Low-Complexity Belief-Propagation Decoding via Dynamic Silent-Variable-Node-Free Scheduling

Chaudhry Adnan Aslam, *Student Member, IEEE*, Yong Liang Guan, *Member, IEEE*,
Kui Cai, *Senior Member, IEEE*, and Guojun Han, *Senior Member, IEEE*

***Abstract*— This letter presents a low-complexity scheduling scheme, referred to as the dynamic silent-variable-node-free scheduling (D-SVNFS) scheme, for the sequential belief propagation decoding of LDPC code. The D-SVNFS regulates the dynamic propagation of message updates based on the check-to-variable message residuals. To determine the next message update, it computes the on-demand message residuals and selects the largest one associated with the last updated variable and check nodes. In addition, the D-SVNFS attempts to propagate more message updates toward the erroneous variable nodes, trying to correct them with higher priority. It is shown that the proposed scheduling reduces the BP decoding complexity by up to 70% compared with prior-art SVNF scheme, without affecting the error-rate performance, at medium to high signal-to-noise ratio over the BI-AWGN and Rayleigh fading channels.**

***Index Terms*— LDPC codes, belief-propagation, dynamic scheduling, decoding complexity, error-rate performance.**

I. INTRODUCTION

LDPC codes have drawn significant attention due to their superior error correction capability under the belief propagation (BP) decoding [1], in which the message updates are iteratively exchanged between the variable and check nodes of the bi-partite code graph. A BP *schedule*, which defines the order of message passing on this code graph, affects the convergence speed, complexity and error-rate performance of the BP decoder. The *flooding* is the conventional BP schedule, where all the variable-to-check (V2C), and subsequently the check-to-variable (C2V), message updates are simultaneously passed. In contrast, the serial schedules pass message updates in a fixed sequential order. The *shuffled* [2], [3] and *layered* [4], [5] are two commonly used variable and check node based sequential schedules which provide decoding convergence twice as fast as the flooding schedule.

To achieve faster convergence performance, a variety of dynamic BP schedules that rely on the message residuals to propagate new updates are reported. The *residual belief-propagation (RBP) [6]* is among the initial dynamic schedules, where the message associated with the largest C2V residual is updated first. Further improvements in the complexity and convergence performance are obtained by using the *node-wise RBP* (NW-RBP) and *approximate-node-wise RBP* (ANS-RBP) algorithms [7], where all the C2V messages originating from the check node with largest C2V residual are updated. Apart from C2V residual based schedules, V2C (VC-RBP) [8] and informed V2C (IVC-RBP) [9] residual based scheduling methods have also been reported in the open literature.

All the above-stated dynamic schedules introduce the existence of a greedy group, where a small number of nodes (edges) in the code graph inconspicuously occupy the computational resources and successively update their messages. Moreover, these dynamic schedules also tend to encounter some silent variable nodes which never get a chance to contribute in the decoding process at all. To overcome these limitations, the *silent-variable-node-free* (SVNF) schedule has been introduced in [10], which implements a fixed sequence of variable node participation by which an equal opportunity is provided to all the participating variable nodes to share their intrinsic information with other graph nodes.

In this letter, we present a *dynamic silent-variable-node-free scheduling* (D-SVNFS) scheme. Contrary to the SVNF scheme where the sequence of variable node participation is fixed, the proposed D-SVNFS scheme allows a dynamic sequence of variable node participation. Doing so not only eliminates the silent variable nodes (which motivates the SVNF), but also utilizes the latest message updates in the decoding process. Furthermore, the D-SVNFS scheme enforces the unsatisfied check nodes to pass new C2V message update. This prioritizes the message propagation towards the likely erroneous variable nodes connected with these unsatisfied check nodes, resulting in faster decoding convergence.

II. DYNAMIC SILENT-VARIABLE-NODE-FREE SCHEDULING (D-SVNFS)

The belief-propagation (BP) algorithm iteratively propagates message (reliability) updates from check node c_m to variable node v_n, $C_{c_m \to v_n}$, and from variable node v_n to check node c_m, $V_{v_n \to c_m}$, by using the following equations [1],

$$C_{c_m \to v_n} = 2 \tanh^{-1}\left(\prod_{v_j \in \mathcal{N}(c_m) \setminus v_n} \tanh\left(\frac{V_{v_j \to c_m}}{2}\right)\right) \tag{1}$$

$$V_{v_n \to c_m} = L_{v_n} + \sum_{c_j \in \mathcal{M}(v_n) \setminus c_m} C_{c_j \to v_n} \tag{2}$$

where L_{v_n} is the channel log-likelihood ratio (LLR) information of variable node v_n, $\mathcal{M}(v_n) \setminus c_m$ is the set of neighbors of variable node v_n except check node c_m and $\mathcal{N}(c_m) \setminus v_n$

