}\right]d\tau, \tag{4.28}$$

and (4.27) can be rewritten as

$$\mu_{c_{\mathbf{c},j}^{g,(l)}} = \Phi^{-1}\left(1-\left(1-\sum_{i=2}^{d_{vMax}}\lambda_i\Phi\left(\mu_{v_i^{(l)}}\right)\right)^{j-1}\right). \tag{4.29}$$

If we average over all CN degree j, we have

$$\mu_{c_{\mathbf{c}}^{g,(l)}} = \sum_{j=2}^{d_{cMax}}\rho_j \cdot \mu_{c_{\mathbf{c},j}^{g,(l)}}. \tag{4.30}$$

The computation of the mean of message send from a Class-**b** CN $\mu_{c_{\mathbf{b}}^{g,(l)}}$ is replace $\mu_{v_i^{(l)}}$ with $\mu_{v_i'^{(l)}}$ in (4.29) where $\mu_{v_i'^{(l)}}$ is mean of message send from a previous group overlapping VN. And $\mu_{v_i'^{(l)}}$ is got by let p at least 1 in (4.22) and (4.23) for $g \neq 1$.

After l iterations, the mean of the message passed from a CN $\mu_{c^{(l)}}$ is

$$\mu_{c^{(l)}} = \frac{r}{G-(G-1)r}\mu_{c_{\mathbf{b}}^{G,(l)}} + \frac{G-Gr}{G-(G-1)r}\mu_{c_{\mathbf{c}}^{G,(l)}}. \tag{4.31}$$

If $\mu_{c^{(l)}} \to \infty$, the connecting VNs achieve error free performance.

Chapter 5

Numerical Results

Figs. 5.1 and 5.2 depict the BER and FER performance of Mackay's (504,252) regular LDPC code with $d_c = 6$, $d_v = 3$ using the standard BP algorithm, the SBP algorithm ($G = 4, 12$) and the proposed H-SBP algorithm ($G = 4, 12$). On the other hand, in Figs. 5.3 and 5.4 we plot the FER and BER performance of Mackay's (408,204) regular LDPC code with $d_c = 6$ and $d_v = 3$ using the standard BP algorithm, the SBP algorithm ($G = 4, 12$) and the proposed H-SBP algorithm ($G = 4, 12$), respectively.

Fig. 5.5 depicts the BER performance of Mackay's (504,252) regular LDPC code with $d_c = 6$, $d_v = 3$ using the standard BP algorithm, the HSBP algorithm ($G = 12$) and the proposed NDGSBP algorithm ($G = 12$, overlapping ratio $r = 0.2, 0.4$). On the other hand, in Fig. 5.6 we show the BER performance of Mackay's (408,204) regular LDPC code with $d_c = 6$ and $d_v = 3$ using the standard BP algorithm, the HSBP algorithm ($G = 12$) and the proposed NDGSBP algorithm ($G = 12$, overlapping ratio $r = 0.2, 0.4$), respectively.

Since our two approaches are independent, we can combine them to get more improvement. Fig. 5.7 shows the BER and FER performance of Mackay's (504,252) regular LDPC code with $d_c = 6$, $d_v = 3$ using respectively the standard BP algorithm, the HSBP algorithm ($G = 12$) and the proposed algorithm ($G = 12$, overlapping ratio $r = 0.2, 0.4$).

The simulation results reported in this chapter assume $I_{Max} = 500$ for the HSBP and BP algorithms. To have fair comparison, we assume the system parameter values that

