Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 6698602, 11 pages
https://doi.org/10.1155/2021/6698602

Research Article

A New Reliability Ratio Weighted Bit Flipping Algorithm for Decoding LDPC Codes

Chakir Aqil, Ismail Akharraz, and Abdelaziz Ahaitouf

Engineering Sciences Laboratory, Sidi Mohamed Ben Abdellah University, Fez, Morocco

Correspondence should be addressed to Chakir Aqil; chakir.aqil@usmba.ac.ma

Received 9 October 2020; Revised 18 March 2021; Accepted 5 May 2021; Published 17 May 2021

Academic Editor: Simone Morosi

In this study, we propose a "New Reliability Ratio Weighted Bit Flipping" (NRRWBF) algorithm for Low-Density Parity-Check (LDPC) codes. This algorithm improves the "Reliability Ratio Weighted Bit Flipping" (RRWBF) algorithm by modifying the reliability ratio. It surpasses the RRWBF in performance, reaching a 0.6 dB coding gain at a Binary Error Rate (BER) of 10^{-4} over the Additive White Gaussian Noise (AWGN) channel, and presents a significant reduction in the decoding complexity. Furthermore, we improved NRRWBF using the sum of the syndromes as a criterion to avoid the infinite loop. This will enable the decoder to attain a more efficient and effective decoding performance.

1. Introduction

Low-Density Parity-Check (LDPC) codes form a block code class characterized by the null space of sparse parity-check matrix H. They were first discovered by Gallager in the early 1960s [1]. In the late 1990s, LDPC codes were rediscovered by MacKay and Neal [2]. Since then, the LDPC codes have been very successful because of their error correction performance which is very close to the Shannon limit [3] and have been adopted in many wireless standards including DVB-S2 [4], IEEE 802.15 [5], IEEE 802.11n [6], and IEEE 802.16e [7] as well as long-term evolution-advanced (LTE-A) communication standards [8]. LDPC codes will play an important role in wireless communication systems in the future, as it has been confirmed that the LDPC codes will be adopted in the 5G system for the enhanced mobile broadband scenario [9].

LDPC codes can be decoded mainly by two algorithms' families. The first one is the so-called soft decision algorithms. Their original algorithm is called Belief Propagation (BP) [2], also called the Sum-Product (SP) algorithm in some publications [10, 11]. Several modified versions of the BP decoding algorithm have been introduced, for instance, the Min-Sum (MS) [12], Offset Min-Sum (OMS), Normalized Min-Sum [13, 14], and Relaxed Min-Sum (RMS) [15] algorithms.

The BP algorithm is known for its effective performance in error correction, but it takes a lot of multiplications to update the variable nodes and controls, which makes it difficult to use in an efficient hardware implementation. Different variants have been proposed for reducing its complexity and improving its performance. Despite these attempts to achieve these goals, this family of decoding algorithms remains much higher in complexity compared to the second decoding family.

This second one includes the algorithms known as the hard decision which originated from the Bit Flipping (BF) algorithm [16]. This algorithm has low complexity but considerable degradation in performance. To improve this property, several derivatives have been proposed in the literature. It began with Weighted Bit Flipping (WBF) [17], Modified Weighted Bit Flipping (MWBF) [18], and Improvement on the Modified Weighted Bit Flipping (IMWBF) [19]. These decoding algorithms are based mainly on the weight of checksums and differ by the use of these weights in the flipping function. The RRWBF [20] is another decoding algorithm which was introduced to improve the BF algorithm. It uses a reliability ratio different from the weight of previous algorithms and demonstrates a significant improvement in performance. An Improved Low Complex Hybrid Weighted Bit Flipping (ILCHWBF) [21] algorithm is an important

hard decision algorithm compared to the previously mentioned ones because the weight used includes parameters from both the MS and BF variants along with an experimental factor determined by simulations.

Researchers in this field continued to propose decoding algorithms to improve the BF algorithm by minimizing the difference in performance compared to the SP one. Among these proposed algorithms in recent years, and which present interesting results, we can cite the following:

(1) Mixed Modified Weighted Bit Flipping (MMWBF) [22]. This algorithm offers a significant performance gain compared to the WBF algorithm and achieves a compromise of performance and complexity between the BF and SP algorithms. This algorithm will be further explained.

(2) Reliability Variance Weighted Bit Flipping (RVWBF) Algorithms for LDPC [23]. The authors of this work used the variance of the values received. These modifications improve performance with a slight increase in calculations at the level of the inversion function and decrease the average number of iterations to maintain high reliability and decoding at a higher speed.

(3) Modified Weighted Bit Flipping Algorithm Based on Intrinsic Information (MWBFII) [24]. This algorithm is based on the reliability of intrinsic information, in opposition to other existing ones, where reliability is based on extrinsic information used in the inversion function. This algorithm only uses additions and subtractions to modify reliability which uses multiplications and divisions.

(4) Classification-Based Algorithm for Bit Flipping Decoding (CBFD) of GLDPC Codes over AWGN Channels [25]. Here, the authors took advantage of fast BF decoding with only the initial help of soft channel information by introducing a classification step at first. Second, they added an auxiliary bit in messages between the variable and control nodes as a tool to increase decision reliability and improve performance. The reliability is presented as a tool to reduce the effect of the trapping sets generated.

(5) In a Two-Bit Weighted Bit Flipping Decoding (TBWFD) Algorithm [26]. The authors produced reliability bits for the bit decision results and syndrome values at the bit and control nodes, respectively. Reliability bits are exchanged between the bit and control nodes as the decoding progresses. Updating messages in control nodes is carefully designed with simple bitwise operations.

(6) Cyclic Switching Weighted Bit Flipping Decoding (CSWBFD) for Low-Density Parity-Check Codes [27]. For this algorithm, the authors carefully chose two criteria for selecting the bit to return and by passing cyclically from one criterion to another according to certain rules during decoding. This proposition can effectively break the infinite decoding loop, which often appears for the Weighted Bit Flipping algorithms and greatly degrades the decoding performance.

(7) Hard Decision Bit Flipping Decoder Based on Adaptive Bit-Local Threshold (HDBFABLT) for LDPC Codes [28]. It consists of an adaptive local threshold bit switching algorithm to improve the decoding performance of LDPC codes. The authors used a threshold for each bit which can be modified adaptively to maximize the number of errors corrected during iterations.

