Bootstrapped Low Complexity Iterative Decoding Algorithm for Low Density Parity Check (LDPC) Codes

Albashir Mohamed[1], Maha Elsabrouty[1], and Salwa El-Ramly[2]

[1] Arab Academy for Science, Technology and Maritime Transport, Electronics and Comunications Engineering, 2033 Al Horriyah, Heliopolis, Cairo, Egypt
albashir.mohamed@staff.aast.edu, maha2000_eg@yahoo.com
[2] Ain Shams University, Faculty of Engineering, Electronics and Communications Engineering Dept., 11517 Abbassia, Cairo, Egypt
sramlye@netscape.net

Abstract. RRWBF (Reliability ratio based weighted bit-flipping) algorithm is one of the best hard decision decoding algorithms in performance. Recently several modifications are done to this technique either to improve performance or to lower the complexity. The IERRWBF (Implementation efficient reliability ratio based weighted bit-flipping) is developed targeting decreasing processing time of the decoding process. Low Complex IERRWBF (Low complex implementation efficient reliability ratio based weighted bit-flipping) algorithm is one of the latest algorithms targeting lowering decoding complexity, by decreasing number of iterations required to decode received code words and to solve the problem faced by IERRWBF which is the exponential increase in complexity as maximum number of iterations increases. In this paper we are targeting improving the performance of recent developed algorithm named Low Complex IERRWBF by adding a bootstrap step to the decoding technique which leads to increase in reliability of received bits then the number of decoded bits will be increased leading to improvement in performance.

Keywords: Bootstrapped Low Complex IERRWBF (Bootstrapped low complex implementation efficient reliability ratio based weighted bit-flipping); low density parity check (LDPC) codes; reliability ratio based weighted bit-flipping (RRWBF).

1 Introduction

Low density parity check (LDPC) coding is a prominent channel coding technique that has excellent performance approaching Shannon limit. It was proposed by Gallager in 1963 [1]. It has received increased attention and became a strong competitor to turbo codes as many digital communication systems adopted it as error control coding technique such as DVD-S2 [11].

LDPC codes can be decoded using various types of decoding techniques. There are three categories of decoding techniques for LDPC codes which are hard decision, soft

Y. Cai et al. (Eds.): Mobilware 2010, LNICST 48, pp. 377–389, 2010.

decision and hybrid decision decoding techniques. The focus of this paper is on hard decision decoding techniques. Another naming of the hard-decision decoding is the bit-flipping family which was first presented in [1]. Modifications to the basic bit-flipping began by the WBF (Weighted Bit-Flipping algorithm) [2], which targeted to increase the performance of the basic bit-flipping algorithm, through a measure of the reliability of received symbols in their decoding algorithm.

Another modification was done for further improvement in the performance which is the MWBF (Modified Weighted Bit-Flipping algorithm) [3], which includes a weighting factor α in the weighted check sum equation of the WBF (Weighted Bit Flipping algorithm) to include the intrinsic message for each bit in this equation, the value of α is determined by simulation for every SNR and column weight of each matrix. To solve the problem of predetermining α for each SNR and column weight of each matrix, RRWBF (reliability ratio based weighted bit-flipping algorithm) was proposed in [4] to solve this problem and also to increase the performance. The main drawback in RRWBF is the long processing time, so IERRWBF (implementation efficient reliability ratio based weighted bit-flipping algorithm) was proposed in [5] to solve this problem as optimization was applied to the technique and a reduction in the processing time was accomplished. Another modification done to IERRWBF to reduce the complexity without affecting the performance which led to the Low Complex IERRWBF proposed in [6]. The Low Complex IERRWBF is based on observing the check node step, and for certain number of repeated syndrome words the decoding is ended declaring failed decoding rather than wasting more computational power if decoding is continued.

In this paper, a modification is done on Low Complex IERRWBF using a bootstrap step following the work done in [7] targeting increasing the performance of Low Complex IERRWBF to be one of the best hard decision decoding techniques which will be named bootstrapped low complex implementation efficient reliability ratio based weighted bit-flipping.

The rest of the paper is organized as follows. Section (2) presents the background of bit flipping algorithms especially the Low Complex IERRWBF. Section (3) presents the modification done to the Low Complex IERRWBF. Section (4) presents the results obtained compared to the original work in [6]. Section (5) illustrates conclusion and future work.