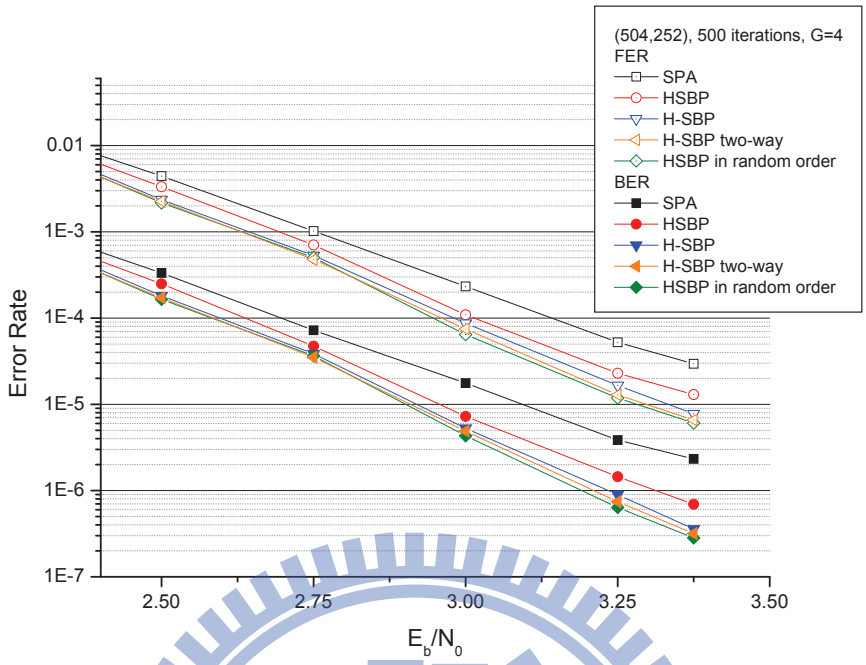


Figure 5.1: Error rate of a $(504, 252)$ $(3, 6)$ LDPC code with standard BP , HSBP and H-SBP decoding with $G = 12$.

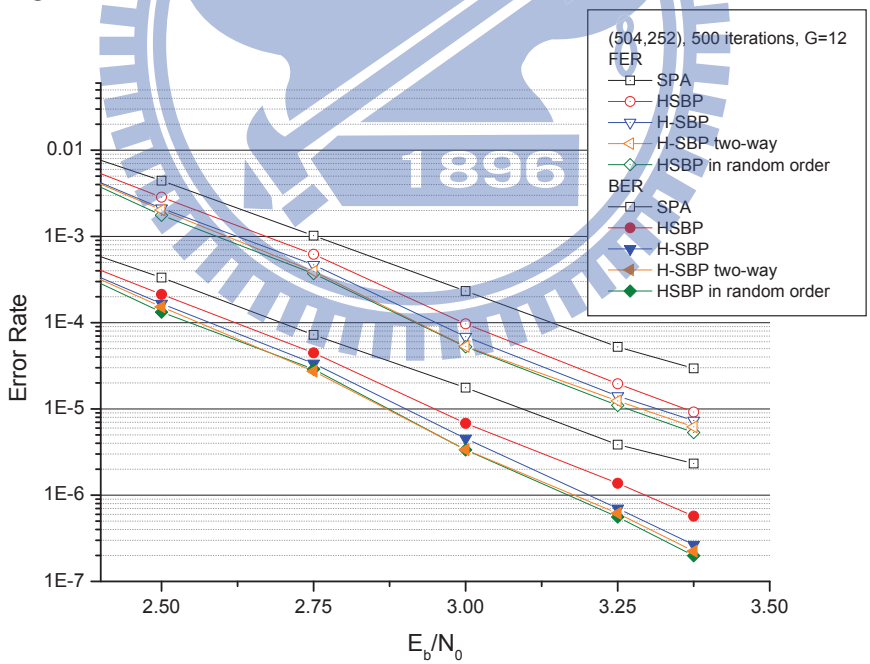


Figure 5.2: Error rate of a $(504, 252)$ $(3, 6)$ LDPC code with standard BP , HSBP and H-SBP decoding with $G = 12$.