(8) High-Throughput Bit Flipping Decoder for Structured (HTBFS) LDPC Codes [29]. A high-speed parallel Bit Flipping (BF) decoder using the BF multiple threshold algorithm is proposed in this work. The decoder is endowed with functionalities such as low interconnection complexity, simpler calculations, and high speed.

(9) Multistage Bit Flipping Decoding (MSBFD) Algorithms for LDPC Codes [30]. Two algorithms, which are made up of soft decision and hard decision BF decoding parts, are presented. This approach is based on the transition from one algorithm to another and the choice of the algorithm which plays the first role and the second one. This is done by adjusting certain parameters to obtain optimal performance.

Within this sense, in the current paper, we propose an improvement of the RRWBF algorithm called "New Reliability Ratio Weighted Bit Flipping" (NRRWBF). In this algorithm, we propose a change in the inversion function, especially in the syndrome's weight (reliability ratio), to improve the performance and to minimize, even more, the difference in performance between the SP algorithms and the BF ones. In a second step, we added an improvement to our algorithm by using the criterion of the syndrome's sum to avoid falling into the infinite loop of decoding. We called this new proposition "Improvement of the New Reliability Ratio Weighted Bit Flipping" (INRRWBF).

After the first section which established an introduction for this study, the rest of this document is organized as follows. Section 2 gives a brief overview of the various BF decoding algorithms, their decoding steps, and their inversion functions. After this, a description of the NRRWBF algorithm is discussed in Section 3. Section 4 deals with the analysis of the complexity of the NRRWBF algorithm, and the description of the INRRWBF algorithm is introduced in Section 5. In Section 6, the simulation results obtained with MATLAB and the discussion of the results are described. Finally, a conclusion of this document is given in Section 7.

2. Brief Review of BF and Its Various Decoding Algorithms

LDPC codes are defined by a hollow matrix (the number of zeros greater than the number of ones) called the parity-check matrix $H = [h_{m,n}]$. This matrix is characterized by N columns which represent the length of the code and M rows which represent the parity-check equations. One can also define a LDPC code by a bipartite graph called a tanner graph. This graph contains two types of nodes: the variable nodes correspond to the number of columns N, and the control nodes correspond to the number of rows M in the H matrix. A variable node n is connected to a control node m if $h_{m,n} = 1$. We define the column weight or column degree d_c as the number of ones per column and the row weight or row degree d_r as the number of ones per row. If d_r is the same for all the rows and d_c is constant for any column, the LDPC code is regular; otherwise, it is irregular.

It is assumed that the binary codeword message is $(c_1, c_2, \cdots, c_N)$; this message becomes $(x_1, x_2, \cdots, x_N)$ using the Binary Phase-Shift Keying (BPSK) modulation with $x_i = 2c_i - 1, i \in [1, N]$. Then, the sequence is transmitted on an Additive White Gaussian Noise (AWGN) channel. The received symbol is $(y_1, y_2, \cdots, y_N)$ with $y_i = x_i + v_i$, and v_i is the Additive White Gaussian Noise with mean zero and variance $N_0/2$.

The following are defined:

(1) $V(m) = \{n : h_{m,n} = 1\}$ represents the set of variable nodes participating in the m control node

(2) $C(n) = \{m : h_{m,n} = 1\}$ represents all the control nodes participating in the n variable node

An example of an H matrix is shown below. It is a very small matrix compared to the used ones in our algorithm.

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}. \quad (1)$$

This regular dimension matrix ($M = 5$, $N = 10$) ($d_c = 3$, $d_r = 6$) can be transformed into a tanner graph with 10 variable nodes and 5 control nodes as shown in Figure 1; a variable node is connected to a control node if and only if the corresponding element in the matrix H is 1.

2.1. BF Decoding Algorithms. In this section, we provide the steps of the BF decoding algorithm and some of its derivatives in general. After receiving the information from the channel, the initialization step is performed to find the estimated values that will be used in the decoding steps.

We initialize a variable z_j (estimation) with $j \in [1, N]$, which is the firm decision on each received bit y_j by

$$\begin{aligned} z_j &= 1 \quad \text{if } y_j \geq 0, \\ z_j &= 0 \quad \text{if } y_j < 0. \end{aligned} \quad (2)$$

The iterative process (decoding steps in general) includes the following:

(1) We calculate the syndrome bits: $s_m = \sum_{n=0}^{N-1} z_n H_{m,n}$, where $m = 1, 2, 3, \cdots, M$ and the matrix product is modulo 2. If all parity control equations are satisfied (i.e., $s_m = 0$), the decoder is passed and the iterative process will be completed. Otherwise, it proceeds to the second step

(2) We calculate the inversion function (the error term) for each variable node as follows:

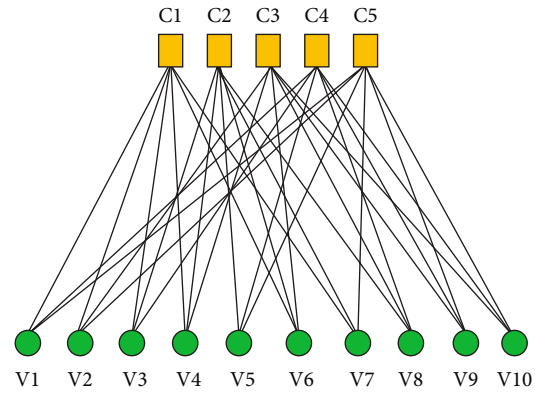

Figure 1: Tanner graph of matrix H.

$$E_n = \sum_{m \in C(n)} (2s_m - 1)\gamma - \delta|y_n|, \quad (3)$$

where s_m is the syndrome bit, with $m \in [1, M]$. γ and δ are parameters related to each algorithm as we will see later, and $|y_n|$ represents the absolute value of y_n received by the variable node channel n.

(3) We determine which variable node has the highest value of E_n ($n = \text{argmax}_{1 \leq n \leq N} E_n$). The estimated value of this node must be flipped

(4) We repeat steps 1-3 until all parity control equations are satisfied or a predefined number of iterations is reached

2.2. The Inversion Function of BF Decoding Algorithms. The inversion function (IF) of the decoding algorithm differs from one algorithm to another depending on the relationships or values given to the parameters γ and δ.

For the original BF algorithm, $\gamma = 1$ and $\delta = 0$.

