Proceedings of APCC2008 copyright (c) 2008 IEICE 08 SB 0083

Shuffled BP Decoding for Punctured LDPC Codes

Sangjoon Park
School of Electrical and Electronic Engineering
Yonsei University
azssa@yonsei.ac.kr

Sunyoung Lee
School of Electrical and Electronic Engineering
Yonsei University
sunyoung@yonsei.ac.kr

Keumchan Whang
School of Electrical and Electronic Engineering
Yonsei University
kcwhang@yonsei.ac.kr

***Abstract*—In this paper, we propose a scheduling method of the shuffled BP decoding algorithm for punctured LDPC codes. The number of iteration required to recover whole punctured variable nodes in the standard BP decoding algorithm is increased with the nubmer of punctured variable nodes. However, only one number of iteration is required in the proposed shuffled BP scheduling regardless of the puncturing algorithm and the number of punctured variable nodes. Furthermore, the proposed shuffled BP scheduling can improve the reliability of the unpunctured variable nodes at the 1st iteration, while impossible in the layered BP scheduling [5] if there are no check nodes that only unpunctured variable nodes participate. Simulation results verify that the proposed shuffled BP scheduling has comparable performance to the layered BP scheduling [5] in spite of its smaller number of scheduling group at rate below 0.9.**

***Index Terms*—LDPC(Low Density Parity Check) Codes; Shuffled BP(Belief Propagation) Decoding; Puncturing; SR(Step-Recovery);**

I. Introduction

Increasing demands for high data rate in wireless communication services require advanced techniques that enable reliable link connections and efficient processes for both system and mobile. One of the techniques in error-control coding area is LDPC (Low Density Parity Check) codes, which were invented in early 1960s by Gallagar [1] and rediscovered by Mackay [2] in 1999. LDPC codes have been widely studied in recent years because of their powerful error-correcting capability and low decoding complexity compared to Turbo codes[8].

There are several algorithms to decode LDPC-encoded codewords. Among them, the standard belief propagation (standard BP) decoding algorithm [2] is the most conventional decoding algorithm for LDPC codes. The standard BP algorithm updates the check nodes and the variable nodes simultaneously in parallel way. By its parallel mechanism, the standard BP algorithm requires a lot of decoding iteration for the convergence of decoding. In addition, if puncturing is applied to a codeword, the number of iteration for the convergence of decoding is increased more and more with the number of punctured variable nodes.

To enhance the convergence speed of the standard BP algorithm, some BP-based decoding algorithms operated in serial ways are suggested. The shuffled BP decoding algorithm [3] is one kind of serial decoding algorithm to decode LDPC codes that is based on a serial update of variable nodes. The layered BP decoding algorithm in [4] is also using a serial scheduling based on a serial update of check node. However, in [3], only a natural scheduling, which is based on a natural increasing order, is applied for the shuffled BP decoding, and, in [4], no specific scheduling method is mentioned.

In [5], scheduling algorithm using the layered BP decoding algorithm is proposed for RCP-LDPC (rate compatible punctured LDPC) codes. The layered BP scheduling algorithm in [5] sort check nodes by related punctured variable nodes and thus accelerates the decoding convergence. The scheme, however, cannot update the unpunctured variable nodes at the 1st iteration if there are no check nodes in which only unpunctured variable nodes participate.

In this paper, we propose a scheduling method of the shuffled BP decoding algorithm for punctured LDPC codes. The proposed shuffled BP scheduling generates scheduling groups by the level of recoverability of each variable node, so only one number of iteration is required to recover whole punctured variable nodes. Furthermore, the proposed shuffled BP scheduling can improve the reliability of unpunctured variable nodes at the 1st iteration, while impossible in the layered BP scheduling [5] if there are no check nodes in which only unpunctured variable nodes participate.

The paper is organized as follows. In Section II, we present the iterative BP decoding algorithm including the shuffled BP decoding algorithm. The proposed shuffled BP scheduling algorithm and its properties are given in Section III. Section IV shows computer simulation results of the proposed shuffled BP scheduling and other conventional BP decoding schemes, and Section V make conclusions.