2 A Brief Review on Bit-Flipping Algorithms

The bit-flipping algorithms started with the work of Gallager in his PhD [1] which contains the basic bit-flipping algorithm. A lot of modifications done to the basic bit-flipping algorithm targeting lowering complexity or contributing performance, these algorithms will be illustrated in details in the following sections starting with the basic bit-flipping algorithm.

The following table summarizes the basic variables used in the bit flipping algorithm.

Table 1. Symbols used in algorithms

Symbol	Definition
$\mathbf{H}_{mn}$	m^{th} row, n^{th} column of parity-check matrix **H**
r_n	n^{th} bit received from the channel
z_m	Hard decision of r_m
s_m	Syndrome of hard decision bit z_m
E_n	weighted check sum that is orthogonal on the code bit *n*
R_{mn}	Reliability Ratio
N (m)	The set of variable nodes that participate in m^{th} check node
M (n)	The set of check nodes in which n^{th} variable node participates

2.1 Basic Bit-Flipping Algorithm

The Basic bit-flipping algorithm [1] is the lowest in complexity compared to its variants and the simplest to be implemented, the algorithm procedures will be illustrated in the following steps:

Step 1: Compute the parity-check sums using eq. (1). If all parity-check equations are satisfied, stop the decoding.

$$s_m = \sum_{n=1}^{N} z_m \mathrm{H}_{mn} \ . \tag{1}$$

Step 2: Find the number of unsatisfied parity check equations for each code bit position denoted as f_i where $i = 0,1,.....n-1$.
Step 3: Identify the set Ω of bits where f_i is the largest.
Step 4: Flip the bits in Ω.
Step 5: Repeat steps1 to 4 until all parity-check equations are satisfied or a maximum number of iterations is reached.

2.2 Weighted Bit-Flipping Algorithm

The simple bit-flipping can be improved to achieve better error performance by including some kind of reliability information of the received symbols in their decoding algorithm [2]. However, more additional decoding complexity is required to achieve such improvement in the performance.

The algorithm is mainly based on computing weighted check sum using eq. (2) to determine which bit will be flipped.

$$E_n = \sum_{s_m^{(l)} \in S_n} (2s_m^{(l)} - 1) \mid y_m \mid_{\min}^{(l)} . \tag{2}$$

The algorithm is briefly discussed through the following steps:

Step 1: Compute the check sums. If all the parity-check equations are satisfied, stop the decoding.
Step 2: Compute E_n based on eq. 2, for $0 \leq l \leq n-1$.
Step 3: Find the bit position l for which E_n is the largest.
Step 4: Flip the bit z_n.
Step 5: Repeat steps1 to 4 until all parity-check equations are satisfied or a maximum number of iterations is reached.

2.3 Modified Weighted Bit-Flipping Algorithm

After observing the weighted bit-flipping algorithm and its limited performance, the modified weighted bit-flipping algorithm improves performance better than the weighted bit-flipping algorithm [3]. The algorithm is based on considering the check constraint messages and the intrinsic message for each bit.

$$E_n = \sum_{m \in M(n)} (2s_m - 1) \mid y \mid_{\min} - \alpha \mid y_n \mid \tag{3}$$

The weighting factor α is a real number and $\alpha \geq 0$. When $\alpha = 0$, the modified weighted bit-flipping is converted into standard weighted bit-flipping algorithm. For a given LDPC code, the optimal α at a given SNR may be defined as the value for which the modified weighted bit-flipping algorithm generates the smallest BER for decoding this LDPC code at that SNR. The optimal value of α at each SNR for a given LDPC code can be determined through simulation.

The decoding procedures for this algorithm are the same as for WBF (weighted bit-flipping algorithm) except the weighted check sum equation used in determining which bit will be flipped.