The IF becomes

$$E_n = \sum_{m \in C(n)} (2s_m - 1). \quad (4)$$

For the WBF algorithm, we find $\gamma = y_m^{\min}$ and $\delta = 0$, with $y_m^{\min} = \min_{\{n:n \in V(m)\}} |y_n|$ which represents the least reliable variable node associated with each control node.

The IF becomes

$$E_n = \sum_{m \in C(n)} (2s_m - 1)\, y_m^{\min}. \quad (5)$$

For the MWBF algorithm, we find $\gamma = y_m^{\min}$ and $\delta = \alpha$, where α is an experimentally determined coefficient.

The IF becomes

$$E_n = \sum_{m \in C(n)} (2s_m - 1)\, y_m^{\min} - \alpha|y_n|. \quad (6)$$

For the IMWBF algorithm, we find $\gamma = y_{n,m}^{\min}$ and $\delta = \alpha$,

Initialization. Set the iteration counter $I = 1$, and initialize a variable z (estimation).
Step 1. Calculate the syndrome bits: $s_m = \sum_{n=0}^{N-1} z_n H_{m,n}$, where $m = 1, 2, 3, \cdots, M$. If all the syndrome bits are zero, then the decoding process is stopped and output z; otherwise, go to step 2.
Step 2. Calculate the inversion function for each variable node by
$E_n = \sum_{m \in C(n)} (2s_m - 1)\,(d_r \cdot y_m^{\min}/|y_n|)$.
Step 3. Determine which variable node has the highest value of E_n ($n = \text{argmax}_{1 \le n \le N} E_n$). The estimated value of this node must be flipped.
Step 4. If $I = I_{\max}$ (a predefined iteration limit is reached), then stop and output the estimation codeword; else, $I \leftarrow I + 1$ and go to step 1.

Algorithm 1: NRRWBF decoding algorithm.

Table 1: Complexity for different algorithms with AWGN channels.

Operation	Algorithm		
	WBF	RRWBF	NRRWBF
Addition	$(d_c - 1)N + (d_r - 1)M$	$(d_r d_c - 1)N + (d_r - 1)M$	$(d_c - 1)N + (d_r - 1)M$
Multiplication	Nd_c	Nd_c	Nd_c
Division	0	N	N
Comparison	$(d_r - 1)M$	0	$(d_r - 1)M$

where $y_{n,m}^{\min} = \min_{\{i \in V(m)/n\}} |y_i|$ represents the minimum of each control node linked to the bit n excluding the information coming from the bit itself (variable node n).

The IF becomes

$$E_n = \sum_{m \in C(n)} (2s_m - 1) y_{n,m}^{\min} - \alpha |y_n|. \tag{7}$$

For the RRWBF algorithm, we find $\gamma = \sum_{n' \in V(m)} |y_{n'}|/|y_n|$ and $\delta = 0$.

The IF becomes

$$E_n = \sum_{m \in C(n)} (2s_m - 1)\, \frac{\sum_{n' \in V(m)} |y_{n'}|}{|y_n|}. \tag{8}$$

For the ILCHWBF algorithm, we find $\gamma = y_{n,m}^{\min}$ and $\delta = 0$.

The IF becomes

$$E_n = \frac{1}{|y_n|} \sum_{m \in C(n)} (2s_m - 1)\, y_{n,m}^{\min} \cdot \theta_{\text{attn}}, \tag{9}$$

where θ_{attn} is an attenuation factor used to improve the accuracy of weighted extrinsic information, with $0 < \theta_{\text{attn}} < 1$.

For MMWBF [21] algorithm, the two WBF algorithms are mixed: RRWBF and IMWBF, acting, respectively, as a primary decoding algorithm and as an auxiliary algorithm.

The inversion function is calculated for each variable node as follows:

Table 2: The number of required operations for different algorithms.

	WBF	RRWBF	NRRWBF
Addition	6900	21300	6900
Multiplication	7200	7200	7200
Division	0	600	600
Comparison	1650	0	1650

In the main algorithm,

$$E_n^1 = \sum_{m \in C(n)} (2s_m - 1) \frac{\sum_{n' \in V(m)} |y_{n'}|}{|y_n|}. \tag{10}$$

In the auxiliary algorithm,

$$E_n^2 = \sum_{m \in C(n)} (2s_m - 1) y_{n,m}^{\min} - \alpha |y_n|. \tag{11}$$

According to E_n^1 and E_n^2, we determine the positions of the nodes of variables most likely to errors by

$$\begin{cases} p = \text{argmax}\, E_n^1 \\ q = \text{argmax}\, E_n^2 \end{cases}. \tag{12}$$

If $p \neq q$ and the q bit position is not switched by the main algorithm, we switch the two bits z_p and z_q; otherwise, we switch only z_p.

Initialization. Set the iteration counter $I = 1$, excluding set $\Omega = \emptyset$, and initialize a variable z (estimation).
Step 1. Calculate the syndrome bits: $s_m(I) = \sum_{n=0}^{N-1} z_n H_{m,n}$, where $m = 1, 2, 3, \cdots, M$. If the $\text{sum}(s_m(I)) = 0$ (the sum of the syndrome bits at iteration I) or $I = I_{\max}$ (a predefined iteration limit is reached), stop and output z; else, $I \leftarrow I + 1$ and go to step 2.
Step 2. Calculate the inversion function for each variable node by
$E_n = \sum_{m \in C(n)} (2s_m - 1)\,(d_r \cdot y_m^{\min} / |y_n|)$.
Step 3. Determine which variable node has the highest value of E_n ($n^* = \text{argmax}_{1 \le n \le N} E_n$). The estimated value of this node must be flipped.
Step 4. Calculate the syndrome $s_m(I + 1)$ again, where $m = 1, 2, 3, \cdots, M$; if the $\text{sum}(s_m(I + 1)) < \text{sum}(s_m(I))$, record the position n^*, then set $\Omega = \Omega \cup \{n^*\}$; otherwise, the bit n^* must be flipped for the second time.

Algorithm 2: INRRWBF decoding algorithm.

3. A New Reliability Ratio Weighted Bit Flipping Decoding Algorithm

Hard decision decoding algorithms are based on the calculation of the inversion function. This function determines the least reliable bit to be flipped at each iteration. Because of this, the parameters used in this function are very important; they must be chosen intelligently.

