Informed Dynamic Scheduling for Majority-Logic Decoding of Non-Binary LDPC Codes

Nanfan Qiu*, Wen Chen*, Yichao Lu† and Yang Yu*
*Department of Electronic Engineering
Shanghai Jiao Tong University, China
Email: {qiunanfan1, wenchen}@sjtu.edu.cn
†Graduate School of Information, Production and Systems, Waseda University, Fukuoka, Japan

***Abstract*—Recent studies show that dynamic scheduling using the latest available information is superior to the flooding scheduling. While the existing works on informed dynamic scheduling focus on belief propagation (BP) based algorithm for binary LDPC codes, in this paper we devise a novel method which is appropriate for majority-logic decoding of non-binary LDPC codes. We firstly clarify that previous methods are not suitable for majority-logic decoding of non-binary LDPC codes. Then we give a practical scheduling strategy which utilizes the stability of check nodes to select the messages for propagation. Furthermore, we discuss the computational complexity of this proposed scheme in detail. Simulation results verify that our approach can achieve better performance compared with layered and flooding schemes.**

***Index Terms*—LDPC; Non-binary LDPC; Informed dynamic scheduling; Majority logic decoding.**

I. Introduction

Low-Density-Parity-Check (LDPC) codes were firstly invented in early 1960s by Gallagar [1] and rediscovered by Mackay [2] in 1996. Nowadays, a great deal of research efforts have been done on non-binary LDPC codes defined over the Galois field GF(q) [3], [4], [5], for some $q > 0$. It is shown that non-binary LDPC codes have potentially better bit error rate (BER) performance at the cost of increased computational complexity.

Belief propagation (BP) algorithm is a general decoding method for LDPC codes. Flooding is the most popular scheduling strategy for BP, in which the same pre-update information is propagated to update all the check or variable nodes simultaneously. Nevertheless, recent studies have shown that sequential scheduling strategies, which propagate the latest available information, can improve decoding performance compared to the conventional flooding scheme [6]. One of the most widely used sequential scheduling is layered BP (LBP) [7], which is a sequential check node update strategy. In this paper, layered and flooding scheduling strategy are adopted for comparison purpose in simulations.

In LBP, messages are updated by a predetermined order, whereas in the informed dynamic scheduling (IDS) scheme, the message update order is calculated dynamically according to the current state of messages in the graph. Examples of IDS include residual belief propagation (RBP), node-wise residual belief propagation (NWRBP) [8], [9] and informed variable-to-check residual belief propagation (IVC-RBP) [10]. By utilizing the residual to organize the message updates, NWRBP can speed up the convergence and have better BER performance than layered scheduling. IVC-RBP outperforms NWRBP by adopting the unstable decision technique. In addition, a mixed scheduling scheme is introduced in [11] to further improve the code performance. However, strategies stated above face the problem that much extra complexity is required for residual computation. In this paper we propose a new ordering metric which requires less computational complexity.

The main obstacle to implement non-binary LDPC codes for practical applications is the high computational complexity. Although Fast Fourier Transform based q-ary sum-product algorithm (FFT-QSPA) can significantly reduce the computational complexity, the number of computations is still too large. Iterative soft-reliability-based majority-logic decoding (ISRB-MLGD) for non-binary LDPC codes is presented in [12], which requires only integer operations and computation over finite field thus achieves low computational complexity. The drawback of this algorithm is that it suffers from significant BER performance degradation compared with BP based algorithms. A double-reliability-based MLGD for non-binary LDPC codes is proposed in [13], which significantly improves the BER performance with limited complexity increasing.

To the best of our knowledge, the existing works on IDS focus on BP based decoding of binary LDPC codes, few efforts have been made on MLGD of non-binary LDPC codes. In this paper, we devise a practical IDS scheme especially for MLGD. We will explain why the previous residual based scheduling methods are not suitable for MLGD of non-binary LDPC codes.

The rest of this paper is organized as follows. Section II presents the modified ISRB MLGD algorithm for non-binary LDPC codes. Section III briefly reviews RBP for LDPC decoding and explains why the residual based scheduling scheme can not be used for MLGD. Then we introduce and justify the proposed algorithm in detail. Section IV analyzes the complexity of the proposed IDS mehtod in one iteration. The simulation results which verify the effectiveness of our proposed methods are discussed in Section V. Finally, Section VI draws a conclusion.