Manuscript received August 22, 2016; revised September 22, 2016; accepted September 30, 2016. Date of publication October 4, 2016; date of current version January 6, 2017. This work was supported by the NTU Research Scholarship, in part by the SUTD-MIT International Design Center (IDC), and in part by the National Natural Science Foundation of China under grant number 61471131. The associate editor coordinating the review of this letter and approving it for publication was C. Jego.

C. Adnan Aslam and Y. L. Guan are with the School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore 639798 (e-mail: ad0001ry@e.ntu.edu.sg; eylguan@ntu.edu.sg).

K. Cai is with the Department of Science, Singapore University of Technology and Design, Singapore 487372 (e-mail: cai_kui@sutd.edu.sg).

G. Han is with Guangdong University of Technology, Guangzhou 510006, China (e-mail: gjhan@gdut.edu.cn).

Digital Object Identifier 10.1109/LCOMM.2016.2615016

is the set of neighbors of check node c_m except variable node v_n. The message passing continues until either the M parity-check (syndrome) equations are satisfied, $s_m = 0\ \forall c_m \in \{c_1, c_2, \ldots, c_M\}$, or when the maximum decoding iterations $I_{\max}$ are exhausted. In the C2V based dynamic schedules, the message residual denotes the absolute difference between the C2V message before and after an update, given by

$$r\left(c_m \to v_n\right) = ||C^{\text{pre}}_{c_m\to v_n} - C_{c_m\to v_n}|| \tag{3}$$

where $C^{\text{pre}}_{c_m\to v_n}$ is the pre-computed C2V message. The message propagation takes place between check node c_m and variable node v_n that corresponds to the largest residual $r\left(c_m \to v_n\right)$. Among the existing dynamic BP schedules, SVNF [10] provides the best error-rate performance by enforcing an equal variable node participation.

Although SVNF eliminates the silent (non participating) variable nodes, it does not specify the order of variable node participation. In fact, it only follows a fixed sequential order in each subsequent round of participation. However, it should be noted that a well-planned scheduling order can improve the decoding convergence. In particular, it has been shown that the updated variable nodes carry more reliable information, and incorporating their message updates can expedite the decoding process [2], [3]. This is how the variable node based sequential schedules achieve faster convergence than the flooding schedule. With this understanding, we propose to allow the last updated variable node to participate in the decoding process in the dynamic silent-variable node-free scheduling (D-SVNFS) scheme. This criterion ensures that the decoding process timely incorporates reliable message updates stemming from already updated variable nodes. The proposed D-SVNFS scheme also preserves the feature of silent-variable-node-free decoding by allowing each variable node to contribute only once in every single round of participation. To make this happen, it uses a binary vector $\mathbf{u} = \{u_n, 1 \le n \le N\}$ to mark the participation status. Whenever a variable node is selected, say v_p, its corresponding term in vector $\mathbf{u}$ is set to 1 ($u_p = 1$). Given a single round of participation, if an already participated variable node is selected twice, the D-SVNFS decoder pre-empts its participation and finds a new variable node who has yet to participate. On the completion of each participation round, the vector $\mathbf{u}$ is reset to zero to clear the participation status.

Furthermore, the proposed D-SVNFS scheme strives to correct the potentially erroneous variable nodes on a priority basis by allowing unsatisfied check nodes to forward new message updates. In this process, given a selected variable node v_p, only its unsatisfied (if any) neighbor check nodes are short-listed for the next C2V message propagation based on the largest C2V residual. This step expedites the decoding convergence speed as the unreliable variable nodes (connected with unsatisfied check nodes) will receive new message updates more often.

To summarize the variable node selection method, if the last updated variable node, say v_p, is eligible for participation (i.e. $u_p = 0$), the next C2V message is selected from the unsatisfied neighbor check nodes of v_p. Otherwise, the next eligible variable node is chosen for decoding participation. Targeting the updated variable nodes for message participation and unsatisfied check nodes for message propagation makes a significant impact on the error-rate convergence and complexity of BP decoder. The detailed steps of the proposed D-SVNFS scheme are outlined in Algorithm 1.