After exploring several algorithms known in the literature, such as MS [12], WBF [17], and MWBF [18], we noticed that they use, in their messages sent from control nodes to variable nodes for the MS and its variants and in the inversion functions for the WBF and its variants, the $y_m^{\min}$ which represents the minimum value of each parity-check node. The importance of $y_m^{\min}$ is that it is the least reliable bit for every parity-check node. Thus, there is a high probability that the inversion function calculated using $y_m^{\min}$ leads us to the right bit to be flipped at each iteration.

Henceforth, we chose the RRWBF algorithm to improve it by integrating the $y_m^{\min}$ in its inversion function.

The RRWBF algorithm, which performs well against some variants of WBF, uses the inversion function E_n:

$$E_n = \sum_{m \in C(n)} (2s_m - 1) \frac{\sum_{n' \in V(m)} |y_{n'}|}{|y_n|}. \tag{13}$$

So, we replace the weight $\sum_{n' \in V(m)} |y_{n'}|$ in the inversion function of the RRWBF by $d_r \cdot y_m^{\min}$, where $y_m^{\min} = \min_{\{n : n \in V(m)\}} |y_n|$.

Therefore, we simplified the sum of messages $\sum_{n' \in V(m)} |y_{n'}|$ by using $d_r \cdot y_m^{\min}$ with d_r the row weight as the number of ones per row in the check matrix H.

The inversion function E_n for the NRRWBF algorithm becomes

$$E_n = \sum_{m \in C(n)} (2s_m - 1) \frac{d_r \cdot y_m^{\min}}{|y_n|}. \tag{14}$$

The NRRWBF algorithm is summarized in Algorithm 1.

Table 3: Parameters used in simulations.

	Code 1	Code 2	Code 3
Number of bits of information K	450	750	1050
Number of parity-check equations M	150	150	150
Length of the codeword N	600	900	1200
Coding rate $R = K/N$	0.75	0.83	0.875
Column weight d_c	3	3	3
Row weight d_r	12	18	24

4. Complexity Analysis

In this section, we analyze the complexity of calculating the inversion function and syndrome to determine which bit will be flipped for iteration. We consider a LDPC code with the weight of the column d_c and the weight of the row d_r. For the NRRWBF algorithm, the calculation of the syndrome requires $(d_r - 1)M$ additions. The inversion function needs $(d_c - 1)N$ additions, Nd_c multiplications, N divisions, and $(d_r - 1)M$ comparison. So the number of additions will be $(d_c - 1)N + (d_r - 1)M$. The comparison is done at the initialization of the algorithm, unlike the other operation which is repeated at each iteration.

The operations for an iteration in WBF, RRWBF, and NRRWBF algorithms are grouped in Table 1.

The operations in the WBF, RRWBF, and NRRWBF algorithms for code 1 ($N = 600$ and $K = 450$) ($d_r = 12$, $d_c = 3$) are grouped in Table 2.

To see the difference between RRWBF and NRRWBF, we calculate E_1, for example, the error term value for the variable node 1, using the matrix H given previously.

For RRWBF, we find

$$\begin{aligned} E_1 = {} & ((2s_1 - 1)(|y_1| + |y_2| + |y_3| + |y_4| + |y_6| + |y_7|) \\ & + (2s_4 - 1)(|y_1| + |y_3| + |y_5| + |y_8| + |y_9| + |y_{10}|) \\ & + (2s_5 - 1)(|y_1| + |y_2| + |y_5| + |y_7| + |y_9| + |y_{10}|)) \frac{1}{|y_n|}. \end{aligned} \tag{15}$$

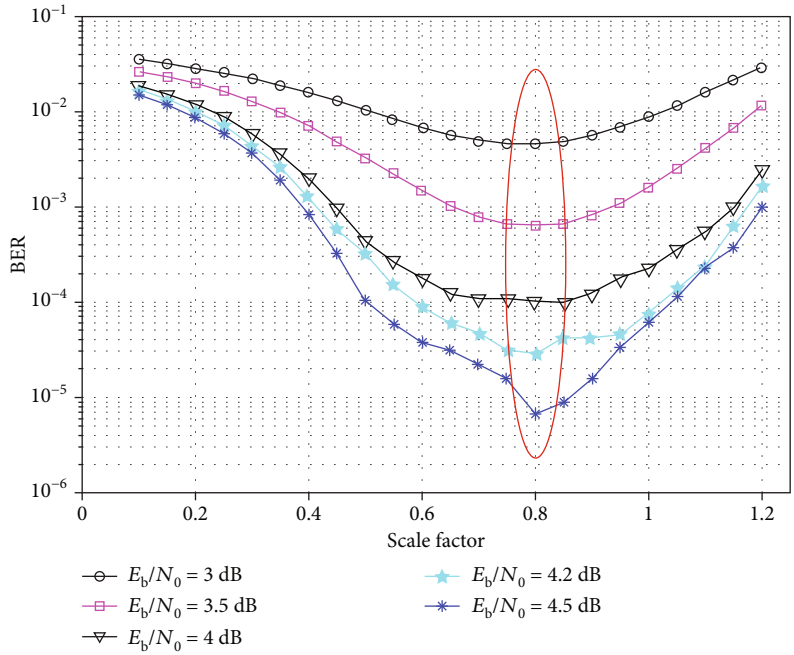

Figure 2: Normalization factor for code 1.

For NRRWBF, we find

$$E_1 = \left((2s_1 - 1)\left(6 \cdot y_1^{\min}\right) + (2s_4 - 1)\left(6 \cdot y_4^{\min}\right) + (2s_5 - 1)\left(6 \cdot y_5^{\min}\right)\right)\frac{1}{|y_n|},$$
$$E_1 = \left((2s_1 - 1)\left(y_1^{\min}\right) + (2s_4 - 1)\left(y_4^{\min}\right) + (2s_5 - 1)\left(y_5^{\min}\right)\right)\frac{6}{|y_n|}. \quad (16)$$

This is due to the use of the minimum value $y_m^{\min}$ of messages related to the control node m in relation (14), instead of the sum of messages related to the control node m in relation (13); this considerably reduces calculations, thus reducing the complexity of the decoder.