This work is supported by the National 973 Project #2012CB316106, by NSF China #61161130529, and by the National 973 Project #2009CB824904.

978-1-4799-1353-4/13/$31.00 ©2013 IEEE

II. A Modified Iterative Soft-Reliability-Based MLGD Algorithm

In this section, a modified ISRB MLGD algorithm will be described in detail. For a review of MLGD for non-binary LDPC codes, one can refer to [12]. The most important feature of ISRB is simplicity. This algorithm only requires integer operations and computations over finite field. The distinction of our algorithm from [12] is as follows: ISRB calculates the reliability measure of variable nodes and check nodes in the initialization. So during each iteration, the reliability measure of each check node is a constant. In each iteration, every check node makes a contribution in predicting which element in Galois field should the connected variable node be decoded into. This prediction changes dynamically during each iteration. In contrast, in the modified algorithm, we update the variable-to-check messages iteratively according to the reliability measure of variable node. By this method, not only the prediction for variable node, but also the reliability measure of variable node and check node are changed dynamically, which improves the decoding performance and makes this algorithm more suitable for IDS as well.

For simplicity, we present the decoding algorithm on a (γ,ρ)-regular LDPC code. Consider a non-binary LDPC code $\mathcal{C}$ defined by a regular $M \times N$ parity-check matrix $\mathbf{H}$ with column weight γ and row weight ρ. Let GF(q) denote a Galois field of size q, where $q = 2^r$. Each entry of $\mathbf{H}$ are taken from GF(q). For a transmitted codeword $\mathbf{c} = (c_0, c_1, \ldots, c_{n-1}) \in \mathcal{C}$, we expand each code symbol into an r-tuple over GF(2). Then a sequence of nr bits are transmitted over a binary-input Additive White Gaussian Noise (AWGN) channel with two-sided power spectral density $\frac{N_0}{2}$. The Binary Phase Shift Keying (BPSK) modulation is adopted, with modulation mapping $x \mapsto 1 - 2x$. Let $\mathbf{y} = (\mathbf{y_0}, \mathbf{y_1}, \ldots, \mathbf{y_{n-1}})$ denote the received sequence, where each $\mathbf{y_j} = (y_{j,0}, y_{j,1}, \ldots, y_{j,r-1})$ is r tuple. For $0 \le k < r$, we quantize each received bit $y_{j,k}$ into $2^\omega - 1$ intervals, where $q_{j,k}$ is the quantized value represented by ω bits with range in $[-(2^\omega - 1), +(2^\omega - 1)]$. Let $(a_{l,0}, a_{l,1}, \ldots, a_{l,r-1})$ denote the binary representation of $a_l \in GF(q)$. For $0 \le j < n$ and $a_l \in GF(q)$, the initialized reliability measure can be computed by

$$R_{j,l}^{(0)} = \sum_{t=0}^{r-1} [(1 - 2a_{l,t}) q_{j,t} + q_{max}], \tag{1}$$

where $a_{l,t}$ is the t-th bit of binary representation of a_l, $q_{max} = 2^\omega$ makes sure that the reliability measure is always larger than 0. By this method, division between reliability measure can make sense which will be introduced in the next section and $R_{j,l}^0$ denotes the reliability measure from channel which gives belief in the j-th symbol $(q_{j,0}, q_{j,1}, \ldots, q_{j,r-1})$ that should be decoded into a_l.

To make the description of this algorithm more clear, we introduce the following notations. $N(c_i)$ denotes the set of variable nodes connected to check node i. $N(v_j)$ denotes the set of check nodes connected to variable node j. $Imax$ denotes the maximum iteration number. $\mathbf{z}^k = (z_0^k, z_1^k, \cdots, z_{n-1}^k)$ represents the decoding result in the k-th iteration. $\mathbf{s} = (s_0, s_1, \cdots, s_{m-1})$ stands for check-sum vector. For $0 \le j < n$, $\mathbf{R}_j^{(k)} = (R_{j,0}^{(k)}, R_{j,1}^{(k)}, \ldots, R_{j,r-1}^{(k)})$ denotes the reliability measure vector for z_j and $R_{j,l}$ is the measure of the reliability that z_j is decoded to the l-th element in GF(q). Let τ_j denote the maximum reliability value in vector $\mathbf{R}_j$ and $\psi_{i,j}$ denote the reliability measure of the voting by check node i. Then the modified ISRB MLGD algorithm is described in Algorithm 1.