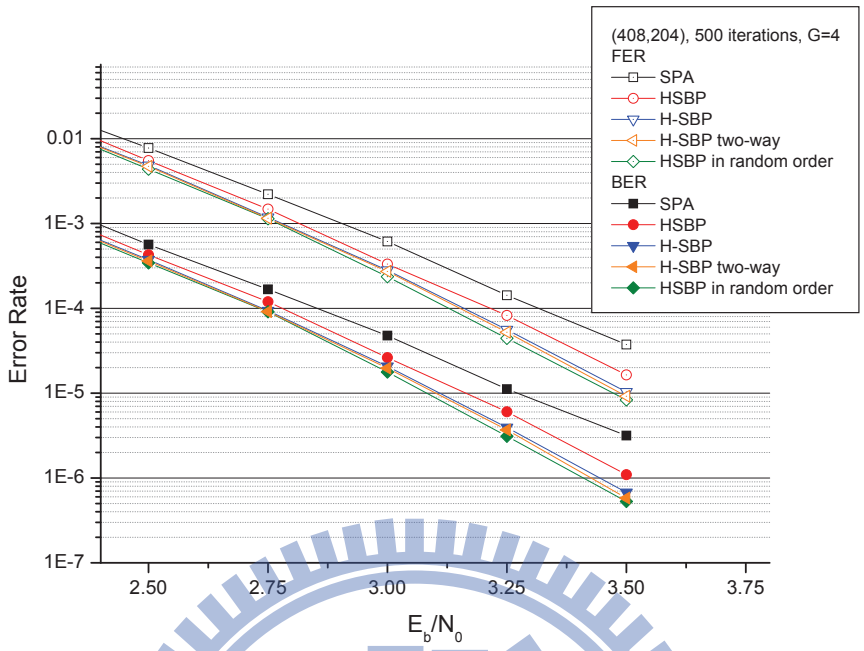


Figure 5.3: Error rate of a $(408, 204)$ $(3, 6)$ LDPC code with standard BP , HSBP and H-SBP decoding with $G = 12$.

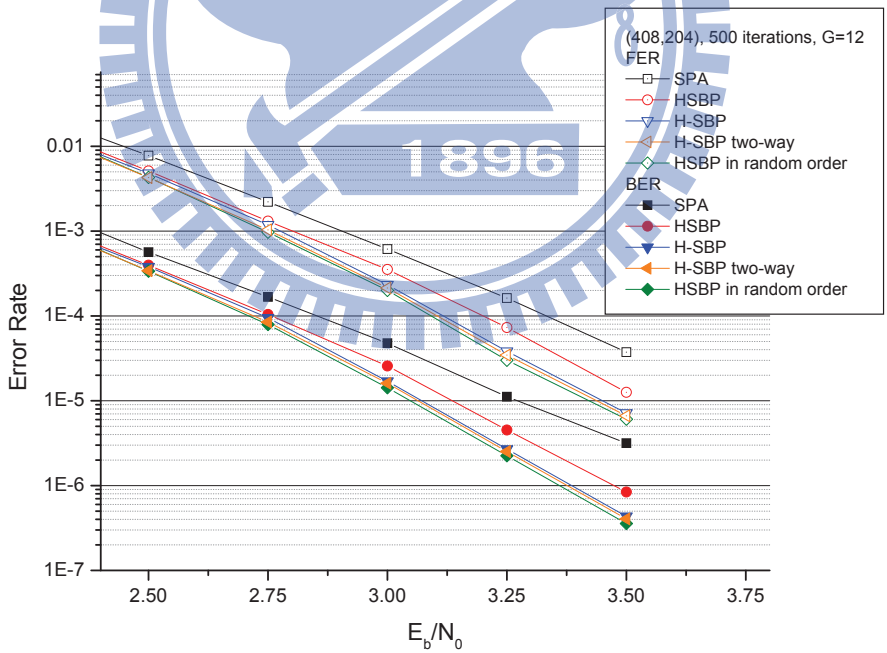


Figure 5.4: Error rate of a $(408, 204)$ $(3, 6)$ LDPC code with standard BP , HSBP and H-SBP decoding with $G = 12$.