5. The Improvement of the New Reliability Ratio Weighted Bit Flipping Decoding Algorithm

During the LDPC decoding process, the decoder may fall into an infinite decoding loop before reaching the predefined maximum iteration number limit. In other words, the decoder may sometimes reverse an already-corrected bit, and this can be repeated several times during the decoding steps. Then, the decoding loop will cause a decoding failure when the maximum number of iterations is reached. To avoid such a situation of the infinite loop, we will adopt a method that allows preventing this problem:

(1) This method is based on the calculation of the sum of the syndromes. As the syndrome is a vector of 0 and 1, if it contains only 0s, the sum of the syndromes is 0 and the estimated codeword is valid; otherwise, it is invalid

(2) In this algorithm, we propose to calculate the sum of the syndromes at each iteration i, after the decoding steps, and compare it with the sum of the syndromes at iteration $i-1$. If the sum of the syndromes decreases, then we exclude the inverse bit (correct bit) at the current iteration, and we put it in a predefined exclusion set Ω. Otherwise, we flip this bit a second time, and the decoding continues. Therefore, the sum of the syndromes can be used to detect the decoding loop, where an increase in the sum of the syndromes is treated as a decoding loop

The INRRWBF algorithm is summarized in Algorithm 2.

6. Simulation Results and Discussion

In this section, we introduce the simulation results found by MATLAB software, using BPSK and AWGN channels.

The parameters used in the simulations for the three codes are presented in Table 3.

To evaluate our NRRWBF algorithm, we need to compare it with algorithms from the two LDPC decoding families cited in the introduction. For the soft decision, the Min-Sum (MS) algorithm was chosen as a good variant of the original Sum-Product (SP) algorithm. For the hard decision, we chose the following algorithms, RRWBF and WBF.

The MS is characterized by a normalization factor noted as S. It belongs to the interval $0 < S < 1$, and it is

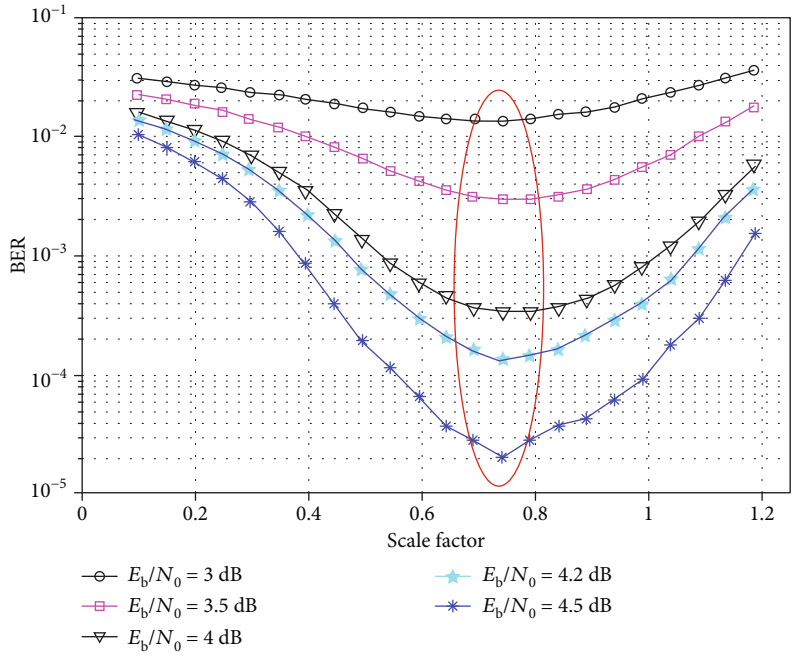

Figure 3: Normalization factor for code 2.

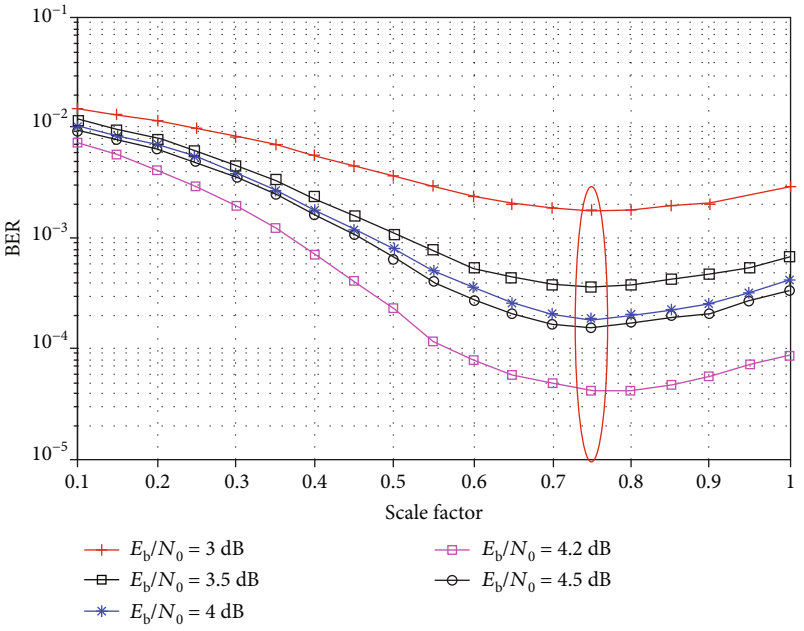

Figure 4: Normalization factor for code 3.

experimentally determined. This factor varies from one code to another.

Then, before running the simulation, we must find the normalization factor S for the MS algorithm for code 1, code 2, and code 3.

Figures 2–4 illustrate the obtained results for the three codes.

The optimum values of the normalization factor S for code 1, code 2, and code 3 are at different signal-to-noise ratios: 0.8, 0.75, and 0.74, respectively.

The simulation results are illustrated in Figures 5 and 6, comparing the NRRWBF algorithm with WBF, RRWBF, and MS.

The simulation results are illustrated in Figures 7, comparing the NRRWBF algorithm with WBF, RRWBF, ILCHWBF, and MS.