Algorithm 1 A Modified Iterative Soft-Reliability-Based Algorithm

1: **Initialization**: $\mathbf{z}^0 = \mathbf{z}$, compute $\mathbf{R}^0$ using (1).
2: **for** $k = 1$ to $Imax$ **do**
3:   **for** $i = 0$ to m-1 **do**
4:     $s_i = \sum\limits_{j \in N(c_i)} h_{i,j} z_j^k$.
5:   **end for**
6:   Stopping criterion test.
7:   **for** $i = 0$ to m-1 **do**
8:     **for** j$\in N(c_i)$ **do**
9:       $\widetilde{s}_{i,j} = (h_{i,j})^{-1} \cdot s_i - z_j^k$.
10:     **end for**
11:   **end for**
12:   **for** $j = 0$ to n-1 **do**
13:     $\tau_j^k = \lambda \left(\underset{l \in GF(q)}{max} R_{j,l}^k \right)$.
14:   **end for**
15:   **for** $i = 0$ to m-1 **do**
16:     **for** j$\in N(c_i)$ **do**
17:       $\psi_{i,j}^k = \underset{j' \in N(c_i) \backslash j}{min} \tau_{j'}^k$.
18:     **end for**
19:   **end for**
20:   **for** $i = 0$ to m-1 **do**
21:     **for** j$\in N(c_i)$ **do**
22:       $R_{j,\widetilde{s}_{i,j}}^{k+1} = R_{j,\widetilde{s}_{i,j}}^k + \psi_{i,j}^k$.
23:     **end for**
24:   **end for**
25:   **Tentative Decoding** :
26:   **for** $j = 0$ to n-1 **do**
27:     $z_j^{k+1} = \text{GFmax}(\mathbf{R_j^k})$.
28:   **end for**
29: **end for**

In Algorithm 1, $\widetilde{s}_{i,j}$ computed in line 9 denotes that check node i votes variable node j decoded into $\widetilde{s}_{i,j}$. The reliability measure of this voting is given by $\psi_{i,j}^k$ in line 17. τ_j in line 13 denotes the maximum reliability measure of variable node j, which will be used to compute the reliability for neighboring check nodes. The parameter λ is the scaling factor which needs to be carefully choosed for optimizing the error correction performance. The optimal value of λ depends on the code structure and signal-to-noise ratio (SNR). Here we keep λ constant for simplicity and it is determined by simulation. Line 22 adds all the votes from check node i to the indicated element of variable node j. Line 27 is the tentative decoding according to the reliability measure.

III. Informed Dynamic Scheduling

In this section, we will introduce the informed dynamic scheduling scheme specifically designed for ISRB MLGD algorithm of non-binary LDPC codes. At first we will summarize the previous works and analyze why these algorithms can not be directly applied to MLGD. Previous works such as RBP, node-wise RBP in [8] and [9], IVC RBP in [10], are residual based scheduling for belief propagation of binary LDPC codes. These algorithms firstly update the message with the largest residual. A residual is the absolute value of the difference between the message before and after an update. For a message m, the residual is defined as

$$r(m) = \|m^{new} - m^{old}\|, \tag{2}$$

where m^{new} denotes message after an update and m^{old} denotes message before an update. As BP converges, the residual will gradually tend to zero. A message with large residual means this part of Tanner graph has not converged yet. Therefore, the intuitive method is to propagate this message first which will speed up the convergence. RBP is a greedy algorithm, which has a higher convergence speed but gets the correct result less often. To solve this problem, a less greedy node-wise RBP is also presented in [9]. Instead of only propagating the message with the largest residual, node-wise RBP update and propagate the messages that correspond to the same check node at the same time.