Algorithm 1 Proposed D-SVNFS Scheme

```
1  Initialize all $V_{v_n\to c_m} = L_{v_n}$, $C_{c_m\to v_n} = 0$;
2  **while** *stopping rule is not satisfied* **do**
3      **if** *all* $\{u_n = 1, 1 \le n \le N\}$ **then**
4          Reset $\{u_n = 0,\ 1 \le n \le N\}$;
5      Select $v_p : u_p = 0, u_i = 1\ \forall i < p$. Set $u_p = 1$;
6      **if** $s_m = 1,\ c_m \in \mathcal{M}\left(v_p\right)$ **then**
7          **for** $c_m \in \mathcal{M}\left(v_p\right), s_m = 1, v_n \in \mathcal{N}\left(c_m\right) \setminus v_p$ **do**
8              Compute $C^{\text{pre}}_{c_m\to v_n}$, $r\left(c_m \to v_n\right)$ with (1), (3);
9          Find $r\left(c_i \to v_j\right) = \mathbf{max}\{r\left(c_m \to v_n\right) |$
10         $c_m \in \mathcal{M}\left(v_p\right), s_m = 1, v_n \in \mathcal{N}\left(c_m\right) \setminus v_p\}$;
11     **else**
12         **for** $c_m \in \mathcal{M}\left(v_p\right), v_n \in \mathcal{N}\left(c_m\right) \setminus v_p$ **do**
13             Compute $C^{\text{pre}}_{c_m\to v_n}$, $r\left(c_m \to v_n\right)$ with (1), (3);
14         Find $r\left(c_i \to v_j\right) = \mathbf{max}\{r\left(c_m \to v_n\right) |$
15         $c_m \in \mathcal{M}\left(v_p\right), v_n \in \mathcal{N}\left(c_m\right) \setminus v_p\}$;
16     Generate and propagate $C_{c_i\to v_j}$ with (1);
17     Set $r\left(c_i \to v_j\right) = 0$;
18     **for** $c_a \in \mathcal{M}\left(v_j\right) \setminus c_i$ **do**
19         Generate and propagate $V_{v_j\to c_a}$ with (2);
20     **if** $u_j = 1$ **then**
21         Proceed to 3;
22     **else**
23         Set $v_p = v_j$, $u_p = 1$. Proceed to 6;
24 End
```

Example 1: Fig. 1 shows the execution of D-SVNFS scheme over a simple code graph. Here, it is assumed that the check node equations corresponding to c_2, c_3 and c_5 are unsatisfied ($s_2 = 1$, $s_3 = 1$, $s_5 = 1$) due to the incorrect variable node v_5. Decoding starts with the participation of variable node v_1, as shown in Fig. 1(a). In this case, only residuals from the unsatisfied check node c_2 are computed on-demand and used for comparison. Based on the largest residual $r\left(c_2 \to v_5\right)$, C2V message $C_{c_2\to v_5}$, and V2C messages $V_{v_5\to c_3}$ and $V_{v_5\to c_5}$ are propagated. Next, the recently updated variable node v_5 is selected. After computing the required (new) on-demand residuals from the unsatisfied check nodes c_2, c_3 and c_5, and finding the largest one among them, the C2V and V2C message updates are propagated accordingly, as shown in Fig. 1(b). Similarly, the participation of variable node v_6, being the last updated node, is followed, as shown in Fig. 1(c). Following the participation of v_6, the last updated variable node v_5 is found to be ineligible for participation ($u_5 = 1$). Therefore, the next available variable node, v_2 in this case ($u_2 = 0$), is given the opportunity to participate, as shown in Fig. 1(d). First round of participation ends when all the variable nodes contribute in the decoding process. In contrast, the SVNF schedule will follow a fixed sequence of participation, given by: $v_1 \Rightarrow v_2 \Rightarrow v_3 \Rightarrow v_4 \Rightarrow v_5 \Rightarrow v_6 \Rightarrow v_7$.

