Joint Shuffled Scheduling Decoding Algorithm for DP-LDPC Codes-Based JSCC Systems

Zhiping Xu, Lin Wang, *Senior Member, IEEE*, Shaohua Hong, Francis C. M. Lau, *Senior Member, IEEE* and Chiu-Wing Sham, *Member, IEEE*

***Abstract*—In this letter, a joint shuffled scheduling decoding algorithm for double-protograph low-density parity-check (DP-LDPC) codes-based joint source-channel coding (JSCC) schemes is presented. The proposed algorithm adopts the shuffled scheduling method to both source decoder and channel decoder, and can improve convergence speed and reduce the decoding complexity. In addition, a unique phenomenon named unequal convergence rates is revealed at the same time. The proposed algorithm can terminate this phenomenon with faster decoding. The results show that this algorithm has faster convergence speed and better error performance, compared with the traditional joint belief propagation (BP) decoding algorithm.**

***Index Terms*—JSCC, DP-LDPC codes, belief propagation, shuffled scheduling**

I. Introduction

Recently, a joint source-channel coding (JSCC) system based on double-protograph low-density parity-check (DP-LDPC) codes [1] has drawn significant interest because of its low-power and low-cost properties. DP-LDPC systems adopt protograph low-density parity-check (P-LDPC) codes as both source code and channel code. This type of codes has the advantages of fast encoding and linear decoding implementation. Compared with double low-density parity-check (D-LDPC) JSCC [2], this system achieves better error performance in the waterfall region and the error-floor region.

Several optimized schemes have been proposed for the DP-LDPC systems. In [3] and [4], the optimized source code and the re-designed channel code have been used, respectively, to improve the error performance in the error-floor region and the waterfall region. The source code and the channel code are also jointly optimized in [5]. Optimizing the edge connections in this system has been studied for enhancing its performance in [6]. With unequal power allocation, the system can obtain better performance by utilizing the source statistics at the decoder [7]. But there is a lack of research on the decoding algorithms though it is very important for the practical usage of this system. At present, the decoding algorithm used in this system is the joint belief-propagation (BP), which has a disadvantage of high decoding complexity.

As we all know, the complexity of decoding algorithms has a great impact on power consumption. The decoding complexity of LDPC codes can be roughly measured by the number of edges and the maximum number of iterations [8]. The former is determined and fixed by the code design. Thus, reducing the number of iterations will be an effective method to reduce complexity, and can further reduce the power dissipation to adapt to low-power practical application scenarios.

The scheduling algorithms which control the order that messages are passed along the edges of a Tanner graph affect the convergence speed of the decoding process. The flooding schedule [9] is the standard scheduling algorithm. In the DP-LDPC system, the joint BP algorithm is also called joint flooding scheduling decoding algorithm. The flooding schedule is a kind of parallel scheduling algorithm, which is very practical though its convergence rate is not high. On the other hand, it has been proved that serial scheduling algorithms converge twice as fast as the flooding scheduling algorithms in a channel coding system [13]. Broadly speaking, there are two types of serial scheduling algorithms: Variable-Nodes-Based (VNB) algorithms [8], [10], [11] and Check-Nodes-Based (CNB) algorithms [12], which are also known as shuffled and layered scheduling decoding algorithms. However, the effect of applying serial scheduling algorithms to DP-LDPC systems is unknown.

In this letter, we will focus on the convergence of the decoding algorithms for DP-LDPC systems. The main contributions of this paper are as follows.

1) This work provides a hardware-friendly decoding algorithm for DP-LDPC systems, which is different from joint BP.

2) The joint shuffled scheduling decoding algorithm is designed for the DP-LDPC system to improve convergence speed and reduce decoding complexity, which also can terminate unequal convergence rates phenomenon at the same time.

In Section II, the DP-LDPC system model is briefly reviewed. The joint shuffled decoding algorithm is detailed in Section III. Simulation results are presented and the complexity is analyzed in Section IV. Finally, conclusions are drawn in Section V.

II. DP-LDPC System Model

The system model and the corresponding joint decoder are shown in Fig. 1 and Fig. 2, respectively. Let $\mathbf{s} = (s_1, s_2, \ldots)$

This work was supported by the National Natural Science Foundation of China (Grant Nos. 61671395) and Natural Science Foundation (No. 2018A030313710).

Z. Xu and L. Wang are with the Department of Communication Engineering, Xiamen University, Xiamen, 361005, P. R. China (e-mail: xzpxmu@gmail.com, wanglin@xmu.edu.cn).