All these existing works are based on the common assumption that as decoding converges, the difference between messages before and after an update will gradually tend to zero. Therefore, the ordering metric which decides the propagating sequence is determined by the residual. In the case of MLGD for non-binary LDPC codes, the reliability of variable node accumulates as the decoding converges, and the reliability measure sent from check node to variable node is calculated by the minimum neighboring variable node's reliability. So neither residual of variable node message nor residual of check node message makes sense in MLGD of non-binary LDPC codes.

In our proposed IDS scheme, we define a new parameter to determine the ordering metric. We observed that if the maximum reliability of variable node is much larger than the second largest reliability in the reliability vector, then this variable node has a large probability to stay stable in later iterations. On the contrary, if the maximum reliability is comparable with the second largest reliability, this variable node would probably be decoded into the element which holds the second largest reliability. Based on this observation, we propose a parameter "stability" defined by the following equation

$$S_{v_j} = \frac{\underset{l \in GF(q)}{max} R_{j,l}}{\underset{l' \in GF(q) \backslash l}{max} R_{j,l'}}. \tag{3}$$

The stability of check node is measured by the minimum stability value in the neighboring variable nodes. It can be calculated using the following equation:

$$S_{c_i} = \underset{j \in N(c_i)}{min} S_{v_j}. \tag{4}$$

For each check node, we use a set "$Vote$" to record it is visited or not. If a check node c_i is visited, update $Vote(c_i) = 0$. After all the check nodes are visited in one iteration, we update all the entries in $Vote$ to 1 for next iteration.

The proposed IDS for MLGD based non-binary LDPC in pseudo-codes is given in algorithm 2:

Algorithm 2 IDS for ISRB MLGD of non-binary LDPC codes

1: Initialize $\mathbf{z}^0 = \mathbf{z}$, compute $\mathbf{R}^0$.
2: Initialize all $S(v_j)$ using equation (3), all $S(c_i)$ using equation (4), all $Vote(c_i) = 1$.
3: **if** there are unsatisfied check nodes with $Vote(c_i) = 1$ **then**
4: Find the largest $S(c_i)$ with unsatisfied check node c_i.
5: **else**
6: Find the largest $S(c_i)$ with $Vote(c_i) = 1$.
7: **end if**
8: Update the set $Vote$.
9: **for** every $v_j \in N(c_i)$ **do**
10: generate and propagate $m_{c_i \to v_j}$.
11: update variable node stability using $S(v_j) = \frac{\underset{l \in GF(q)}{max} R_{j,l}}{\underset{l' \in GF(q) \backslash l}{max} Rj,l'}$.
12: tentative decoding $z_j = GFmax(\mathbf{R_j})$.
13: **for** every $c_a \in N(v_j) \backslash c_i$ **do**
14: compute check sum for c_a.
15: update check node stability using $S(c_a) = \underset{k \in N(c_a)}{min} S(v_k)$.
16: **end for**
17: **end for**
18: **if** Stopping rule is not satisfied **then**
19: Go to line 3;
20: **end if**

To make it clear, we define the procedure from line 3-17 in the proposed IDS algorithm as an update. When all the check nodes have been updated, we call this an IDS iteration. Without loss of generality, we assume that c_i with the largest stability is selected to be updated. For message update, firstly message from c_i to v_j is generated and propagated to update all the neighboring nodes of c_i as line 10 indicates. Secondly we calculate the stability of each updated variable node and make a tentative decoding. Then for each neighboring check node $c_a \in N(v_j) \backslash c_i$, new check sum will be calculated, and the stability of check node which is the ordering metric will also be updated, as line 14-15 indicate. Finally we check whether the stopping rule is satisfied. Similar to the node-wise RBP, our proposed algorithm is also a less greedy algorithm which updates and propagates messages corresponding to the same check node at the same time.