2.4 Reliability Ratio Based Weighted Bit-Flipping Algorithm

The reliability ratio based weighted bit-flipping decoding for LDPC [4] performs very efficiently among the existing bit-flipping algorithms that have appeared. As it was shown before that weighted bit-flipping and modified weighted bit-flipping algorithms have some drawbacks. As for the weighted bit-flipping, it considers only the parity-node based information during the evaluation of the error term (*En*). The modified bit-flipping decoding algorithm performs better than the weighted bit-flipping algorithm since it considers both the check-node based and the message-node based information during calculation of error term (*En*). However, a drawback of the modified bit-flipping algorithm is its dependence on α, so it is required to find optimum α for each individual SNR. Both the weighted bit-flipping and modified weighted bit-flipping considers the specific check-node based information. However, all message nodes participating in the m^{th} parity check are contributing.

Hence, a new quantity is defined named as "Reliability Ratio" and is given by:

$$R_{mn} = \beta \frac{| y_n |}{| y_m^{\max} |}. \tag{4}$$

where $| y_m^{\max} |$ represents the highest soft magnitude of all message nodes participating in the m^{th} parity check. The factor β is the normalization factor to ensure that:

$$\sum_{n:n\in N(m)} R_{mn} = 1. \tag{5}$$

The error- term is calculated from:

$$E_n = \sum_{m\in M(n)} \frac{(2s_m - 1)}{R_{mn}}. \tag{6}$$

The decoding procedures is the same as modified bit-flipping except for non-computing of α due to using of reliability ratio instead of it to determine which bit is reliable and which one is non-reliable for the flipping process.

2.5 Implementation Efficient Reliability Ratio Based Weighted Bit-Flipping Algorithm

Reliability ratio based weighted bit-flipping algorithm is an efficient hard decision decoding algorithm but it has a main drawback which is the long decoding time taken by the algorithm. Implementation efficient reliability ratio based weighted bit-flipping was proposed in [5] to solve this problem by optimizing the algorithm to reduce the decoding time to be suitable for simulation and hardware implementation especially when the maximum number of iterations assigned for the algorithm is small.

The algorithm is divided into four steps: initialization step, variable node step, decision step and check node step. The operation done in each step is shown as follows:

Initialization step:

$$T_m = \sum_{n\in N(m)} | r_n |. \tag{7}$$

Variable node step:

$$E_n = \frac{1}{|r_n|} \sum_{m\in M(n)} (2s_m - 1)T_m. \tag{8}$$

Decision step: Flip the bit z_n for $\mathbf{n} = \arg\max_n \mathbf{E}_n$.

Check node step:

$$s_m = \sum_{n=1}^{N} z_m \mathrm{H}_{mn}. \tag{9}$$

2.6 Low Complex Implementation Efficient Reliability Ratio Based Weighted Bit-Flipping Algorithm

One of the drawbacks of the IERRWBF algorithm as stated in [5], which the author failed to solve is that the algorithm spends a larger percentage of the time at the variable node step and the check node step. As the maximum iteration number assigned for decoding increases, delay increases without any significant improvement in the performance.

In case of channels with low Signal-to-Noise Ratio (SNR) having AWGN, the hard-decision based bit-flipping algorithms sometimes fails to decode the received codeword correctly, as the decoding in such a case causes the syndrome vector to be a non-zero vector, which leads to failed decoding of the received codeword. Even with large iteration number assigned for the algorithm (500 iterations and more), there will be no significant improvement in the error performance. It has been observed that in such low SNR the syndrome vector will continue to be a non-zero vector and the decoding will keep flipping endlessly consuming computational power with no improvement throughput. The low complex implementation efficient reliability ratio based weighted bit-flipping decoding algorithm is based on the original scheme described as in [5]. The initialization, decision and variable node steps are the same as the original algorithm. However, the modification is in the check node step. Such a modification has a great effect on significant reduction of the decoding time.