S. Hong is with Shenzhen Research Institute of Xiamen University, Shenzhen, 518057, P. R. China (e-mail: hongsh@xmu.edu.cn).

Francis C. M. Lau is with the Department of Electronic and Information Engineering, Hong Kong Polytechnic University, Kowloon, Hong Kong (email: francis-cm.lau@polyu.edu.hk).

Chiu-Wing Sham is with the Department of Computer Science, The University of Auckland, New Zealand (e-mail: b.sham@auckland.ac.nz).

Fig. 1. A DP-LDPC system model.

Fig. 2. Joint decoder of the DP-LDPC system.

be a source sequence ($s_i \in \{0,1\}$) generated by a binary independent and identically distributed (i.i.d.) Bernoulli source with entropy $H(\mathbf{s}) = -p\log_2(p) - (1-p)\log_2(1-p)$, where $p = \Pr(s_i = 1)$ and $p \neq 0.5$. The source sequence $\mathbf{s}$ is compressed into a sequence $\mathbf{b}$ by a P-LDPC code with rate $R_{sc} = M_{sc}/N_{sc}$. The compressed sequence is then protected by another P-LDPC code of rate $R_{cc} = M_{cc}/N_{cc}$ to realize a reliable transmission. Finally, the sequence $\mathbf{c}$ produced by the channel code is modulated by binary-phase-shift keying (i.e., bit $0/1$ mapped to $+1/-1$) and transmitted over an additive white Gaussian noise (AWGN) channel as $\mathbf{x}$. The received signal $\mathbf{y}$ is a corrupted version of $\mathbf{x}$.

As shown in Fig. 2, the joint decoder runs in parallel via a joint Tanner graph taking advantages of the BP algorithm and the source statistics. The edges between the two decoders connect the variable nodes (VNs) of the channel code (left) to the check nodes (CNs) of the source code (right), and are used to exchange extrinsic messages between the two decoders.

Through observing the iterative decoding process of the DP-LDPC system, we find that the channel decoder and the source decoder do not converge at the same rate. As shown in Fig. 3, the average numbers of iterations required for convergence by these two decoders in the joint BP algorithm are not the same at a given E_b/N_0. This phenomenon, which we call unequal convergence rates (UCR) and is unique to this system, leads to higher decoding complexity and power consumption.

III. Joint Shuffled Decoding Algorithm

Based on [2], [10], we consider the i-th iteration of the joint decoder. For ease of exposition, six types of log-likelihood ratios (LLRs) are defined as follows and are shown in Fig. 2.

- $\varepsilon_{mn}^{cc,(i)}$ represents the LLR sent from the m-th CN to the n-th VN and $z_{mn}^{cc,(i)}$ represents the LLR sent from the n-th VN to the m-th CN in the channel decoder.
- $\varepsilon_{mn}^{sc,(i)}$ represents the LLR sent from the m-th CN to the n-th VN and $z_{mn}^{cc,(i)}$ represents the LLR sent from the n-th VN to the m-th CN in the source decoder.
- $\ell_n^{sc\to cc,(i)}$ is the LLR sent from the corresponding CN in source decoder to the n-th VN in channel decoder, and $\ell_n^{cc\to sc,(i)}$ is the LLR sent from the n-th VN in channel decoder to the corresponding CN in source decoder. These two types of LLRs are indexed only by n because each CN in the source decoder is connected to only a single VN in the channel decoder.

We denote $M(n)$ as all CNs connected to the n-th VN, and $M(n)/m$ as all CNs connected to the n-th VN excluding the m-th CN. Similarly, we denote $N(m)$ as all VNs connected to the m-th CN, and $N(m)/n$ as all VNs connected to the m-th CN excluding the n-th VN. We also use F_n^{cc} $(n = 1,\ldots,N_{cc})$ to represent the channel LLR of the n-th VN in the channel decoder and F_n^{sc} $(n = N_{cc}+1,\ldots,N_{cc}+N_{sc})$ to represent the source LLR of the n-th VN in the source decoder. We denote the parity check matrices of the channel protograph and source protograph as $\mathbf{H}_{cc}$ and $\mathbf{H}_{sc}$, respectively. We set the maximum number of decoding iterations to $I_{\max}$ and assume an AWGN channel with zero mean and variance σ^2.

Based on the above definitions, the proposed algorithm is described as follows.