II. Iterative Decoding Algorithm

Consider a binary (N, K) LDPC code of length N with parity check matrix $\mathbf{H} = [H_{mn}]$. The number of variable nodes and check nodes are N and M $(= N - K)$, respectively. We denote the set of bits that participate in check m by $N(m) = \{n : H_{mn} = 1\}$, and the set of checks in which bit n participates as $M(n) = \{m : H_{mn} = 1\}$. Assume a codeword $\overline{\mathbf{w}} = \{w_1, w_2, \cdots, w_N\}$ is transmitted over an additive white Gaussian noise (AWGN) channel with zero mean and variance $N_0/2$ using binary phase shift keying (BPSK) signaling, and let $\overline{\mathbf{y}} = \{y_1, y_2, y_3, \cdots, y_N\}$ be the corresponding received sequence.

A. Standard BP Decoding Algorithm

Let L_n be the log-likelihood ratio (LLR) of variable node n and initially set $L_n = (E_b/N_o) \cdot 4y_n$. Let ε^i_{mn} and z^i_{mn} be the LLRs of variable node n which is sent from check node m to variable node n, and sent from variable node n to check node m, respectively. Let z^i_n denote the a posteriori LLR of variable node n. Then the standard BP algorithm can be written as follows:

Initialization: Set $i = 1$, and the maximum number of iteration to I_{max}. For each m, n, set $z^0_{mn} = L_n$.

Step 1)

a) Horizontal step, for $1 \leq n \leq N$ and each $m \in M(n)$, process

$$r^i_{mn} = \prod_{n' \in N(m)/n} \tanh(z^{i-1}_{mn'}/2) \tag{1}$$

$$\varepsilon^i_{mn} = \log[(1 + r^i_{mn})/(1 - r^i_{mn})] \tag{2}$$

b) Vertical step, for $1 \leq n \leq N$ and each $m \in M(n)$, process

$$z^i_{mn} = L_n + \sum_{m' \in M(n)/m} \varepsilon^i_{mn} \tag{3}$$

$$z^i_n = L_n + \sum_{m' \in M(n)} \varepsilon^i_{mn} \tag{4}$$

Step 2) Hard decision and stopping criterion test.

a) Create $\widehat{\mathbf{w}}^i = [\widehat{w}^i_n]$ such that $\widehat{w}^i_n = 0$ if $z^i_n > 0$, and $\widehat{w}^i_n = 1$ otherwise.

b) If $\mathbf{H}\widehat{\mathbf{w}}^i = \mathbf{0}$ or $i = I_{max}$, stop the decoding iteration and go to Step 3). Otherwise, set $i = i + 1$ and go to Step 1).

Step 3) Output $\widehat{\mathbf{w}}^i$ as the decoded codeword.

B. Shuffled BP Decoding Algorithm

Assume that N bits of a codeword are divided into G group, and each scheduling group $S_g (1 \leq g \leq G)$ contains N_g bits such that $\sum_{g=1}^{G} N_g = N$ and $S_a \cap S_b = \emptyset$ if $a \neq b$. Then each decoding step of the shuffled BP algorithm is carried out as follows:

Initialization: Set $i = 1$, and the maximum number of iteration to I_{max}. For each m, n, set $z^0_{mn} = L_n$.

Step 1) For $1 \leq g \leq G$, process jointly the following two steps.

a) Horizontal step, for $n \in S_g$ and each $m \in M(n)$, process

$$\begin{aligned} r^i_{mn} = & \prod_{n' \in N(m)/n, n' \in \bigcup_{k<g} S_k} \tanh(z^i_{mn'}/2) \\ & \cdot \prod_{n' \in N(m)/n, n' \in \bigcup_{k \geq g} S_k} \tanh(z^{i-1}_{mn'}/2) \end{aligned} \tag{5}$$

$$\varepsilon^i_{mn} = \log[(1 + r^i_{mn})/(1 - r^i_{mn})] \tag{6}$$

b) Vertical step, for $n \in S_g$ and each $m \in M(n)$, process

$$z^i_{mn} = L_n + \sum_{m' \in M(n)/m} \varepsilon^i_{mn} \tag{7}$$

$$z^i_n = L_n + \sum_{m' \in M(n)} \varepsilon^i_{mn} \tag{8}$$

Step 2) Hard decision and stopping criterion test.

a) Create $\widehat{\mathbf{w}}^i = [\widehat{w}^i_n]$ such that $\widehat{w}^i_n = 0$ if $z^i_n > 0$, and $\widehat{w}^i_n = 1$ otherwise.

b) If $\mathbf{H}\widehat{\mathbf{w}}^i = \mathbf{0}$ or $i = I_{max}$, stop the decoding iteration and go to Step 3). Otherwise, set $i = i + 1$ and go to Step 1).