Figure.1 shows an example to explain how our algorithm works. Without loss of generality, we assume v_1, v_2 are erroneous variable nodes. v_3, v_4, v_5 are correct variable nodes. The stability of variable nodes satisfy that $S(v_1) < S(v_2) < S(v_3) < S(v_4) < S(v_5)$. Since v_1, v_2 contain errors in them, c_2 is certainly an unsatisfied check node. c_1 may also be an unsatisfied check node with a large probability. c_3 is a satisfied check node. According to equation (4), we have $S(c_1) = S(v_1), S(c_2) = S(v_2), S(c_3) = S(v_3)$. In layered

TABLE I
COMPLEXITY IN ONE ITERATION COMPARISON

Methods	Update No.	For Ordering Metric			
Layered	Md_c	0			
Flooding	Md_c	0			
NW RBP	Md_c	Real Mul		Special Operations	
		$2Md_c^2(d_v-1)(d_c-1)$		$Md_c^2(d_v-1)(d_c-1)(\tanh+\tanh^{-1})$	
Our Method	Md_c	GF addition	GF Mul	Real comparison	Real Div
		$Md_c(d_v-1)(d_c-1)$	$Md_c(d_v-1)d_c$	$Md_c(2q-3)+Md_c(d_v-1)(d_c-1)$	Md_c

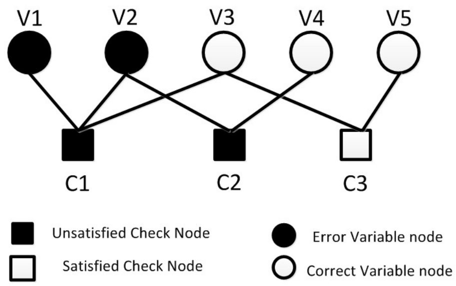

Fig. 1. Example of proposed method.

scheduling, the update order is c_1, c_2, c_3. First we propagate the messages from c_1 to v_1, which are generated using v_2 and v_3. As v_2 is erroneous, this propagation will reinforce the error in v_1. In our proposed method, as c_2 is the unsatisfied check node which holds the largest stability, it will be selected first, then v_2 can be corrected. c_1 is selected as the next check node after v_2 is corrected. Therefore messages pass from c_1 to v_1 which is generated using v_2 and v_3 will indicate the correct element in GF(q). v_1 will probably be corrected by this update. Finally c_3 is selected and propagate messages to strengthen the reliability of neighboring variable nodes. Considering the most troublesome trapping set problem where only degree-1 and degree-2 check nodes exist in the sub-graph [9], [10]. Our method focuses on the unsatisfied check nodes with maximum stability first. In the trapping set, degree-1 check nodes are certainly unsatisfied, while degree-2 check nodes may be satisfied according to the error and parity check matrix. It's likely that degree-1 check nodes will be updated first, then the connected variable node can be corrected. Therefore, degree-2 check nodes become degree-1 in the subgraph. Then another variable node will be corrected. So this method can solve some trapping set problems which can not be solved by layered scheduling.

IV. Complexity Analysis

In this section, we will analyze the computational complexity of the proposed IDS algorithm in detail.

Let M denote the number of rows in parity check matrix H, d_c be the check node degree and d_v be the variable node degree. From algorithm 2 we can observe that selected check node will propagate d_c messages to neighboring variable nodes. Thus in an iteration there are Md_c message generations and propagations. The complexity analysis is given by Table I. The NWRBP algorithm [9] needs to calculate the check-to-variable messages to be propagated in the next update step. However, only part of the messages corresponding to the selected check node will be used for update while the other messages are just for residual computation. As we know, check node update is the most complicated part in BP algorithm. Therefore we can reduce the complexity by only calculating the stability of variable and check nodes in the current update. To justify this, we compare our algorithm with NWRBP style scheduling in Table I.