Initialization:

For all m and n, set $\varepsilon_{mn}^{sc,(0)} = 0, \varepsilon_{mn}^{cc,(0)} = 0, \ell_n^{sc\to cc,(0)} = 0$. For $n = 1,\ldots,N_{cc}$, set $z_{mn}^{cc,(0)} = F_n^{cc} = 2y_n/\sigma^2$ where $y_n = (1-2x_n) + G_n$ and $G_n \sim N(0,\sigma^2)$. For $n = N_{cc}+1,\ldots,N_{cc}+N_{sc}$, set $z_{mn}^{sc,(0)} = F_n^{sc} = \ln((1-p))/p)$. Set $i=1$.

Step 1: LLRs updating.

Channel decoder:

1) For $1 \le n \le N_{cc}$, and each $m \in M(n)$, process the next two steps jointly.

1.1) Horizontal Step: Compute $\varepsilon_{mn}^{cc,(i)}$ using

$$\tanh(\frac{\varepsilon_{mn}^{cc,(i)}}{2}) = \prod_{\substack{n'\in N(m)/n \\ n'<n}} \tanh(\frac{z_{mn'}^{cc,(i)}}{2}) \times \prod_{\substack{n'\in N(m)/n \\ n'>n}} \tanh(\frac{z_{mn'}^{cc,(i-1)}}{2}). \quad (1)$$

1.2) Vertical Step: When $1 \le n \le N_{cc} - M_{sc}$, compute

$$z_{mn}^{cc,(i)} = F_n^{cc} + \sum_{m'\in M(n)/m} \varepsilon_{m'n}^{cc,(i)}. \quad (2)$$

When $N_{cc} - M_{sc} + 1 \le n \le N_{cc}$, compute

$$z_{mn}^{cc,(i)} = F_n^{cc} + \ell_n^{sc\to cc,(i-1)} + \sum_{m'\in M(n)/m} \varepsilon_{m'n}^{cc,(i)}. \quad (3)$$

2) For $N_{cc} - M_{sc} + 1 \le n \le N_{cc}$ and each $m \in M(n)$, compute

$$\ell_n^{cc\to sc,(i)} = F_n^{cc} + \sum_{m\in M(n)} \varepsilon_{mn}^{cc,(i)}. \quad (4)$$

Source decoder:

1) For $N_{cc}+1 \leq n \leq N_{cc}+N_{sc}$ and each $m \in M(n)$, process the next two steps jointly.

1.1) Horizontal Step: Compute $\varepsilon_{mn}^{sc,(i)}$ using

$$\tanh(\frac{\varepsilon_{mn}^{sc,(i)}}{2}) = \tanh(\frac{\ell_n^{cc\rightarrow sc,(i)}}{2})\times \prod_{\substack{n'\in N(m)/n \\ n'<n}} \tanh(\frac{z_{mn'}^{sc,(i)}}{2}) \times \prod_{\substack{n'\in N(m)/n \\ n'>n}} \tanh(\frac{z_{mn'}^{sc,(i-1)}}{2}). \quad (5)$$

1.2) Vertical Step: Compute

$$z_{mn}^{sc,(i)} = F_n^{sc} + \sum_{m'\in M(n)/m} \varepsilon_{m'n}^{sc,(i)}. \quad (6)$$

2) For $1+N_{cc} \leq n \leq N_{sc}+N_{cc}$ and each $m \in M(n)$, compute $\ell_n^{sc\rightarrow cc,(i)}$ using

$$\tanh(\frac{\ell_n^{sc\rightarrow cc,(i)}}{2}) = \prod_{n'\in N(m)} \tanh(\frac{z_{mn'}^{sc,(i)}}{2}). \quad (7)$$

Step 2: Hard decision.

Compute the *a posteriori* LLRs $z_n^{cc,(i)}$ of the VNs in the channel decoder using (8) for $n=1,...,N_{cc}-M_{sc}$ and (9) for $n=N_{cc}-M_{sc}+1,...,N_{cc}$. Set $\hat{\mathbf{c}}=\{\hat{c}_n^{(i)}\}$ where $\hat{c}_n^{(i)}=0$ if $z_n^{cc,(i)}\geq 0$, and $\hat{c}_n^{(i)}=1$ otherwise.

$$z_n^{cc,(i)} = F_n^{cc} + \sum_{m'\in M(n)} \varepsilon_{m'n}^{cc,(i)} \quad (8)$$

$$z_n^{cc,(i)} = F_n^{cc} + \ell_n^{sc\rightarrow cc,(i)} + \sum_{m'\in M(n)} \varepsilon_{m'n}^{cc,(i)} \quad (9)$$