TABLE I

COMPLEXITY ANALYSIS OF FIXED AND DYNAMIC SCHEDULING SCHEMES (PER ITERATION)

Schedule	V2C update	C2V update	Residual computation	Real-value comparison
Flooding [1] / Shuffled [2] / Layered [4]	E	E	0	0
IVC-RBP [9]	E	$E\left[\left(\bar{d}_v-1\right)+\left(\bar{d}_c-1\right)\right]$	$E\left(\bar{d}_v-1\right)\left(\bar{d}_c-1\right)$	$E\left(E-1\right)$
NW-RBP [7]	$E\left(\bar{d}_v-1\right)$	E	$E\left(\bar{d}_v-1\right)\left(\bar{d}_c-1\right)$	$E\left(E-1\right)/\bar{d}_c$
SVNF [10]	$E\left(\bar{d}_v-1\right)$	E	$E\left(\bar{d}_v-1\right)\left(\bar{d}_c-1\right)$	$E\left[\bar{d}_v\left(\bar{d}_c-1\right)-1\right]$
D-SVNFS (proposed)	$E\left(\bar{d}_v-1\right)$	E	$\leqslant E\left(\bar{d}_v-1\right)\left(\bar{d}_c-1\right)$	$\leqslant E\left[\bar{d}_v\left(\bar{d}_c-1\right)-1\right]$

$E \Rightarrow$ total edges in the code graph. $\bar{d}_v \Rightarrow$ average variable node degree. $\bar{d}_c \Rightarrow$ average check node degree.

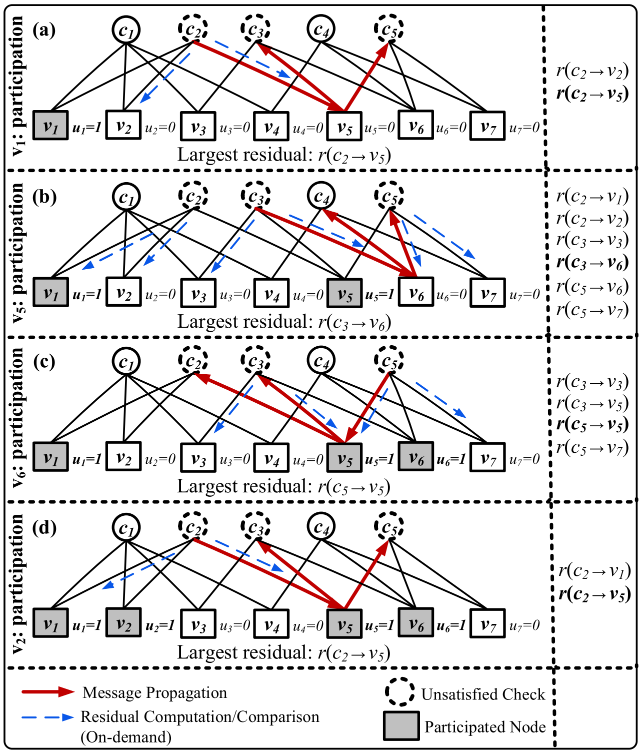

Fig. 1. BP decoding using the proposed dynamic silent-variable-node-free scheduling (D-SVNFS): dynamic order of variable node participation (first round follows $v_1 \Rightarrow v_5 \Rightarrow v_6 \Rightarrow v_2$ and so on).

III. D-SVNFS PERFORMANCE EVALUATION

A. Complexity Analysis (per Iteration)

Table I presents the total number of message updates (V2C/C2V), residual computations and real comparisons performed per decoding iteration, where one iteration can be counted as E number of C2V update. For Flooding [1], Shuffled [2] and Layered [4] schedules, E number of V2C and C2V message updates are performed, whereas no additional run-time operations are required. Similarly, NW-RBP [7], SVNF [10] and D-SVNFS schemes also perform E number of C2V message update, while IVC-RBP [9] performs E number of V2C message update. The prior-art IVC-RBP, NW-RBP and SVNF schemes require the same number of residual computations. Comparing with the SVNF scheduling, the proposed D-SVNFS scheme requires fewer residual computations and real-value comparisons as the C2V residuals are computed and compared on-demand only from the unsatisfied neighbor check nodes of participating variable node.

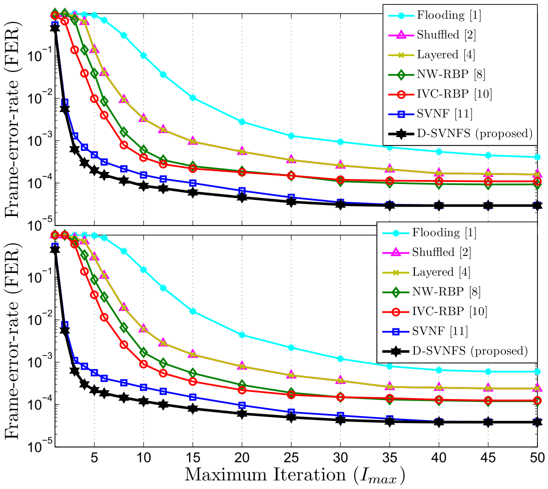

Fig. 2. Frame-error-rate (FER) vs. number of iterations: WIMAX code [11], rate-1/2, block-length 1152, SNR = 2.25 dB for BI-AWGN (*top*) and SNR = 4.50 dB for Fading (*bottom*) channels.

