Journal of Advances in Computer Research
Quarterly pISSN: 2345-606x eISSN: 2345-6078
Sari Branch, Islamic Azad University, Sari, I.R.Iran
(Vol. 9, No. 1, February 2018), Pages: 85-100
www.jacr.iausari.ac.ir

Search Based Weighted Multi-Bit Flipping Algorithm for High-Performance Low-Complexity Decoding of LDPC Codes

Ehsan Olyaei Torshizi[1], Mohammad Amir Nazari Siahsar[1], Ali Akbar Khazaei[2], Hossein Sharifi[3]
1) Young Researchers and Elite Club, Mashhad Branch, Islamic Azad University, Mashhad, Iran
2) Department of Electrical Engineering, Mashhad Branch, Islamic Azad University, Mashhad, Iran
3) Department of Electrical & Computer Engineering, Shahid Beheshti University, Tehran, Iran
eh.olyaei@mshdiau.ac.ir; a.nazari.s@mshdiau.ac.ir; khazaei@mshdiau.ac.ir; sharifi@aryasatel.com

Received: 2017/03/07; Accepted: 2017/04/25

Abstract

In this paper, two new hybrid algorithms are proposed for decoding Low Density Parity Check (LDPC) codes. Original version of the proposed algorithms named Search Based Weighted Multi Bit Flipping (SBWMBF). The main idea of these algorithms is flipping variable multi bits in each iteration, change in which leads to the syndrome vector with least hamming weight. To achieve this, the proposed algorithms do multi-dimensional searching between all possible bit position(s) that could flip in each iteration to select the best choices. It goes without saying that each iterative decoding algorithm provides a distinct trade-off between complexity and performance. SBWMBF algorithm can flip several bits in each iteration. This ability causes a faster convergence rate and less hardware complexity compared to the other hybrid decoding algorithms. Then, in order to simplicity and reduction in run time of original version, we have proposed a simplified form. Simulation results, when compared to other known decoding algorithms, illustrate that two proposed algorithms are highly efficient. They converge significantly faster and have a tangible reduction in iteration number and computational complexity. Also the presented algorithms have superior performance but with little performance penalty than the robust BP algorithm.

Keywords: *Belief Propagation (BP), Low-Density Parity-Check (LDPC) Codes, Modified Weighted Bit-Flipping Algorithm, Hybrid Iterative Decoding, Error Performance*

1. Introduction

Low-Density Parity-Check (LDPC) codes were originally invented by Gallager [1], and brought back into prominence by Mackay and Neal [2]. It is a kind of binary linear block code whose parity check matrix is sparse which has much fewer 1's than a common matrix. A sparse parity check matrix facilitates simple decoding algorithms and low-complexity decoder designs. Parity check matrix of LDPC code often represent by a bipartite graph, called Tanner graph [2], which is composed of n variable nodes and m check nodes. Those variable nodes and check nodes are connected by edges defined by nonzero entries of matrix **H**. The number of 1s in each column of **H** determines the number of edges for each variable node connected to check nodes. Similarly the number of 1s in each row of **H** determines the connections from each check node to variable nodes. Tanner graph shows a clear view of all the information exchange links in a decoding process. Parity check matrix of a regular LDPC code with column weight $\omega_c = 3$ and row weight $\omega_r = 4$ along with its corresponding tanner graph shown in figure 1. In this figure the corresponding edges of first column and first row in parity check matrix **H** is shown by red and blue lines, respectively.

$$\mathbf{H} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

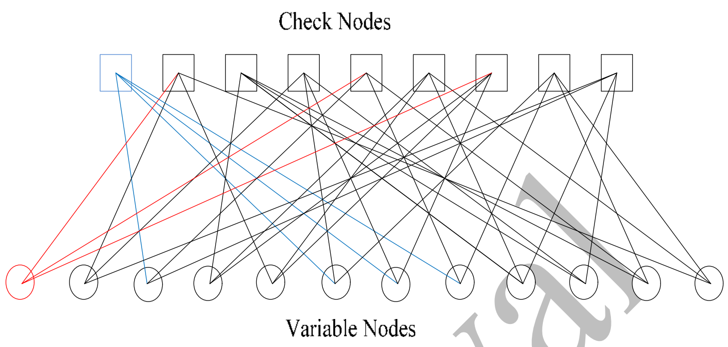

Figure 1. Example of a parity check matrix along with its corresponding Tanner graph

There have been wide variety methods for decoding of LDPC codes, ranging from low to high complexity and from reasonably good to very good performance. These decoding methods may be classified into three categories: hard-decision, soft-decision and hybrid decoding schemes. These decoding algorithms include: One-Step Majority-Logic (OSMLG) decoding [6,7], Gallager's bit flipping (BF) decoding [1], weighted MLG decoding (WMLG)[8], weighted BF decoding (WBF)[9], and other modified version of WBF algorithm like improved weighted bit flipping (IWBF)[10], Modified Weighted bit-flipping[11] and IVMWBF algorithm[25], A Posteriori Probability (APP) decoding[1], iterative decoding based on belief propagation (IDBP) (commonly known as sum-product algorithm (SPA))[12-15], and approximations of the IDBP like Min-Sum[16], Normalized / Offset-BP [17-19] algorithms. OSMLG and BF decoding belong to hard-decision decoding methods. APP, IDBP/SPA decoding and its approximations are soft-decision schemes. WBF, IWBF, Modified WBF and IVMWBF algorithms are hybrid decoding methods.