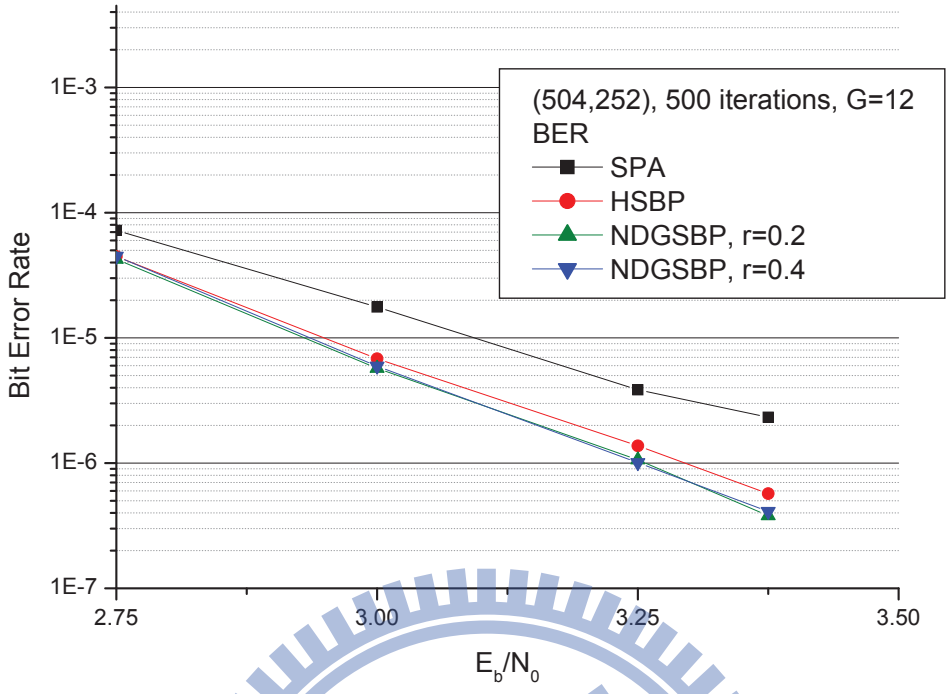


Figure 5.5: BER performance of Mackay's (504,252) regular LDPC code with $d_c = 6$ and $d_v = 3$ using the decoding algorithms: NDGSBP, HSBP for $G = 12$ and standard BP.

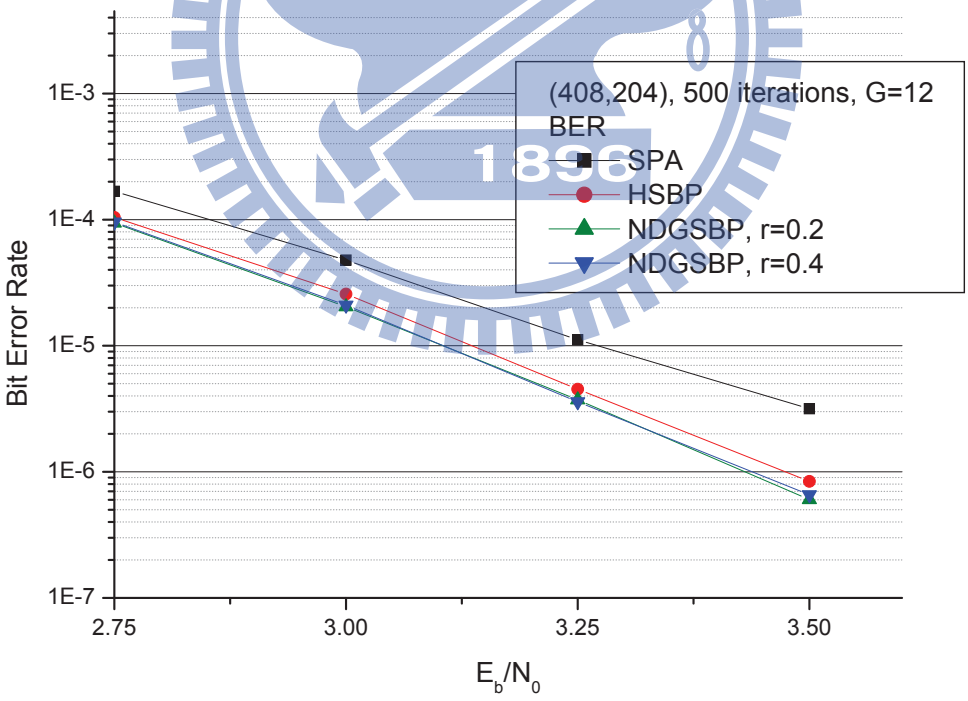


Figure 5.6: BER performance of Mackay's (408,204) regular LDPC code with $d_c = 6$ and $d_v = 3$ using the decoding algorithms: NDGSBP, HSBP for $G = 16$ and standard BP.