Figures 5–7 show the simulation results using a low-density matrix characterized by ($N = 600$, $K = 450$) ($d_c = 3$, $d_r = 12$) (code 1), ($N = 900$, $K = 750$) ($d_c = 3$, $d_r = 18$) (code 2), and ($N = 1200$, $K = 1050$) ($d_c = 3$, $d_r = 24$) (code 3),

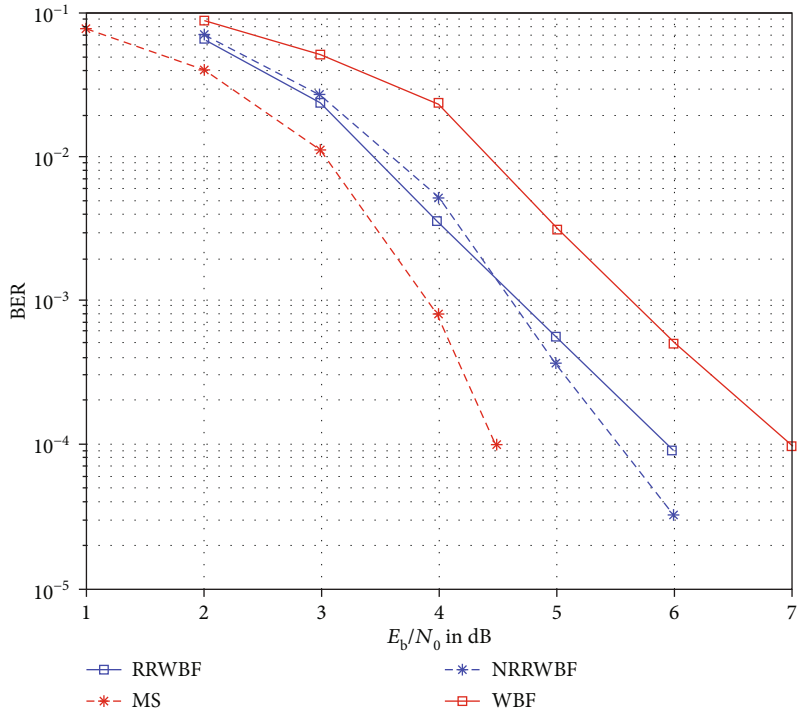

Figure 5: BER performance of different decoding algorithms for code 1.

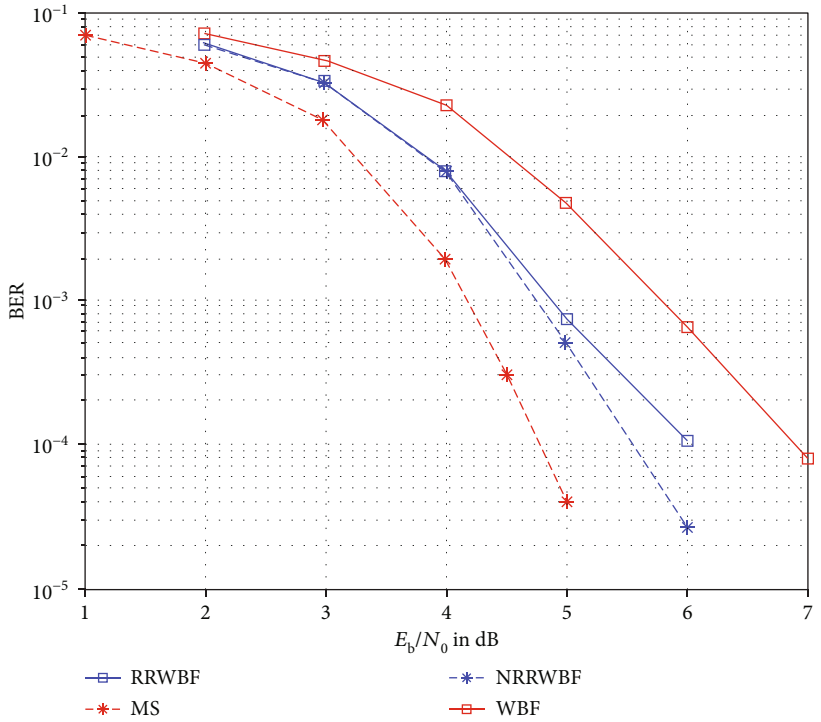

Figure 6: BER performance of different decoding algorithms for code 2.

respectively. We set the maximum number of iterations to $I_{max} = 100$ for each value of (E_b/N_0) dB for NRRWBF, ILCHWBF, RRWBF, and WBF; we also set the maximum number of iterations to $I_{max} = 6$ for MS.

Simulation results show that the NRRWBF algorithm gives the same performance as the RRWBF algorithm for the low SNR because the decoders in this interval do not distinguish between wrong and correct bits during the flipping

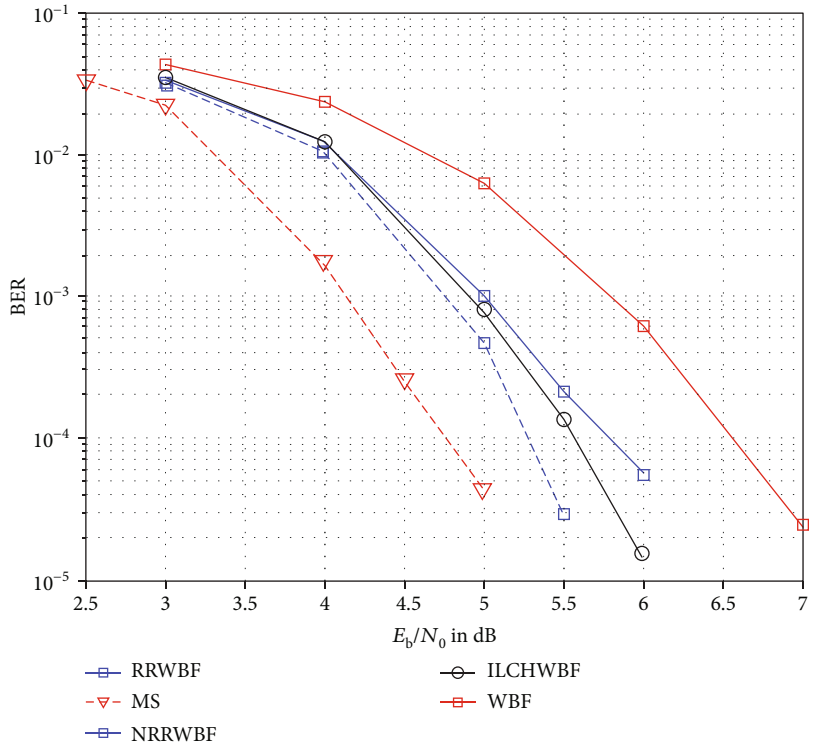

Figure 7: BER performance of different decoding algorithms for code 3.