The check node step used in all bit-flipping algorithms is a mere syndrome check condition that checks if the decoded codeword is a valid codeword or not. If the syndrome vector is all-zero vector, then the decoded codeword is a valid codeword and if the syndrome vector is not all-zero vector, then the decoded codeword is not decoded correctly and the algorithm continues till the syndrome becomes all-zero vector or the maximum number of iterations is achieved without fulfilling the syndrome condition. As explained, such a condition is rendered in case of low SNR. However, what is required in such cases is to try decreasing the number of iterations required at each SNR for decoding any received codeword to a limited number of iterations as extending the algorithm to more number of iterations will not achieve any observable or enhanced performance. This is done by adding an extra conditional step to examine such situations and control the iteration loop by deciding whether to continue decoding or exit the iteration loop and stop the algorithm to have a final decoded codeword. So a mechanism of the added condition will be started by obtaining syndrome vector at the end of each iteration and be stored in 3-entry register. The register is chosen to be of minimum size equal to 3 precisely because we need at least 2 iterations to get the same decoded codeword and the same syndrome vector, starting from initial vector, if the same bit in the decoded codeword is being flipped two times to return to the initial state, so the 3 entries correspond to the initial syndrome (initial vector) in the register, the syndrome vector after first iteration and the syndrome vector after second iteration. So, size 3 is the minimum size of register that could be used for storing syndrome vector for comparing between the first and the third (last one) where each new entry is stored at the top of the register and the remaining entries are shifted down to remove the last entry. So when the point of oscillation is reached which means that no contribution to performance will occur while continuing to the maximum number of iterations, so decoding failure is declared and decoding is halted, this added condition significantly reduces the complexity without any effect on performance compared to IERRWBF algorithm.

3 Bootstrapped Low Complex Implementation Efficient Reliability Ratio Based Weighted Bit-Flipping Algorithm

Low complex implementation efficient reliability ratio based weighted bit-flipping is the last modification done to reliability ratio based weighted bit-flipping targeting lowering complexity. Our goal is to increase the performance of low complex implementation efficient reliability ratio based weighted bit-flipping which will be in this case a very efficient hard decision decoding technique.

The idea of the algorithm comes from the work done in [7]. In [7] a modification to the weighted bit-flipping algorithm is done by adding a bootstrap step to the decoding algorithm which leads to significant contribution in performance. Bootstrap step will firstly be discussed for further explanation of the idea of using bootstrap step to contribute to performance.

Before explaining the bootstrap step, we need to make a few definitions: we call a received value and its corresponding variable node "unreliable" if $| y_n | < \alpha$, for some predetermined value α, and "reliable" otherwise. A check node is referred to as "reliable" with respect to an unreliable variable node if all the other variable nodes connected to that check node are reliable.

We initiate the decoding process by identifying and erasing all the unreliable variable nodes. We then assign improved values and reliabilities to the erased bits by passing them messages from the reliable variable nodes through the reliable check nodes. The new value y'_n that substitutes y_n for an erased variable is computed as

$$y'_n = y_n + \sum_{m \in M_r(n)} \Big(\prod_{n' \in N(m) \setminus n} \operatorname{sgn}(y_{n'}) \Big) \min_{n' \in N(m) \setminus n} |y'_n| . \tag{10}$$

where $M_r(n)$ denotes the set of the reliable check nodes connected to the unreliable bit node n, all this operation is illustrated using fig. 1. If there is no reliable check node connected to an unreliable bit node, that node keeps the original received value y_n. The algorithm then proceeds with the conventional algorithm using the improved value y'_n.

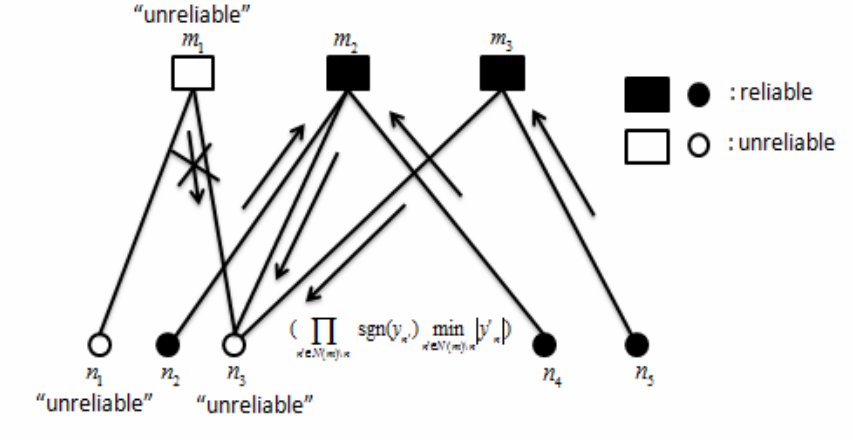

Fig. 1. Bootstrap decoding procedure over tanner graph

After studying and observing the bootstrap step in the bootstrapped low complex implementation efficient reliability ratio based weighted bit-flipping it was found that it combines horizontal step with vertical step of min-sum algorithm in one equation to be processed individually on the unreliable bits predetermined using certain threshold (α).