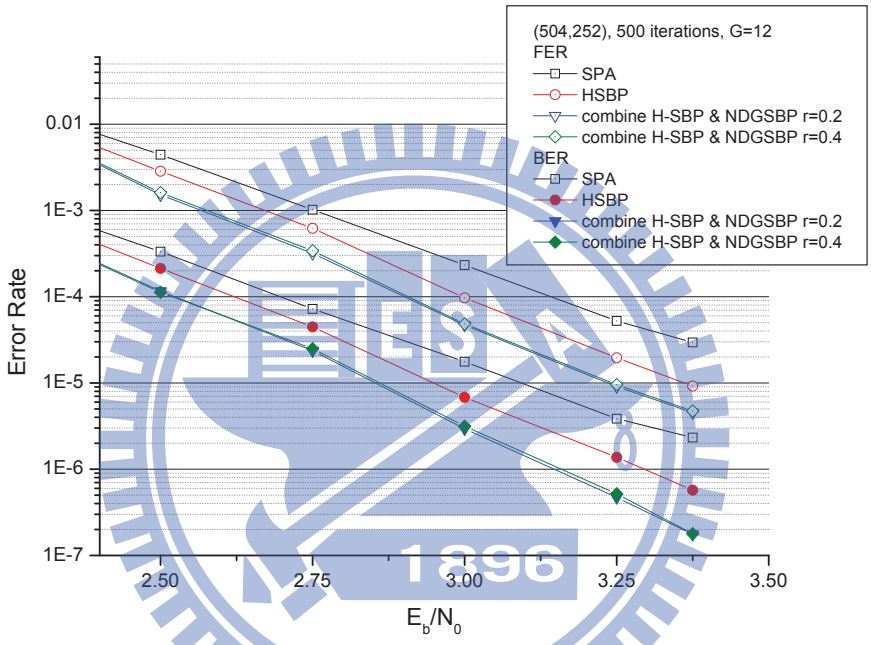


Figure 5.7: BER performance of Mackay's (504,252) regular LDPC code with $d_c = 6$ and $d_v = 3$ using the decoding algorithms: proposed algorithm, HSBP for $G = 12$ and standard BP.

result in the same or similar computation complexity for all algorithms. For example, to decode the (504,252) LDPC code using the NDGSBP decoder with $G = 12$ and $r = 0.4$ imply that $N_G = 34$ and it is allowed to have at most $\frac{m \cdot I_{Max}}{m+(G-1)\times N_G \cdot r} = \frac{252 \cdot 500}{252+11 \cdot 34 \cdot 0.4} \approx 310$ decoding iterations.

We use the GA approach outlined in Chapter 4 to analyze the performance of the H-SBP, NDGSBP, BP and HSBP decoders. Given the code rate and degree distribution of LDPC codes, the thresholds estimated by the GA approach for BP, HSBP H-SBP and NDGSBP decoding are the same. In Table 5.2 and Table 5.1, we list the number of iterations for error free performance at SNR equals threshold. We examine the NDGSBP performance in decoding two ensemble LDPC codes using the same overlapping ratio $r = 0.2$ but different group number G. The table shows the H-SBP and NDGSBP decoder consistently outperform the other two decoders in convergence rate.

Table 5.1: Gaussian approximation for the binary-input AWGN channel under BP, HSBP and H-SBP. We listed the number of iterations for exceed BER 10^{-10} at $(E_b/N_0) =$ threshold.

					HSBP			H-SBP		
d_v	d_c	R	$(E_b/N_0)_{\mathrm{GA}}$	BP	$G=4$	12	36	$G=4$	12	36
3	6	1/2	1.163	424	294	263	252	270	220	201
					HSBP			H-SBP		
d_v	d_c	R	$(E_b/N_0)_{\mathrm{GA}}$	BP	$G=4$	16	34	$G=4$	16	34
4	6	1/3	1.730	633	439	387	377	403	317	301

Table 5.2: Gaussian approximation for the binary-input AWGN channel under BP, HSBP and NDGSBP with $r = 0.2$. We listed the number of iterations for exceed BER 10^{-10} at $(E_b/N_0) =$ threshold.