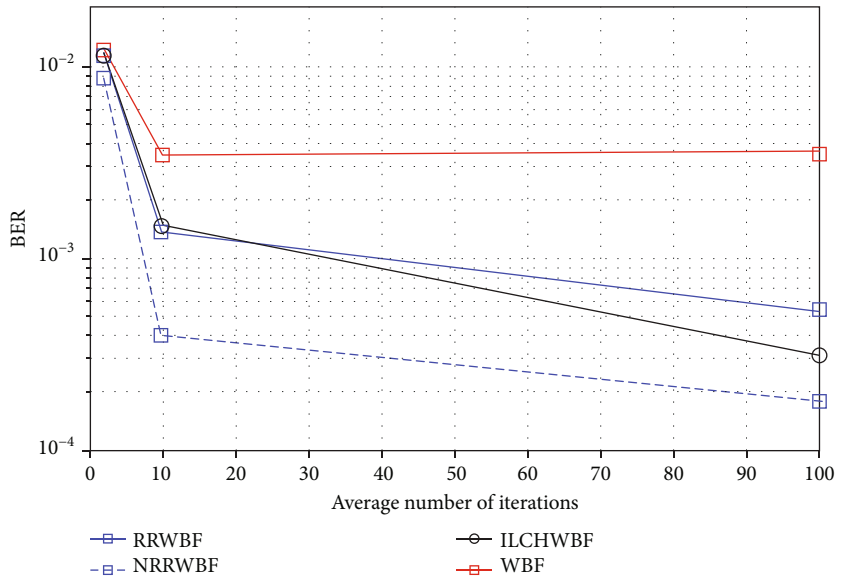

Figure 8: BER convergence comparison on code 1 at $E_b/N_0 = 5$ dB.

step as long as the noise is higher. In this same interval, we observe that our algorithm surpasses the WBF algorithm by 0.7 dB and presents a mediocrity in performance compared to MS by 0.8 dB.

For the loud signal-to-noise ratio, it is noted from Figure 5, for the 10^{-4} value of BER, that the NRRWBF algorithm surpasses the RRWBF algorithm by 0.3 dB and far by 1.1 dB of the MS algorithm.

From Figure 6, NRRWBF can be compared with RRWBF and MS for the 10^{-4} BER value; it is found that the NRRWBF algorithm surpasses the RRWBF algorithm by 0.5 dB and far by 0.8 dB of the MS algorithm.

From Figure 7, NRRWBF can be compared with RRWBF, ILCHWBF, and MS for the 10^{-4} BER value. It is found that the NRRWBF algorithm surpasses both the RRWBF algorithm by 0.6 dB and the ILCHWBF algorithm

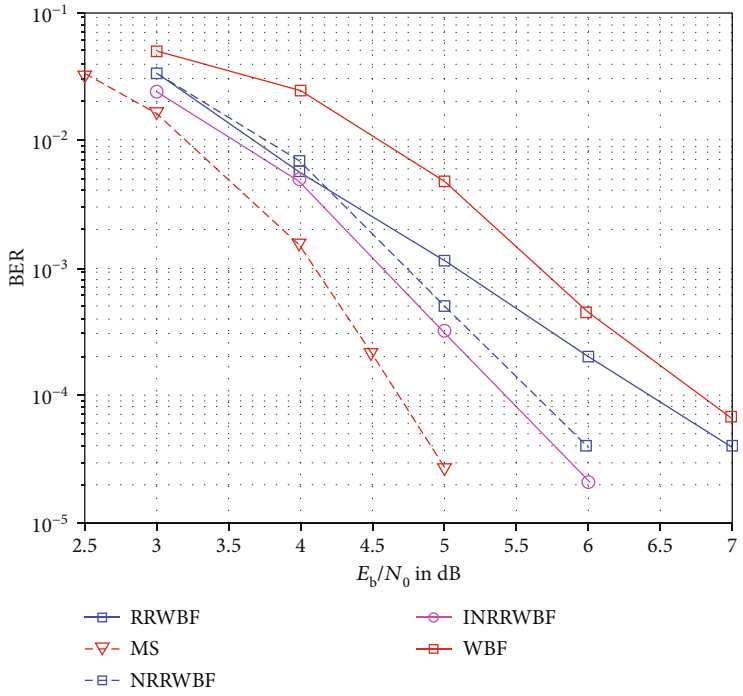

Figure 9: BER performance of different decoding algorithms for code 2.

by 0.3 dB, but it is lower by 0.7 dB compared to the MS algorithm.

All simulations reveal the good performance for our algorithm compared to the WBF algorithm.

Results indicate that there is a critical performance improvement when using the large line weight $d_r = 24$ versus $d_r = 12$, which also shows that the performance is important when the control matrix size is large.

These results are due to the importance of using the minimum of the absolute value of the received values by variable nodes for each control node in the inversion function. This makes it possible to determine the least reliable bit, which increases the probability of determining exactly the erroneous bit at each iteration so that it can be corrected. This method is more effective when the size of the codeword N and the row weight d_r are large.

Figure 8 shows the BER for different algorithms as a function of the maximum number of iterations (MNI) at $E_b/N_0 = 5$ dB. The result shows that our algorithm is almost ten times faster than the others ($7 \cdot 10^{-4}$ for 8 iterations instead of 75 for ILCHWBF and 100 for RRWBF).

In addition, for the same number of iterations (100 in this case), the BER in our case is better.

In Figure 9, we show the simulation results using a low-density matrix characterized by $(N = 900,\ K = 750)$ $(d_c = 3,\ d_r = 18)$ for the second code (code 2). We set the maximum number of iterations to $I_{max} = 100$ for each value of (E_b/N_0) dB for INRRWBF, NRRWBF, RRWBF, and WBF. For MS, we set the maximum number of iterations to $I_{max} = 6$.

We observe that there is a 0.25 dB performance gain at a 10^{-4} BER for the INRRWBF compared to the NRRWBF. This shows firstly the importance of using the sum of the syndromes as a decoding loop detector and secondly that there is an addition of performance compared to the NRRWBF algorithm, which mainly comes down to the sum of the syndromes as a criterion to avoid the infinite loop. Therefore, in this way, we have to take advantage of all the efficiency of our first algorithm to correct the errors found during the decoding step.