Step 3) Output $\widehat{\mathbf{w}}^i$ as the decoded codeword.

If $G = 1$, the shuffled BP decoding algorithm becomes the standard BP decoding algorithm. The layered BP decoding algorithm is similiar to the shuffled BP decoding algorithm, except that scheduling is performed based on the check nodes, instead of the variable nodes.

III. The Proposed Shuffled Scheduling Algorithm

A. The Proposed Shuffled BP Scheduling Algorithm

Consider a punctured LDPC codeword with the number of punctured variable nodes is P. Decoder sets the initial LLR of the punctured variable nodes as zero since there is no priori information about the punctured variable nodes from the channel. If some variable nodes including punctured variable nodes have zero LLR during the iterative decoding process, it cannot update the other variable nodes participate in checks in which the punctured variable nodes participate. In other words,

$$r^i_{mn} = \varepsilon^i_{mn} = 0 \text{ if } z^{i-1}_{mn'} = 0, n' \in N(m)/n \tag{9}$$

If a punctured variable node is getting nonzero LLR for the first time at ith iteration, we refer it as a punctured variable node is *recovered*, and the variable node is called $i - SR$(step-recoverable) variable node. (Or equivalently, the level of recoverability of the punctured variable node is i). From (9), we can define $i - SR$ variable node of the standard BP decoding algorithm as follows:

$$SR_{BP}(n) = i \text{ if } z^{i-1}_n = 0 \;\&\; z^{i-1}_{mn'} \neq 0 \;, n' \in N(m)/n \tag{10}$$

We can define L_{BP} as

$$L_{BP} = \max(SR_{BP}(n)), \; 1 \leq n \leq N \tag{11}$$

L_{BP} represents the number of iteration required to recover whole punctured variable nodes in the standard BP decoding algorithm, and it is varied by puncturing algorithm and P. Usually L_{BP} is incresed with P. During the iterative decoding process, if some of the punctured variable nodes are not recovered before the current decoding process, then the set of checks in which unrecovered variable node n participates cannot take part in the current decoding process for the nonzero-LLR variable nodes. Therefore, if we apply the standard BP decoding algorithm at the decoder, large L_{BP} causes huge

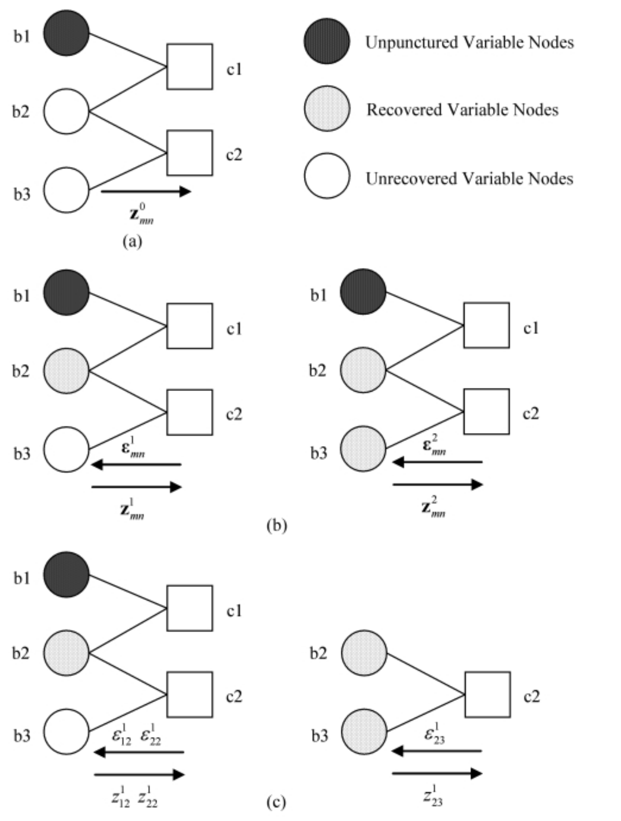


Fig. 1. The recovering process of punctured variable nodes (a) After initialization step (b) the 1st and the 2nd iteration of the standard BP algorithm (c) the decoding process of $S_1(1-SR)$ and $S_2(2-SR)$ in the proposed shuffled BP scheduling at the 1st iteration