					HSBP			NDGSBP		
d_v	d_c	R	$(E_b/N_0)_{\mathrm{GA}}$	BP	$G=4$	12	36	$G=4$	12	36
3	6	1/2	1.163	424	294	263	252	272	239	230
					HSBP			NDGSBP		
d_v	d_c	R	$(E_b/N_0)_{\mathrm{GA}}$	BP	$G=4$	16	34	$G=4$	16	34
4	6	1/3	1.730	633	439	387	377	407	360	353

Chapter 6

Conclusion

In this thesis, we propose two novel group shuffled BP decoding scheduling schemes to improve the performance of the conventional GSBP algorithm for decoding LDPC codes. The proposed NDGSBP scheme enhances the connectivity of the code graph by having overlapped CNs in neighboring CN groups. The enhanced connectivity allows each VN (or CN) to obtain related information from more VNs (or CNs) within a decoding iteration, accelerating the message-passing rate and thus the convergence speed. The H-SBP performs VSBP and HSBP decoding alternately to achieve a pseudo-random decoding schedule to avoid the unequal error-correcting capability of the coded bits. As a result, the H-SBP scheme gives better convergence performance than that of the conventional GSBP approach.

We also analyze the decoding behavior of different decoding schedules in this thesis. The GA approach is used to track the first-order statistical information flow of the proposed NDGSBP algorithm and the H-SBP algorithm. The GA analysis verifies that the NDGSBP decoder and H-SBP decoder do give faster convergence rates with respect to those of the GSBP and BP decoders. Numerical results also demonstrate that, with the same decoding computation complexity, the new algorithms yields improved BER and FER performance.

For NDGSBP decoding, the VNs are grouped in natural increasing order and the

non-disjoint parts are randomly selected from the available CNs. A proper VN ordering and overlapping VN selection that take the code structure into account will certainly give better performance. The optimal decoding schedule and the optimal CN group overlapping ratio r remain to be found and some analytic performance metrics may be needed in our search of the desired solution.

Bibliography

[1] R. G. Gallager, *Low-density parity-check codes*, Cambridge, MA: M.I.T. Press, 1963.

[2] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inf. Theory*, vol. IT-27, pp. 533V547, Sept. 1981.

[3] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electronics Lett.*, vol. 32, no. 18, pp. 1645-1646, Aug. 1996.

[4] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, pp. 399-431, Mar. 1999.

[5] J. Zhang and M. Fossorier, "Shuffled belief propagation decoding," *IEEE Trans. Commun.*, vol. 53, pp. 209-213, Feb. 2005.

[6] A. Segard, F. Verdier, D. Declercq and P. Urard, "A DVB-S2 compliant LDPC decoder integrating the horizontal shuffle schedule," In *Proc. of IEEE ISPACS 2006*, pp. 1013-1016, Dec. 2006.

[7] E. Sharon, S. Litsyn and J. Goldberger, "Efficient serial message-passing schedules for LDPC decoding," *IEEE Trans. Inf. Theory*, vol. 53, pp. 4076 - 4091, Nov. 2007.

[8] J. Zhang, Y. Wang, M. P. C. Fossorier and J. S. Yedidia, "Iterative decoding with replicas," *IEEE Trans. Inf. Theory*, vol. 53, No. 5, pp. 1644-1663, May 2007.

[9] C.-Y. Chang, Y.-L. Chen, C.-M. Lee, and Y. T. Su, "New group shuffled BP decoding algorithms for LDPC codes," presented at *IEEE ISIT 2009*, pp. 1664-1668, Jun. 2009.

[10] Y. Yang, J.-Z. Huang, S. Tong and X.-M. Wang, "Replica horizontal-shuffled iterative decoding of low-density parity-check codes," *The Journal of China Universities of Posts and Telecommunications*, vol. 13, pp.32-40, Jun. 2010.

[11] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, No. 2, pp. 599-617, Feb. 2001.

[12] S.-Y. Chung and T. J. Richardson, "Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation," *IEEE Trans. Inf. Theory,* vol. 47, No. 2, pp. 657-670, Feb. 2001.

[13] S. T. Brink and G. Kramer, "Design of low-density parity-check codes for modulation and detection," *IEEE Trans. commun.*, vol. 52, No. 4 pp. 670-678, Apr. 2004.

[14] E. Sharon and A. Ashikhmin, "Analysis of low-density parity-check codes based on EXIT functions," *IEEE Trans. commun.*, vol. 54, No. 8, pp. 1407-1414, Aug. 2006.

[15] Z. Song, R. Yu, and P. Ma, "Gaussian approximation for LDPC codes under group shuffled belief propagation decoding," presented at *WiCOM 2010*, pp. 1-4, sep. 2010.