The embedding of bootstrap step in the weighted bit-flipping decoding algorithm leads to significant increase in reliability of received codeword that will lead to increase in performance. Our idea is to do the same for low complex implementation efficient reliability ratio based weighted bit-flipping algorithm to increase its performance, resulting a very good hard decision decoding technique that have two main properties; low complexity and high performance, then new algorithm is created

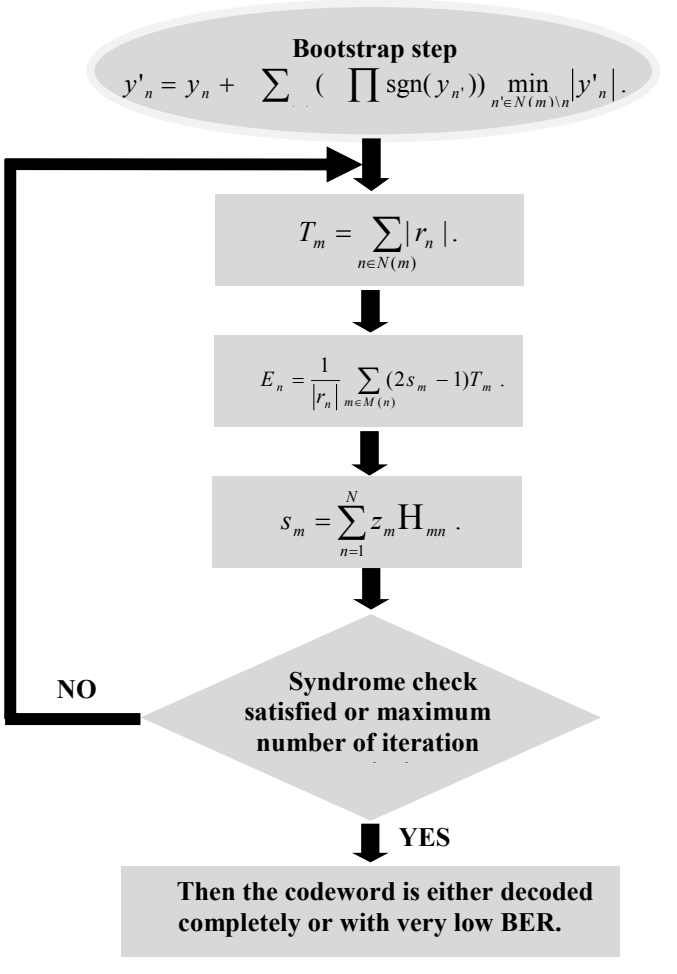

Fig. 2. Flow chart of Bootstrapped Low Complex IERRWBF decoding algorithm

which is called bootstrapped low complex implementation efficient reliability ratio based weighted bit-flipping. After processing on the bootstrap step the Low Complex IEERRWBF algorithm begins decoding the improved received codeword as mentioned in the last section, Also the whole decoding process is described in fig. 2 that contains flow chart of the new algorithm.

4 Simulation Results

The simulations are performed using *(n,k,v)* regular LDPC codes where *n* stands for the codeword length, *k* is the message length, and *v* is the number of '1's per column in the parity check matrix **H.** Progressive edge growth (PEG) and Gallager LDPC codes were used as the construction methods for generating the parity check matrix **H** of these regular LDPC codes. The regular LDPC codes used in the simulation were PEG (504,252,3), Gallager (504,252,3) and Gallager (204,102,3) by decoding each LDPC code using WBF, LLR-BP, IERRWBF algorithm presented in [5], Low Complex IERRWBF algorithm presented in [6] and the proposed algorithm in this paper. All codes used are of rate ½. The coded data are then modulated using a BPSK modulation scheme and sent over a channel having AWGN to simulate the effect of real wireless channel. The simulations were run on MATLAB.