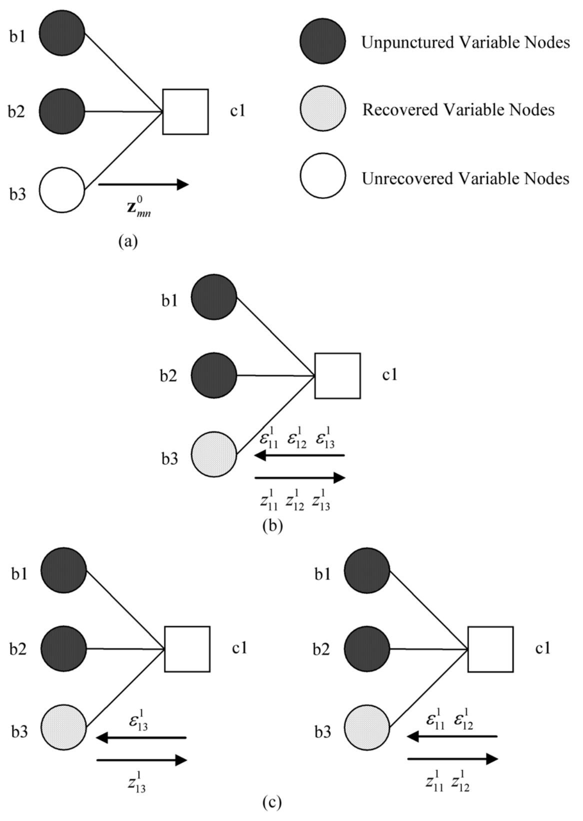

b1
b2
b3
c1
Unpunctured Variable Nodes
Recovered Variable Nodes
Unrecovered Variable Nodes
z^0_{mn}
(a)
ε^1_{11} ε^1_{12} ε^1_{13}
z^1_{11} z^1_{12} z^1_{13}
(b)
ε^1_{13}
z^1_{13}
ε^1_{11} ε^1_{12}
z^1_{11} z^1_{12}
(c)

Fig. 2. The update process of the unpunctured variable nodes during the 1st iteration (a) After initialization step (b) the layerer BP scheduling from [5] (c) the proposed shuffled BP scheduling

performance degradation, especially when the maximum number of iteration is small.

Let $SR_{SH}(n)$ be the level of recoverability of the punctured variable node n in the shuffled BP decoding algorithm. Then $SR_{SH}(n)$ can be written as follows:

$$\begin{aligned}&SR_{SH}(n) = i \text{ if } z_n^{i-1} = 0\\&\mathit{and}\ z^i_{mn'} \neq 0 \text{ for } n' \in N(m)/n, n' \in \bigcup_{k<g} S_k\\&\mathit{and}\ z^{i-1}_{mn'} \neq 0 \text{ for } n' \in N(m)/n, n' \in \bigcup_{k\geq g} S_k\end{aligned} \tag{12}$$

From (12), we can notice that the maximum $SR_{SH}(n)$, L_{SH}, can be varied not only by the puncturing algorithm and P, but also by the scheduling method. Therefore, if we apply the appropriate scheduling algorithm which considers the recovering properties of punctured variable nodes, we can reduce L_{SH} and thus accelerate the convergence of decoding. From (10) and (12), we can easily know the following two properties:

Property 1: $SR_{SH}(n) = SR_{BP}(n)$ if $SR_{BP}(n) = 0$ or 1.

Property 2: When the shuffled BP decoding algorithm is apply to the decoder, a variable node n with $SR_{BP}(n) = i$ in S_g is recoverable at the decoding process of S_g if variable node k with $1 \leq SR_{BP}(k) \leq i-1$ are already recovered.

Using these properties, we can minimize L_{SH}, the number of shuffled BP decoding iteration required to recover whole punctured variable nodes, to one, regardless of L_{BP} and P. The proposed shuffled scheduling algorithm to minimize L_{SH} to one is summarized to the follows:

$$\begin{aligned}&\text{for variable node } n\ (1 \leq n \leq N),\\&\mathit{if}\ SR_{BP}(n) = i (\neq 0),\ \mathit{assign\ to}\ S_i\\&\mathit{else\ if}\ SR_{BP}(n) = 0,\ \mathit{assign\ to}\ S_{L_{BP}+1}\end{aligned} \tag{13}$$

Fig. 1 represents the comparison of the recovering process for a punctured LDPC codeword between the standard BP decoding algorithm and the proposed shuffled BP scheduling algorithm. By its well-designed scheduling nature, the proposed shuffled BP scheduling requires only one number of iteration to recover whole punctured variable nodes in contrast to the standard BP algorithm requires L_{BP} iteration. Note that if P = 0, the proposed shuffled BP scheduling becomes the standard BP algorithm.