In one iteration, number of check-to-variable messages for update is the same among these four algorithms. The extra complexity of IDS scheduling is dominated by the computations only for ordering metric. In Node-wise RBP, we have to calculate the extra check-to-variable messages, which is the most complicated part in BP algorithm. In our proposed algorithm, additional complexity for ordering metric calculation corresponds to line 14 and 15, which will be executed by $Md_c(d_v-1)$ times in one iteration. For calculating one check sum, d_c-1 GF addition and d_c GF multiplication are needed. In one iteration, Md_c variable node's stability should be calculated, requiring Md_c real divisions according to equation (3). Complexity of real comparison operation can be divided into two parts. The first item $Md_c(2q-3)$ comes from finding the maximum reliability and the second maximum reliability of Md_c variable nodes in equation (3). The second item $Md_c(d_v-1)(d_c-1)$ comes from calculating the stability of check node in equation (4), where $Md_c(d_v-1)$ is the execution number, d_c-1 is the complexity to find the minimum value from d_c values. From Table I, we can easily observe that our method reduces the computational complexity compared with NWRBP. The complexity of NWRBP is $O(Md_c^3d_v)$ for real multiplication and special operations which require high computational complexity. In contrast, our algorithm requires $O(Md_c^2d_v)$ GF operation and real multiplication, which have low computational complexity. A small number of real division, i.e. Md_c, is used in our proposed algorithm. Therefore, our algorithm reduce the complexity to calculate the ordering metric.

V. Simulation Results

In this section we will show the error correction performance of different scheduling strategies over AWGN channel. In the simulations we use the same rate-1/2 non-binary LDPC code with block length 1008 over GF(8). Four different algorithms will be used in our simulations, including the ISRB [12], the modified ISRB, the layered schduling for modified ISRB and the proposed IDS for modified ISRB, where the ISRB and modified ISRB both employ flooding scheduling.

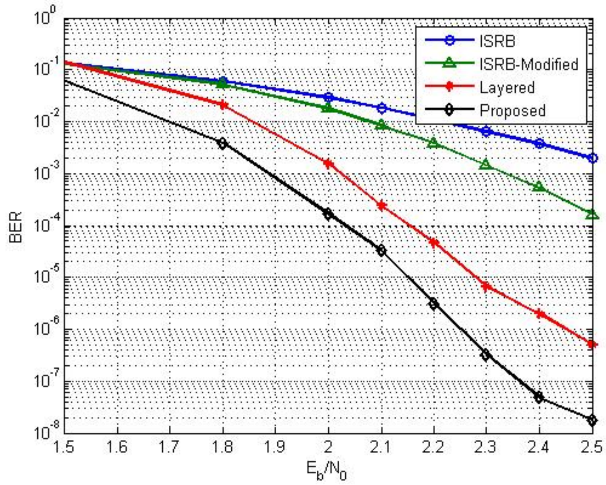

Fig. 2. BER performance of ISRB, Modified ISRB, Layered Scheduling and proposed IDS scheduling for at most 8 iterations with non-binary LDPC code over GF(8) of blocklength-1008 rate-$1/2$.

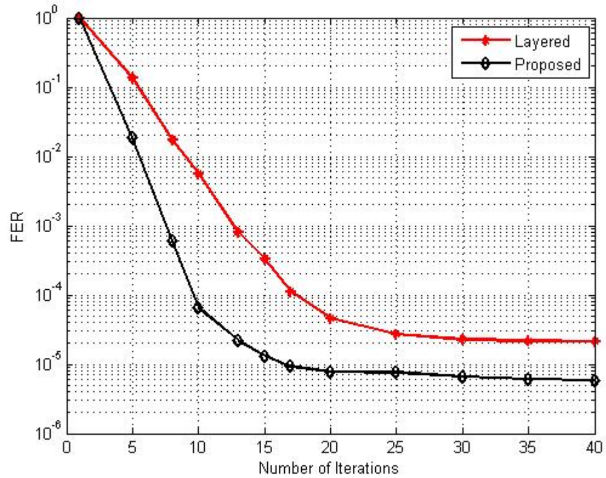

Fig. 3. FER performance vs the number of iterations for non-binary LDPC codes over GF(8) with blocklength-1008 rate-0.5 over AWGN channel using layered scheduling and proposed IDS scheduling at a fixed $E_b/N_0 = 2.3dB$.

The BER of different algorithms mentioned above for 8 iterations is presented in Fig. 2. We can see that the modified ISRB outperforms ISRB. The reason is that modified ISRB updates reliability of variable and check nodes iteratively. In addition, modified ISRB with layered scheduling improves the BER performance which meets the empirical results because messages are sequentially updated. Furthermore, our proposed IDS strategy outperforms layered method by about 0.22 dB at the BER of 5×10^{-7}. The excellent performance of our proposed IDS scheduling justifies the effectiveness of the proposed ordering metric.