The figures show the observable contribution in error performance. Fig. 3 shows the performance of decoding PEG (504, 252, 3) LDPC codes with 25 maximum iterations using WBF, Low Complex IERRWBF and Bootstrapped Low Complex IERRWBF. Also 15 maximum iterations assigned for Bootstrapped Low Complex IERRWBF for further proof of superiority of the new algorithm. Only 5 maximum iterations are assigned for LLR-BP. It is clear that the performances of the LLR-BP and the Bootstrapped Low Complex IERRWBF have the best performance. The high performance of the new technique is due to the insertion of bootstrap step, which combines vertical step with horizontal step of min-sum algorithm, which increases reliability of received code word leading to high performance as shown in the fig. 3.

Fig. 4 shows the performance of Gallager (504,252,3) LDPC codes decoded with 25 maximum iterations using WBF, Low Complex IERRWBF and Bootstrapped Low Complex IERRWBF. Also 15 maximum iterations assigned for Bootstrapped Low Complex IERRWBF for further proof of superiority of the new algorithm. Only 5 maximum iterations are assigned for LLR-BP. From the results the performance of Gallager code is inferior to that of the PEG code, but still the new algorithm is superior over the original work which is low complex implementation efficient reliability ratio based bit-flipping.

Fig. 5 shows the performance of Gallager (204,102,3) regular LDPC code when decoded with Low Complex IERRWBF and Bootstrapped Low Complex IERRWBF for 5 maximum iterations. It is clear that the performance of Bootstrapped Low Complex IERRWBF is better than that of the Low Complex IERRWBF at the same number of iterations. With increasing number of iterations for Bootstrapped Low Complex IERRWBF decoding to 25 iterations, the difference in performances between the Low Complex IERRWBF and Bootstrapped Low Complex IERRWBF is significantly increased. This shows that the Bootstrapped Low Complex IERRWBF decoding is better than that of the Low Complex IERRWBF decoding in terms of performance.

Table 2 shows the decoding time of IERRWBF, Low Complex IERRWBF and Bootstrapped Low Complex IERRWBF using machine with Intel® core™ 2 Duo T7200 @ 2 GHz with 2000 MB memory and 32-bit operating system is used. The results show that the proposed algorithm has extremely reduced complexity compared with original algorithm IERRWBF. As shown the complexity of Low Complex IERRWBF has the lowest algorithm but is inferior in performance compared by proposed algorithm as shown in fig. 2, fig. 3 and fig. 4. So the proposed algorithm is superior in performance with complexity close to Low Complex IERRWBF which is massively reduced compared with IERRWBF algorithm.

According to the obtained results the new technique proves that it is superior over all existing hard decision decoding techniques. Also the new technique combines between low complexity as it is created from low complex technique which is low complex implementation efficient reliability ratio based weighted bit-flipping and high performance, due to using of bootstrap step that enhanced the received codeword, so at same Eb/No and same number of iterations performance increased as shown in the obtained results.