Compute the *a posteriori* LLRs $z_n^{sc,(i)}$ of the VNs in the source decoder using (10) for $n=N_{cc}+1,...,N_{cc}+N_{sc}$. Set $\hat{\mathbf{s}}=\{\hat{s}_n^{(i)}\}$ where $\hat{s}_n^{(i)}=0$ if $z_n^{sc,(i)}\geq 0$, and $\hat{s}_n^{(i)}=1$ otherwise.

$$z_n^{sc,(i)} = F_n^{sc} + \sum_{m'\in M(n)} \varepsilon_{m'n}^{sc,(i)} \quad (10)$$

Step 3: Stopping condition.

If $\mathbf{H}_{sc}\hat{\mathbf{s}}=\mathbf{0}$ and $\mathbf{H}_{cc}\hat{\mathbf{c}}=\mathbf{0}$ are both satisfied, or $i=I_{\max}$, the iteration will be stopped and go to Step 4. If the conditions are not met, set $i=i+1$ and go to Step 1.

Step 4: Output $\hat{\mathbf{s}}$ as the decoded source sequence.

The joint shuffled method contains the same steps as the traditional joint BP method, except that the LLR-updating process is different. We consider the i-th iteration. In the joint BP method, firstly all check-to-variable (C2V) messages are updated by utilizing the variable-to-check (V2C) messages calculated in the $(i-1)$-th iteration, i.e., every $\varepsilon_{mn}^{sc,(i)}$ is updated by utilizing $\{z_{mn'}^{sc,(i-1)} : n'\in N(m)/n\}$, and $\varepsilon_{mn}^{cc,(i)}$ is updated by utilizing $\{z_{mn'}^{cc,(i-1)} : n'\in N(m)/n\}$. Secondly, all the V2C messages are updated by using C2V messages that have been updated in the last step, i.e., every $z_{mn}^{sc,(i)}/z_{mn}^{cc,(i)}$ is updated from $\{\varepsilon_{m'n}^{sc,(i)} : m'\in M(n)/m\}/\{\varepsilon_{m'n}^{cc,(i)} : m'\in M(n)/m\}$.

Different from joint BP method, the joint shuffled method uses partly $z_{mn'}^{sc,(i)}$ and $z_{mn'}^{cc,(i)}$, which have been updated by $\varepsilon_{mn}^{sc,(i)}$ and $\varepsilon_{mn}^{cc,(i)}$, to replace $z_{mn'}^{sc,(i-1)}$ and $z_{mn'}^{cc,(i-1)}$. Then the remaining values of $\varepsilon_{mn}^{sc,(i)}$ and $\varepsilon_{mn}^{cc,(i)}$ are calculated by the updated V2C messages. In this way, more independent information can be utilized to allow the iterative process converging faster.

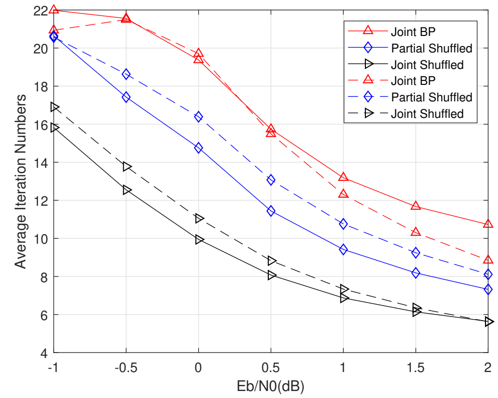

Fig. 3. The average number of iterations for the channel decoder (dashed curves) and the source decoder (solid curves) to converge under different decoding algorithms. $p=0.02$. Other simulation parameters are detailed in Section IV.

IV. Simulation Results

In this section, we illustrate the advantages of the joint shuffled decoding algorithm through simulations, and the simulation environment is in C++ using Visual Studio 2013. Besides the joint shuffled algorithm and BP algorithm [1], a partial shuffled decoding algorithm is also simulated. A partial shuffled algorithm can adopt the shuffled method to either the source or the channel decoder, while the other decoder keeps applying the BP algorithm. In the joint BP algorithm, the convergence speed of the source decoder is slower than that of the channel decoder. In order to improve source decoder's convergence speed, the shuffled method is adopted to the source decoder in the partial shuffled algorithm.