B. Error-Rate Performance and Average Complexity

To show the error-rate and average complexity results, we construct two binary LDPC codes of rate-1/2 and block-length 1152 and 1296 bits, based on the IEEE WIMAX [11] and WIFI [12] standards, respectively. Both LDPC codes have distinct structured parity check matrices, each partitioned into sub-matrices consisting of the cyclic-permutation and all-zero matrices. Keeping the SNR fixed at 2.25 dB and 4.50 dB for BI-AWGN and Rayleigh fading channels, we plot the frame-error-rate (FER) curves against the maximum number of BP iteration ($I_{\max}$) for these 2 codes in Fig. 2 and Fig. 3, respectively. It can be observed that the D-SVNFS and SVNF schemes outperform the NW-RBP and IVC-RCP schedules, irrespective of the maximum iteration number. This is because the greedy nature of the later schemes limits their error-rate performance. For iteration count less than 25, the D-SVNFS scheme performs better than the SVNF scheme.

Apart from the error-rate performance, it is the average decoding complexity that makes the D-SVNFS more attractive than the SVNF scheme. Here, the complexity is measured in terms of the average number of message *updates* (V2C/C2V), message *residuals* and real *comparisons*. To show the complexity results, we simulate the LDPC codes under the D-SVNFS and SVNF schedules until the decoding converges or when $I_{\max} = 50$ iteration is reached, and record the average number of message updates, residual computations and real comparison operations. In order to demonstrate the relative performance improvement, we plot the complexity reduction

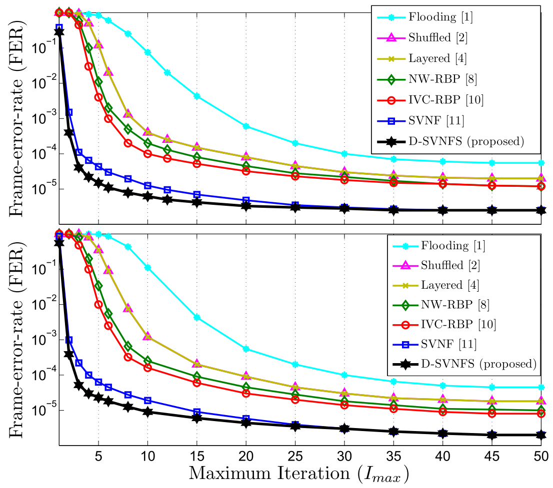

Fig. 3. Frame-error-rate (FER) vs. number of iterations: WIFI code [12], rate-1/2, block-length 1291, SNR = 2.25 dB for BI-AWGN (*top*) and SNR = 4.50 dB for Fading (*bottom*) channels.

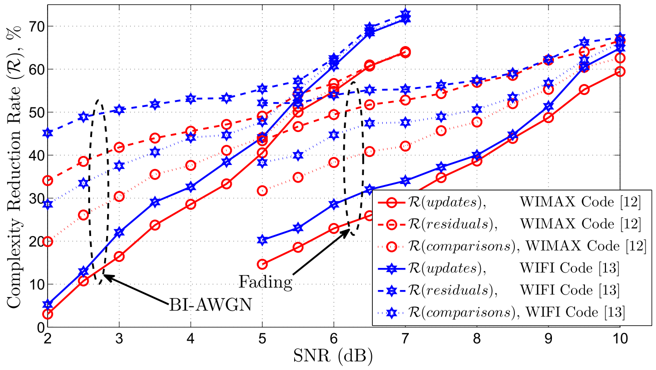

Fig. 4. Reduction in V2C/C2V message updates, $\mathcal{R}(updates)$, message residuals, $\mathcal{R}(residuals)$, and real comparisons, $\mathcal{R}(comparisons)$, employing the proposed D-SVNFS scheme, compared to SVNF [10]: WIMAX [11] and WIFI [12] codes.

rate ($\mathcal{R}$) in Fig. 4, given by

$$\mathcal{R} = \left(1 - \frac{\{updates|residuals|comparisons\}_{\text{D-SVNFS}}}{\{updates|residuals|comparisons\}_{\text{SVNF}}}\right)100 \tag{4}$$