In the proposed shuffled BP scheduling algorithm the 0-SR (unpunctured) variable nodes are included to the last scheduling group. When we decode a received codeword using the shuffled BP decoding algorithm, the reliability of the variable nodes in S_k is increased as k is increased since more independent information is used to update the messages [3]. And, in the proposed shuffled BP scheduling, after the decoding process of $S_{L_{BP}}(i=1)$ whole punctured variable nodes are recovered. Thus, only assigning the 0-SR variable nodes to $S_{L_{BP}}+1$ can pass the messages from whole punctured variable nodes to the 0-SR variable nodes when $i = 1$.

B. Comparison with the Layered BP Scheduling [5]

In [5], scheduling algorithm using the layered BP decoding is proposed for RCP-LDPC (rate compatible punctured LDPC) codes. The layered BP scheduling algorithm in [5] sorts check nodes by related punctured variable nodes and thus accelerates the decoding convergence. The main differences between the proposed shuffled BP scheduling and the layered BP scheduling [5] can be summarized to the follows. First, the layered BP decoding algorithm is based on a serial update of check node, while the shuffled BP decoding algorithm is based on a serial update of variable nodes. Second, the layered BP scheduling observes the levels of recoverability of the punctured variable nodes at the highest considered code rate and applied to all considerable code rates, while the proposed shuffled BP scheduling focuses on currently applied puncturing.

The advantage of the proposed shuffled BP scheduling over the layered BP scheduling is in the decoding process of the 0-SR variable nodes when $i = 1$. Regardless of puncturing algorithm and P, the proposed shuffled BP scheduling always updates all $(N - P)$ 0-SR variable nodes during the 1st iteration because of bit-wise scheduling nature of the shuffled BP decoding. In contrast to the proposed shuffled BP scheduling, by its check-wise scheduling nature, the layered scheduling must have the set of check nodes in which only 0-SR variable nodes participate in order to update the 0-SR variable nodes during the 1st iteration. However, since the most puncturing algorithm is based on uniform puncturing which does not allow the overrap of check nodes as many as possible between two punctured variable nodes, the number of check in which only 0-SR variable nodes participate is rapidly decreased as P is increased. Especially, if P exceeds a certain threshold T, there exists no check node in which only 0-SR variable nodes participate. In this case, it is impossible to update the 0-SR variable nodes during 1st iteration in the layered BP scheduling. Note that T is almost same as the number of check nodes divided by the sum of degree of the punctured variable nodes.

In Fig. 2, the update processes of the unpunctured variable nodes during the 1st iteration are shown. The layered BP scheduling algorithm in Fig. 2-(b) cannot update the 0-SR variable nodes since there exist no remaining scheduling group after recovering of variable node $b3$. In contrast to the layered scheduling algorithm, the proposed shuffled scheduling algorithm can update the 0-SR variable nodes $b1$ and $b2$ after recovering of variable nodes $b3$ because of its bit-wise scheduling nature.

IV. Simulation Results

To verify the performance of the proposed shuffled BP scheduling, we consider an LDPC mother code from [6] with a code rate $R = 0.5$ and $N = 1152$. The mother code is punctured using the algorithm from [7]. Maximum number of decoding iteration is set to 8. We compare the proposed shuffled BP scheduling with the case of the standard BP algorithm, the layered BP scheduling[5], and the natural shuffled BP scheduling [3]. The number of scheduling group for each decoding scheme is described in Table 1.