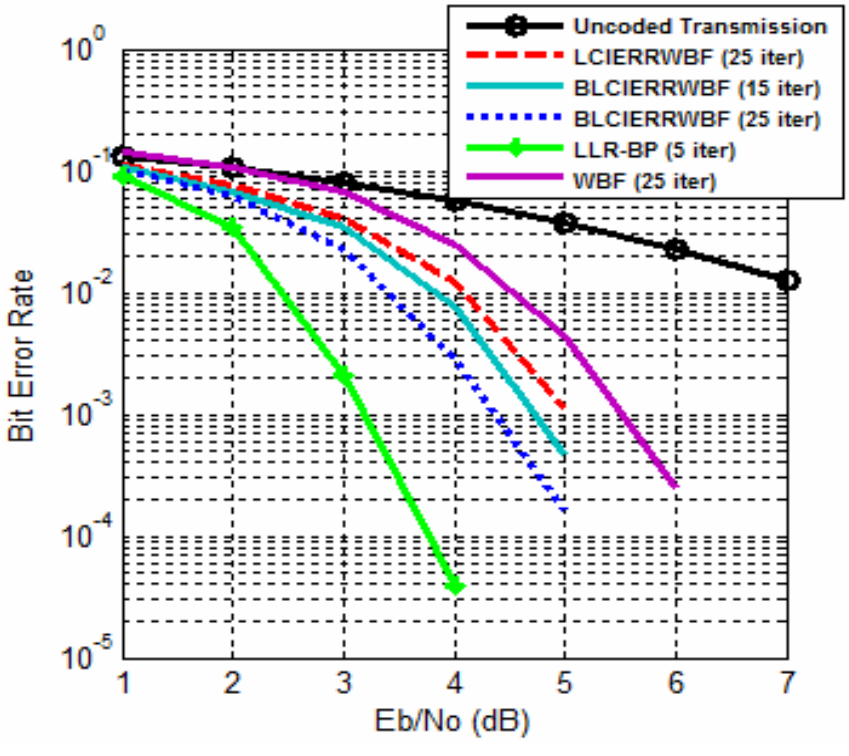

Fig. 3. BER for (504,252,3) PEG-LDPC code decoded by WBF, Low Complex IERRWBF, Bootstrapped Low Complex IERRWBF, and LLR-BP

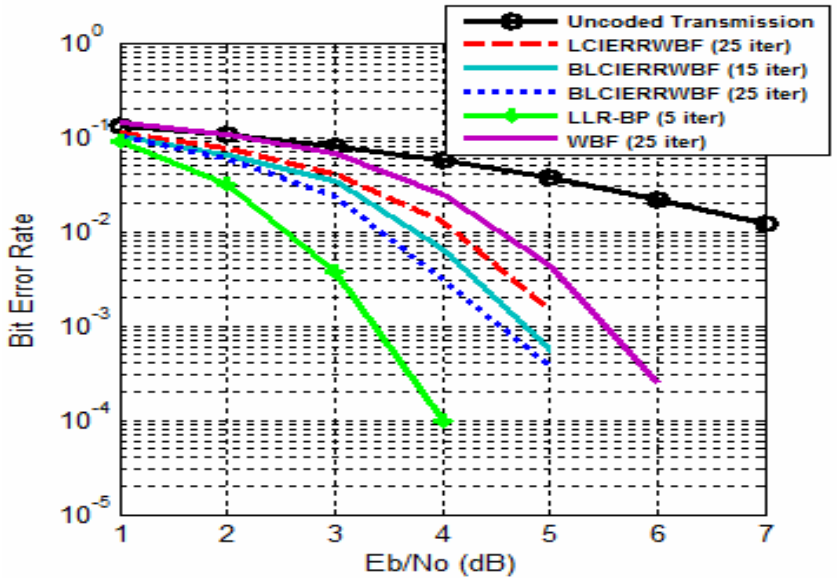

Fig. 4. BER for (504,252,3) Gallager LDPC code decoded by WBF, Low Complex IERRWBF, Bootstrapped Low Complex IERRWBF, and LLR-BP

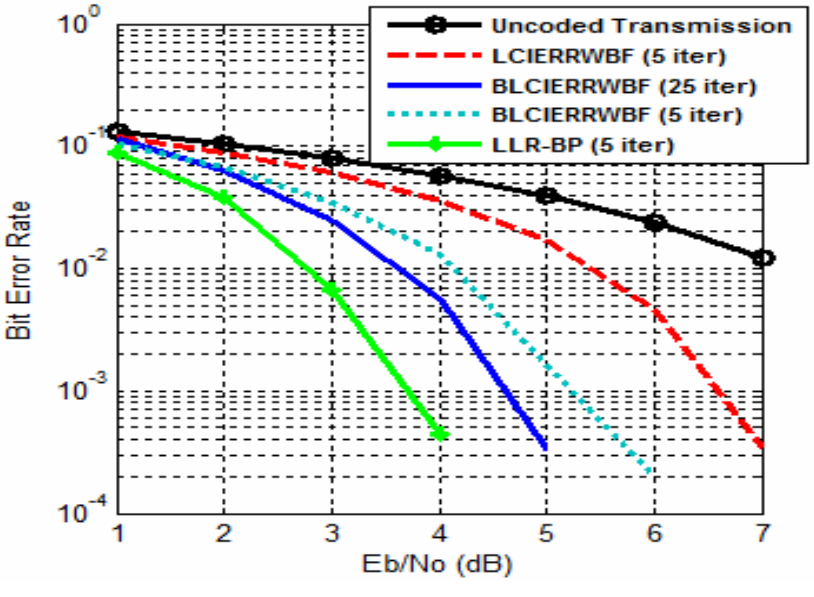

Fig. 5. BER for (204,102,3) Gallager LDPC code decoded by Low Complex IERRWBF, Bootstrapped Low Complex IERRWBF and LLR-BP