In all the simulations, the frame length is set as 3200 bits and the maximum number of decoding iterations $I_{\max}$ as 30. The rate-$1/4$ R4JA code is used as the source code and the rate-$1/2$ AR4JA code is used as channel code. Their corresponding base matrices are given as follows.

$$\mathrm{B}_{R4JA} = \begin{bmatrix} 3\,1\,3\,1\,3\,1\,1\,1 \\ 1\,2\,1\,3\,1\,3\,1\,2 \end{bmatrix}, \quad \mathrm{B}_{AR4JA} = \begin{bmatrix} 1\,2\,0\,0\,0 \\ 0\,3\,1\,1\,1 \\ 0\,1\,2\,2\,1 \end{bmatrix} \quad (11)$$

Fig. 3 and Fig. 4 plot the average number of iterations versus E_b/N_0. $p=0.02$ is assumed. In Fig. 3, the UCR phenomenon is clearly observed. The source decoder (solid curves) converges more slowly than channel decoder (dashed curves) in the joint BP algorithm (two red curves). Moreover, the partial shuffled algorithm can significantly outperform the joint BP algorithm by improving the convergence rate of the source decoder. For example, the average number of iterations is reduced from 16.4 to 13.3 at $E_b/N_0=0.5$ dB. Note that the partial shuffled algorithm not only directly improves the convergence rate of the source decoder, but

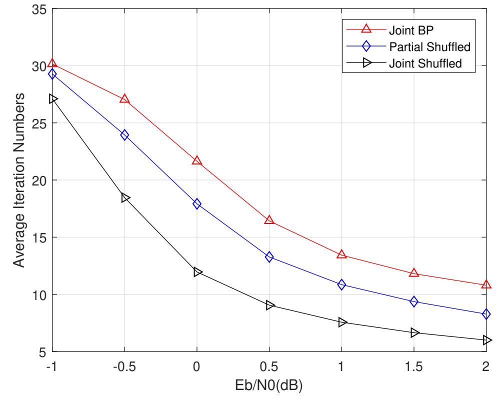

Fig. 4. The average number of iterations for different types of decoding algorithms. $p = 0.02$.

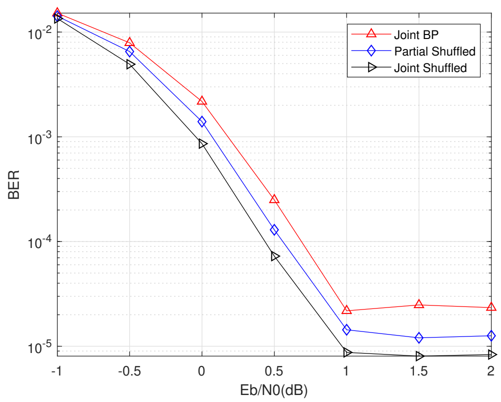

Fig. 5. BER performance of the decoding algorithms. $p = 0.02$.

also indirectly improves that of the channel decoder. Among the three algorithms, the joint shuffled algorithm achieves the lowest average number of iterations for both the source decoder and the channel decoder. In addition, the numbers of iterations required for convergence in both decoders are very similar. Fig. 4 shows that compared with those of the joint BP method and partial shuffled method, the average number of iterations for the joint shuffled algorithm is, respectively, 44% and 27% lower at $E_b/N_0 = 2$ dB. In other words, the convergence speed of the proposed algorithm is the fastest.

Fig. 5 and Fig. 6 plot the bit error rate (BER) performance of the different decoding algorithms for the DP-LDPC system under $p = 0.02$ and $p = 0.01$, respectively. Compared with joint BP scheme, the partial shuffled scheme can achieve improvement in both waterfall region and error-floor region. Moreover, among the three algorithms, the proposed joint shuffled algorithm can achieve the best BER performance in both waterfall region and error-floor region. For example, Fig. 6 shows that the BER performance of the joint shuffled scheme outperforms the joint BP method and the partial shuffled scheme by 0.2 dB and 0.1 dB, respectively, at $\text{BER} = 10^{-6}$.

Finally, we compare the overall decoding complexity of the proposed joint shuffled algorithm and the joint BP algorithm. In both algorithms, all the edge messages (i.e., $\varepsilon_{mn}^{cc,(i)}, z_{mn}^{cc,(i)}, \ell_n^{cc\to sc,(i)}, \varepsilon_{mn}^{sc,(i)}, z_{mn}^{sc,(i)}, \ell_n^{sc\to cc,(i)}$) are updated only once in each iteration. These two scheduling methods therefore have the same complexity per iteration. As the joint shuffled algorithm requires a smaller number of iterations to converge, its decoding complexity is lower than that of the joint BP algorithm.