Fig. 3 shows the frame error rate (FER) performance of different scheduling strategies as the number of iterations increase. It is shown that our proposed method can achieve a better performance in a smaller number of iterations than layered scheduling strategy. Layered strategy needs nearly 27 iterations for convergence while our proposed IDS method needs only 17 iterations for convergence. The fast convergence speed is mainly attributed to that we locate the unsatisfied check sum in each iteration.

VI. Conclusions

In this paper we propose a new informed dynamic scheduling algorithm specifically designed for ISRB MLGD of non-binary LDPC codes. A new ordering metric is proposed in this paper to describe the stability information of variable and check nodes. In addition, unsatisfied check sum which locates the error variable nodes is utilized to speed up the convergence. Furthermore, we ensure that every check node votes once in each iteration for fairness. These three criterions dynamically decide which check node to be selected in each update and greatly attribute to the excellent performance of proposed scheduling method. We also discuss the complexity of proposed method in detail, and verify that the proposed IDS costs less computational complexity compared with NWRBP. Furthermore, clarified by the analysis and simulation results, we show that the proposed IDS scheduling scheme can achieve better BER performance and accelerate convergence speed compared with layered scheduling methods for non-binary LDPC codes. Therefore, this method is appropriate for high-speed applications and low-power applications.

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 Trans. Information Theory*, vol. 45, no. 2, pp. 399-431, March 1999.

[3] D. Declercq and M. Fossorier, “Decoding algorithms for nonbinary LDPC codes over GF(q),” *IEEE Trans. Commun.*, vol. 55, no. 4, pp.633 -643, 2007.

[4] Y. Yu, and Wen Chen, “Design of Low Complexity Nonbinary LDPC Codes with an Approximated Performance-Complexity Tradeoff,” *IEEE Communications Letters*, vol. 16, no. 4, pp. 514-517, 2012.

[5] Y. Yu, Wen Chen, and L. Wei, “Design of Convergence-Optimized Non-binary LDPC Codes over Binary Erasure Channel,” *IEEE Wireless Communications Letters*, vol. 1, no. 4, pp. 336-339, 2012.

[6] P. Radosavljevic , A. de Baynast and J. R. Cavallaro, “Optimized message passing schedules for LDPC decoding,” *Proc. 39th Asilomar Conf. Signals, Syst. Comput.*, pp.591 -595,2005.

[7] D. Hocevar, “A reduced complexity decoder architechture via layered decoding of LDPC codes,” *Proc. Signal Process. Syst. SIPS*, pp.107 -112 2004.

[8] A. Casado, M. Griot and R. D. Wesel, “Informed dynamic scheduling for belief-propagation decoding of LDPC codes,” *IEEE International Conference on Communications*, pp.932 -937, June 2007.

[9] A. Casado, M. Griot and R. D. Wesel, “LDPC decoders with informed dynamic scheduling,” *IEEE Trans. Commun.*, vol. 58, no. 12, pp.3470 -3479, Dec. 2010.

[10] Y. Gong, X. Liu, W. Ye, and G. Han, “Effective informed dynamic scheduling for belief propagation decoding of LDPC codes,” *IEEE Trans. Commun.*, vol. 59, no. 10, pp.2683 -2691, Oct. 2011.

[11] K. Saejoon, “Trapping Set Error Correction through Adaptive Informed Dynamic Scheduling Decoding of LDPC Codes,” *IEEE Commun. Lett.*, vol. 16, no. 7, pp.1103 -1105, July. 2012.

[12] C.-Y. Chen, Q. Huang, C.-C. Chao and S. Lin, “Two low-complexoty reliability-based message-passing algorithms for decoding non-binary LDPC codes,” *IEEE Trans. Commun.*, vol. 58, no. 11, pp.3140 -3147 , Nov. 2010.

[13] Y. Lu, N. Qiu, Z. Chen, S. Goto, “An efficient majority-logic based message-passing algorithm for non-binary LDPC decoding,” *IEEE Asia Pacific Conference on Circuits and Systems* , pp. 479- 482 , Dec. 2012.