It can be observed that the D-SVNFS scheme provides reduction in the number of message updates, $\mathcal{R}(updates)$, message residuals, $\mathcal{R}(residuals)$, and real comparisons, $\mathcal{R}(comparisons)$, by 60% to 70% at medium-to-high SNR under BI-AWGN and Rayleigh fading channels. This gain is obtained as a result of utilizing the latest and more reliable message updates. Furthermore, since the unsatisfied parity-check constraint is imposed on the selection of next C2V message, relative to SVNF scheme, the D-SVNFS propagates more number of message updates towards the erroneous variable nodes. This is shown in Fig. 5 where the fraction of total C2V message updates sent towards erroneous variable nodes are plotted. Consequently, these erroneous variable nodes are corrected more rapidly, requiring fewer number of updates to achieve the error-rate convergence.

IV. CONCLUSION

A C2V residual-based dynamic silent-variable-node-free scheduling (D-SVNFS) scheme is introduced. In this scheme, the last updated variable and check nodes are allowed to participate in the decoding process, ensuring that more reliable updates are exchanged between the variable and check nodes. Meanwhile, to prioritize the correction of erroneous variable nodes, the unsatisfied check nodes are enforced to propagate the next message update towards the likely erroneous variable nodes. Simulation results shows significant complexity reduction under the proposed D-SVNFS scheme. Moreover, the error performance improves for the initial decoding iterations and remains comparable to the prior-art scheduling schemes for subsequent decoding iterations.

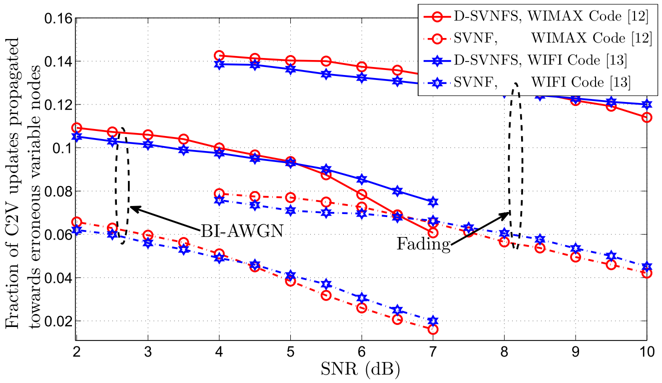

Fig. 5. Fraction of C2V message updates propagated towards erroneous variable nodes: WIMAX [11] and WIFI [12] codes.

REFERENCES

[1] R. Gallager, "Low-density parity-check codes," *IRE Trans. Inf. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[2] J. Zhang and M. P. C. Fossorier, "Shuffled iterative decoding," *IEEE Trans. Commun.*, vol. 53, no. 2, pp. 209–213, Feb. 2005.

[3] C. A. Aslam, Y. L. Guan, and K. Cai, "Improving the belief-propagation convergence of irregular LDPC codes using column-weight based scheduling," *IEEE Commun. Lett.*, vol. 19, no. 8, pp. 1283–1286, Aug. 2015.

[4] D. E. Hocevar, "A reduced complexity decoder architecture via layered decoding of LDPC codes," in *Proc. IEEE Workshop. Signal Process. Syst.*, Oct. 2004, pp. 107–112.

[5] M. Kim, D. Kim, and Y. H. Lee, "Serial scheduling algorithm of LDPC decoding for multimedia transmission," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast.*, Jun. 2012, pp. 1–4.

[6] A. I. V. Casado, M. Griot, and R. D. Wesel, "Informed dynamic scheduling for belief-propagation decoding of LDPC codes," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2007, pp. 932–937.

[7] A. I. V. 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.

[8] J.-H. Kim, M.-Y. Nam, and H.-Y. Song, "Variable-to-check residual belief propagation for LDPC codes," *IET Electron. Lett.*, vol. 45, no. 2, pp. 117–118, Jan. 2009.

[9] 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.

[10] H.-C. Lee, Y.-L. Ueng, S.-M. Yeh, and W.-Y. Weng, "Two informed dynamic scheduling strategies for iterative LDPC decoders," *IEEE Trans. Commun.*, vol. 61, no. 3, pp. 886–896, Mar. 2013.

[11] *IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface for Broadband Wireless Access Systems*, IEEE Standard 802.16-2009, 2009.

[12] *IEEE Draft Standard for Information Technology—Telecommunications and Information Exchange Between Systems—Local and Metropolitan Area Networks—Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard 802.11-2012, 2012.