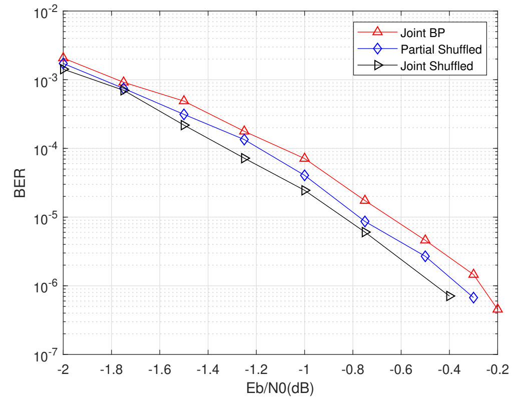

Fig. 6. BER performance of the decoding algorithms. $p = 0.01$.

V. Conclusions

In this letter, we have proposed a joint shuffled scheduling decoding algorithm to speed up the convergence speed of the DP-LDPC system. The algorithm can mitigate unequal convergence rate phenomenon, effectively reduce the number of iterations required for convergence, and hence lower the decoding complexity and power consumption. Simulation results show that the proposed algorithm achieves faster convergence speeds and better error performances in both the waterfall region and the error-floor region compared with the joint BP algorithm and the partial shuffled algorithm. Moreover, our proposed algorithm is applicable to other D-LDPC systems.

References

[1] J. G. He, L. Wang, and P. P. Chen, “A joint source and channel coding scheme base on simple protograph structed codes,” in *Proc. IEEE ISCIT*, pp. 65–69, Oct. 2012.

[2] M. Fresia, F. Perez-Cruz, H. V. Poor, and S. Verdu, “Joint source and channel coding,” *IEEE Signal Processing Magazine*, vol. 27, pp. 104–113, Nov. 2010.

[3] Q. Chen, L. Wang, S. Hong and Z. Xiong,“Performance Improvement of JSCC Scheme Through Redesigning Channel Code,” *IEEE Communications Letters*, vol. 20, no. 6, pp. 1088–1091, Jun. 2016.

[4] C. Chen, L. Wang and S. Liu, “The Design of Protograph LDPC Codes as Source Codes in a JSCC System,” *IEEE Communications Letters*, vol. 22, no. 4, pp. 672–675, Apr. 2018.

[5] C. Chen, L. Wang and F. C. M. Lau, “Joint Optimization of Protograph LDPC Code Pair for Joint Source and Channel Coding,” *IEEE Transactions on Communications*, vol. 66, no. 8, pp. 3255–3267, Aug. 2018.

[6] S. Hong, Q. Chen and L. Wang, “Performance analysis and optimisation for edge connection of JSCC system based on double protograph LDPC codes,” *IET Communications*, vol. 12, no. 2, pp. 214–219, Feb. 2018.

[7] J. He, Y. Li, G. Wu, S. Qian, Q. Xue and T. Matsumoto, “Performance Improvement of Joint Source-Channel Coding With Unequal Power Allocation,” *IEEE Wireless Communications Letters*, vol. 6, no. 5, pp. 582–585, Oct. 2017.

[8] T. Kim and J. Heo, “Information-Propagation-Based Scheduling for the Fast Convergence of Shuffled Decoding,” *IEEE Communications Letters*, vol. 22, no. 7, pp. 1314-1317, Jul. 2018.

[9] D. J. C. MacKay and R. M. Neal, “Near Shannon limit performance of low density parity check codes,” *Electronics Letters*, vol. 32, no. 18, pp. 1645–1646, Aug. 1996.

[10] J. Zhang and M. P. C. Fossorier, “Shuffled iterative decoding,” *IEEE Transactions on Communications*, vol. 53, no. 2, pp. 209–213, Feb. 2005.

[11] C. A. Aslam, Y. L. Guan and K. Cai, “Improving the Belief-Propagation Convergence of Irregular LDPC Codes Using Column-Weight Based Scheduling,” *IEEE Communications Letters*, vol. 19, no. 8, pp. 1283–1286, Aug. 2015.

[12] D. E. Hocevar, “A reduced complexity decoder architecture via layered decoding of LDPC codes,” *IEEE Workshop onSignal Processing Systems*, pp. 107-112, 2004.

[13] E. Sharon, S. Litsyn and J. Goldberger, “Efficient Serial Message-Passing Schedules for LDPC Decoding,” *IEEE Transactions on Information Theory*, vol. 53, no. 11, pp. 4076–4091, Nov. 2007.