Table 2. Decoding time of presented decoding techniques compared with IERRWBF

SNR(dB)	1	2	3	4	5	6
IERRWBF	19.879	17.855	17.691	17.564	10.247	9.567
LCIERRWBF	0.0352	0.0256	0.0253	0.0252	0.0231	0.0212
BLCIERRWBF	0.2945	0.2643	0.2295	0.2063	0.1593	0.1375

5 Conclusion and Future Work

In this paper a new algorithm is created by modifying low complex implementation efficient reliability ratio based weighted bit-flipping by adding bootstrap step which leads to very good performance and very low complexity. The new algorithm is named bootstrapped low complex implementation efficient reliability ratio based weighted bit-flipping algorithm, which has very low complexity compared to IERRWBF and comparable complexity compared to low complex IERRWBF, also its performance is superior to IERRWBF and low complex IERRWBF. So the new algorithm has very low complexity and superior performance over hard decision decoding algorithms. As the performance of the proposed algorithm is superior over all hard decision decoding algorithms it is still inferior to the soft decision decoding algorithm represented by LLR-BP but it has very high complexity if it is compared to the proposed algorithm.

As for future work, while the new algorithms have very good performance and very low complexity it need more adjustments, where the value of α is obtained through simulation for each column weight and SNR, so before proceeding in decoding, α must be computed first. Then the new algorithms need a new mechanism instead of α to define the unreliable bits and the reliable ones to solve the problem of pre-computing of α before proceeding in decoding the received codeword. Also the complexity of the algorithms needs to be minimized, as the complexity of the algorithms is increased due to the insertion of bootstrap step which leads to more processing time as received codeword reliability increased. Then more adjustments to the algorithm are needed to minimize complexity.

Acknowledgment

The authors wish to thank D. J. MacKay for providing them with the parity-check matrices of the LDPC codes.

References

1. Gallager, R.G.: Low-Density Parity-Check Codes. MIT Press, Cambridge (1963)
2. Kou, Y., Lin, S., Fossorier, M.: Low density parity check codes based on finite geometries: A rediscovery and more. IEEE Trans. Inform. Theory 47, 2711–2736 (2001)

3. Zhang, J., Fossorier, M.P.C.: A modified weighted bit-flipping decoding of low-density parity-check codes. IEEE Communication Letters 8(3), 165–167 (2004)
4. Guo, F., Hanzo, L.: Reliability ratio based weighted bit-flipping decoding for low-density parity-check codes. Electronics Letters 40(21), 1356–1358 (2004)
5. Lee, C.-H., Wolf, W.: Implementation-efficient reliability ratio based weighted bit-flipping decoding for LDPC codes. Electronics Letters 41(13) (June 23, 2005)
6. Zeidan, H.R., Elsabrouty, M.M.: Low Complexity Iterative Decoding Algorithm for Low-Density Parity-Check (LDPC) codes. In: Wireless Days. WD 2008. 1st IFIP, pp. 1–5 (2008)
7. Nouh, A., Banihashemi, A.H.: Bootstrap decoding of low-density parity-check codes. IEEE Comrrtun. Len. 6(9), 391–393 (2002)
8. Kou, Y., Lin, S., Fossorier, M.P.C.: Low-density parity-check codes based on finite geometries: A rediscovery and new results. IEEE Trans. Inform. Theory 47, 2711–2736 (2001)
9. MacKay, D.J.: Encyclopedia of Sparse Graph Codes, `http://www.inference.phy.cam.ac.uk/mackay/codes/data.htm`
10. Inaba, Y., Ohtsuki, T.: Performance of Low Density Parity Check (LDPC) Bootstrap Decoding Algorithm on a Fast Fading Channel. In: IEEE Vehicular Technology Conference, pp. 333–337 (2004)
11. Digital video broadcasting (DVB); User guidelines for the second generation system for broadcasting, interactive services, news gathering and other broad-band satellite applications (DVB-S2). European Telecommunications Standards Institute (ETSI), TR 102 376