Fig. 3 shows the simulation results for the average FER (frame error rate) over an AWGN channel with BPSK signaling. We can recognize that the performance gain of the proposed shuffled BP scheduling over the other schemes is increased as P is increased. The layered BP scheduling shows the best performance among the decoding schemes when r = 0.6. When r = 0.7, the layered BP scheduling still shows the best performance, but the performance gap to the proposed shuffled BP scheduling is extremely small. When r = 0.8 and 0.9, the proposed shuffled BP scheduling achieves the lowest FER and the performance gain over the layered scheduling is 0.2dB and 0.4dB, respectively. Note that if $r \neq 0.9$, the number of scheduling groups in the proposed shuffled BP scheduling is smaller than the number of scheduling groups in the layered BP scheduling and the natural shuffled BP scheduling. In Fig. 4, we compare the required E_b/N_o values of each decoding scheme for a FER of 10^{-3} when P = 512 bits and r = 0.9. Horizontal axis in Fig. 4 represents the maximum number of iteration. When the maximum number of iteration is set to 4, the performance gain of the proposed shuffled BP scheduling over the layered BP scheduling is around 1.2dB. The performance gain over the layered BP scheduling is then reduced as the maximum number of iteration is increased.

V. Conclusion

In this paper, we propose a shuffled BP scheduling algorithm for punctured LDPC codes. The proposed algorithm requires only one number of iteration to recover whole punctured variable nodes, regardless of the puncturing algorithm and the number of punctured variable nodes. In addition, the proposed shuffled BP scheduling can update the unpunctured variable nodes during the 1st iteration which may be impossible in the layered BP scheduling [5]. Simulation results show that the proposed shuffled BP scheduling outperform the standard BP decoding algorithm and the natural shuffled BP scheduling [3], and shows comparable performance to the layered BP scheduling in spite of its smaller number of scheduling group at rate below 0.9.

Acknowledgment

This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment) (IITA-2008-(C1090-0801-0011))

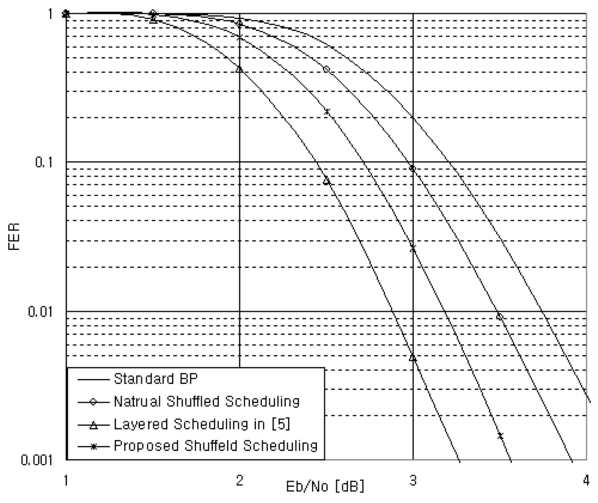
References

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

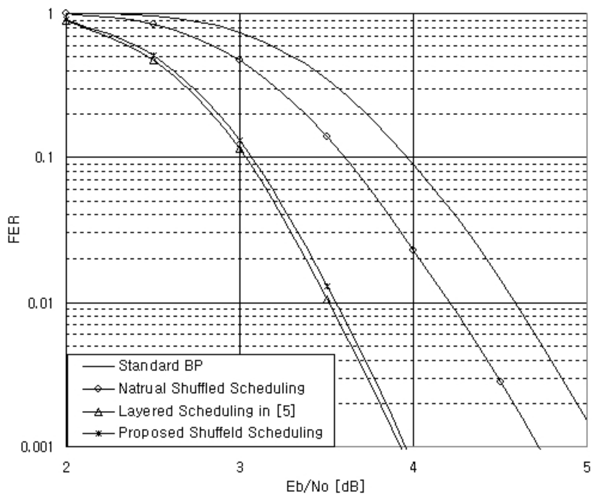
[2] D. J. C. MacKay, *"Good error-correcting codes based on very sparse matrices"*, IEEE Transactions on Information Theory, Vol. 45, pp. 399431, Mar. 1999

[3] Juntan Zhang and Marc P.C. Fossorier, *"Shuffled Iterative Decoding"* , IEEE Transactions on Communications, Vol. 53, pp. 209-213, No. 2, Feb. 2005