7. Conclusions

In this article, we have, in a first step, proposed the NRRWBF algorithm to improve the performance of LDPC codes. The simulation results show that this algorithm gives good performance compared to the RRWBF algorithm at a high SNR for different line weights, with a significant reduction in complexity. In a second step, we present an amelioration of the proposed algorithm called the INRRWBF algorithm. This proposition has increased performance thanks to the criterion of avoiding the infinite loop which results in greatly enhancing the efficiency of the decoder.

Data Availability

The data used in our work are declared in the manuscript, and the matrices used in the simulations to evaluate our algorithm against the algorithms found in the literature are available from us.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] R. G. Gallager, "Low-density parity check codes," *IEEE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962.

[2] 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. 1645–1946, 1996.

[3] Sae-Young Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Communications Letters*, vol. 5, no. 2, pp. 58–60, 2001.

[4] E. Chen, J. L. Koslov, V. Mignone, and J. Santoru, "DVB-S2 Backward-compatible modes: a bridge between the present and the future," *International Journal of Satellite Communications and Networking*, vol. 22, pp. 341–365, 2004.

[5] H. Singh, S. K. Yong, J. Oh, and C. Ngo, "Principles of IEEE 802.15.3c: Multi-Gigabit Millimeter-Wave Wireless PAN," in *Proceedings of 18th International Conference on Computer Communications and Networks*, San Francisco, CA, USA, 2009.

[6] Y. Xiao, "IEEE 802.11N: enhancements for higher throughput in wireless LANs," *IEEE Wireless Communications*, vol. 12, pp. 82–91, 2006.

[7] C. H. Liu, S. W. Yen, C. L. Chen et al., "An LDPC Decoder Chip Based on Self-Routing Network for IEEE 802.16e Applications," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 3, pp. 684–694, 2008.

[8] A. Ghosh, R. Ratasuk, B. Mondal, N. Mangalvedhe, and T. Thomas, "LTE-advanced: next-generation wireless broadband technology," *IEEE Wireless Communications*, vol. 17, no. 3, pp. 10–22, 2010.

[9] H. Li, J. Guo, C. Guo, and D. Wang, "A low-complexity min-sum decoding algorithm for LDPC codes," in *2017 IEEE 17th International Conference on Communication Technology (ICCT)*, Chengdu, China, October 2017.

[10] J. H. Lee and M. H. Sunwoo, "Low-complexity high-throughput bit-wise LDPC decoder," *Journal of Signal Processing Systems*, vol. 91, no. 8, pp. 855–862, 2019.

[11] Z. R. M. Hajiyat, A. Sali, M. Mokhtar, and F. Hashim, "Channel coding scheme for 5G mobile communication system for short length message transmission," *Wireless Personal Communications*, vol. 106, no. 2, pp. 377–400, 2019.

[12] M. P. C. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity-check codes based on belief propagation," *IEEE Transactions on Communications*, vol. 47, no. 5, pp. 673–680, 1999.

[13] J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier, and X. Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Transactions on Communications*, vol. 53, no. 8, pp. 1288–1299, 2005.

[14] J. Zhao, F. Zarkeshvari, and A. H. Banihashemi, "On implementation of min-sum algorithm and its modifications for decoding low-density parity-check (LDPC) codes," *IEEE Transactions on Communications*, vol. 53, no. 4, pp. 549–554, 2005.

[15] S. Hemati, F. Leduc-Primeau, and W. J. Gross, "A relaxed min-sum LDPC decoder with simplified check nodes," *IEEE Communications Letters*, vol. 20, no. 3, pp. 422–425, 2016.

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

[17] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity-check codes based on finite geometries: a rediscovery and new results," *IEEE Transactions on Information Theory*, vol. 47, no. 7, pp. 2711–2736, 2001.

[18] J. Zhang and M. P. C. Fossorier, "A modified weighted bit-flipping decoding of low-density parity-check codes," *IEEE Communications Letters*, vol. 8, no. 3, pp. 165–167, 2004.

[19] Ming Jiang, Chunming Zhao, Zhihua Shi, and Yu Chen, "An improvement on the modified weighted bit flipping decoding algorithm for LDPC codes," *IEEE Communications Letters*, vol. 9, no. 9, pp. 814–816, 2005.

[20] F. Guo and L. Hanzo, "Reliability ratio based weighted bit-flipping decoding for LDPC codes," *IEEE Communications Letters*, vol. 40, pp. 1356–1358, 2004.

[21] M. K. Roberts and R. Jayabalan, "An improved low complex hybrid weighted bit-flipping algorithm for LDPC codes," *Wireless Personal Communications*, vol. 82, no. 1, pp. 327–339, 2015.

[22] H. Huang, Y. Wang, and G. Wei, "Mixed modified weighted bit-flipping decoding of low-density parity-check codes," *IET Communications*, vol. 9, no. 2, pp. 283–290, 2015.

[23] D. Harita and K. P. Rajan, "Reliability variance based weighted bit flipping algorithms for LDPC," *IEEE Transactions on Communications*, pp. 593–595, 2019.

[24] M. Velmurugan and K. Pargunarajan, "Modified weighted bit flipping algorithm based on intrinsic information for LDPC codes," *IEEE Transactions on Communications*, 2017.

[25] S. Elsanadily, A. Mahran, and O. Elghandour, "Classification-based algorithm for bit-flipping decoding of GLDPC codes over AWGN channels," *IEEE Communications Letters*, vol. 22, no. 8, pp. 1520–1523, 2018.

[26] J. Oh and J. Ha, "A two-bit weighted bit-flipping decoding algorithm for LDPC codes," *IEEE Communications Letters*, vol. 22, no. 5, pp. 874–877, 2018.

[27] Y. Wang and G. Wu, "Cyclic switching weighted bit-flipping decoding for low-density parity-check codes," *IET Communications*, vol. 12, no. 3, pp. 271–275, 2018.

[28] Y. Liu and M. Zhang, "Hard-decision bit-flipping decoder based on adaptive bit-local threshold for LDPC codes," *IEEE Communications Letters*, vol. 2, no. 1, 2019.

[29] S. Kalipatnapu and I. Chakrabarti, "High-throughput bit flipping decoder for structured LDPC codes," *IET Communications*, vol. 13, no. 14, pp. 2168–2172, 2019.

[30] T. C. Y. Chang, P. H. Wang, and Y. T. Su, "Multi-stage bit-flipping decoding algorithms for LDPC codes," *IEEE Communications Letters*, vol. 23, no. 9, pp. 1524–1528, 2019, 23.