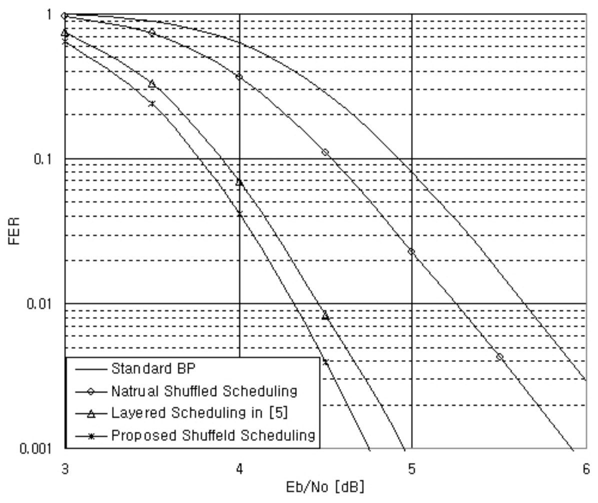
[4] D. E. Hocevar, *"A reduced complexity decoder architecture via layered decoding of LDPC codes"*, in Proc. SIPS 2004, pp. 107112.



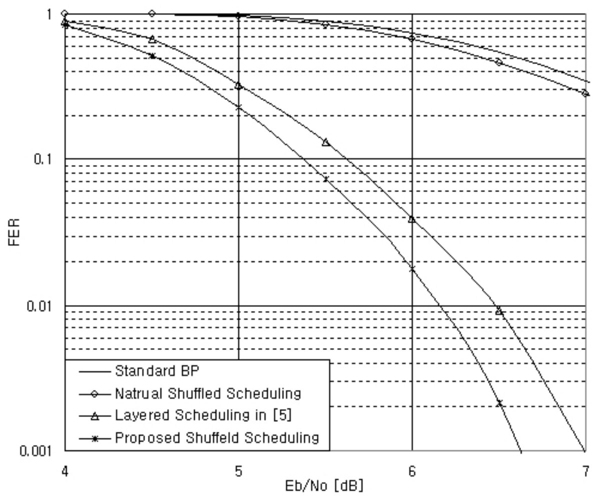
(a)



(b)



(c)



(d)

Fig. 3. FER of each decoding algorithm (a) P=192 bits (r=0.6) (b) P=329 bits (r=0.7) (c) P=432 bits (r=0.8) (d) P=512 bits (r=0.9)

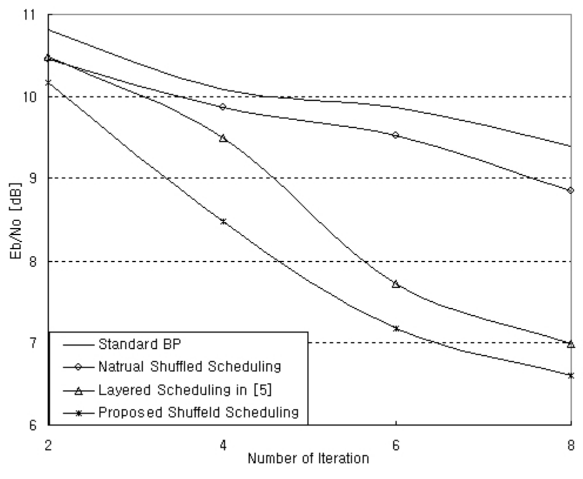


Fig. 4. The Required E_b/N_o values for a FER of 10^{-3} when P= 512 bits(r=0.9). Horizontal-axis represents the maximum number of iteration.

TABLE I
THE NUMBER OF SCHEDULING GROUP

Code Rate	Proposed Shuffled BP Scheduling	Layered BP Scheduling from [5]	Natural Shuffled BP Scheduling
0.6	2	6	7
0.7	3	6	7
0.8	3	6	7
0.9	7	6	7

[5] J.Ha, D. Klinc, J. Kwon, and Steven W. McLaughlin, *"Layered BP Decoding for Rate-Compatible Punctured LDPC Codes"*, IEEE Communication Letter, Vol. 11, pp. 440442, May. 2007

[6] IEEE Std 802.16e-2005, IEEE Standard for Local and metropolitan area networks, *"Part 16: Air interface for fixed and mobile broadband wireless access systems"*, Feb. 2006.

[7] H.Y. Park, J.W. Kang, K.S. Kim, and K.C. Whang, *"Efficient Puncturing Method for Rate-Compatible Low-Density Parity-Check Codes"*, IEEE Transactions on Wireless Communications, Vol. 6, pp. 3914-3919, Nov 2007

[8] Claude Berrou, Alain Glavieux, *"Near Optimum Error Correcting Coding and Decoding: Turbo Codes*, IEEE Transactions on Communicaions, Vol. 44, pp. 1261-1271, Oct 1996