Each iterative decoding algorithm provides a distinct trade-off between complexity and performance. It goes without saying that the soft-decision algorithms in which message passing is used to carry out their decoding procedures can achieve better performance compared with those based on hard-decision. However, due to their higher complexity in terms of calculations and the hardware implementation of the relevant decoders, the use of these algorithms is unfeasible. The performance of hard-decision algorithms is approximately 3dB lower than that of soft-decision algorithms.

Hybrid decoding algorithms which are in fact based on both soft and hard decisions have been developed to bridge the performance gap between these two groups of algorithms. Results obtained from analytical simulations and studies indicate that there is still a wide performance gap between hybrid algorithms and those based on soft decision. The present work is an effort to propose a new high-performance low-complexity hybrid decoding algorithm. In order to achieve this, we shall first propose its original version of algorithm called the SBWMBF algorithm, which while having the capacity for flipping several bits per iteration. This algorithm offers a faster convergence rate and less hardware complexity compared to the modified WBF algorithm and the other version of hybrid algorithms. Then, in order to simplicity and reduction in run time of original version, we will introduce a simplified version that is new and highly efficient. Also this algorithm has acceptable performance compared with the BP algorithm and less complexity and fewer iterations required. The remainder of the paper is organized as follows: Section 2 presents notations, definitions and the proposed algorithms. Section 3 provides some simulation results to illustrate the performance of the two proposed algorithms. We discuss about simulation results on section 4 and finally Section 5 concludes this paper.

2. Decoding Algorithms

2.1 Notation and Basic Definition

Suppose that a binary LDPC code (N,K) for error control is used over a channel with binary input and an additive white Gaussian noise (BI-AWGN) with a mean value of 0 and a power spectral density of $N_0/2$. In addition, the signaling used here is BPSK with unit energy. A code-word $\mathbf{c} = (c_1, c_2, \dots, c_N) \in \{GF(2)\}^N$ is mapped before transmission into the $\mathbf{x} = (x_1, x_2, \dots, x_N)$ bipolar sequence, where $x_i = 2c_i - 1, 1 \le i \le N$. Furthermore, $\boldsymbol{y} = (y_1, y_2, \dots, y_N)$ is defined as the received sequence based on soft decision in the output of the receiver matched filter. For $1 \le i \le N$, $y_i = x_i + n_i$, where n_i is a gaussian random variable with a mean value of 0 and a variance of $N_0/2$. An initial binary sequence based on the hard decision used for the received sequence $\mathbf{z}^{(0)} = (z_1^{(0)}, z_2^{(0)}, \dots, z_N^{(0)})$ is determined as follows:

$$z_i^{(0)} = \begin{cases} 1, & if\ y_i \ge 0 \\ 0, & if\ y_i < 0 \end{cases} \tag{1}$$

For each sequence based on the hard decision $\mathbf{z}$ built up at the end of each iteration, the syndrome vector can be calculated as follows:

$$\mathbf{s} = \mathbf{z}.\mathbf{H}^T \tag{2}$$

For each channel output, the Log-Likelihood-Ratio (LLR) is defined using the following equation:

$$L_i \triangleq \ln \frac{P(c_i = 1 | y_i)}{P(c_i = 0 | y_i)} = \frac{4 y_i}{N_0} \tag{3}$$

The absolute value of L_i, $|L_i|$, is defined as the reliability of the initial decision $z_i^{(0)}$. Let's define $HW(\mathbf{v})$ as the hamming weight for each binary vector $\mathbf{v} = (v_1, v_2, \dots, v_N)$. Consider $\mathbf{u}_i$ denote as the N-dimensional unit vector, i.e., a vector with "1" at the ith position and "0" at other positions. We define $\mathbf{u}_{i_1,\dots,i_p}$ as sum of the p unit vector $\mathbf{u}_{i_1}, \dots, \mathbf{u}_{i_p}$. It is obvious that $\mathbf{u}_{i_1,\dots,i_p}$ is a vector with p "1" at $i_1, \dots, i_p$ positions and N-p "0" at the other locations.

$$\mathbf{u}_{i_1,\ldots,i_p} \triangleq \sum_{k\in\{i_1,\ldots,i_p\}} \mathbf{u}_k \tag{4}$$

2.2 Main Idea

Most existing algorithms for decoding LDPC codes are based on the assumption that the iterative algorithm can flip only one bit of the received sequence in each iteration. The Standard WBF decoding algorithm along its improved versions such as IWBF and Modified WBF algorithm are among these algorithms. Because of flipping only one single bit in each iteration, these algorithms have slow convergence rate, and require numerous iteration to reach an acceptable performance. Furthermore, there is still a significant difference between these algorithms and those based on soft decision in terms of performance.

In this paper, two new algorithms proposed for decoding of LDPC codes based on the idea of flipping several bits in each iteration. However, it should be born in mind that making mistakes in flipping even a single bit in each iteration extends the chain error to the whole algorithm, and ultimately leads to its divergence. Therefore, the need for using a high-precision mechanism for determining the number and position of the bits to be flipped in each iteration becomes obvious. The idea of flipping bits in parallel in each iteration has many advantages over flipping only one single bit in each iteration. These advantages include rapid increase of the convergence rate of the algorithm and a decrease in the number of iterations needed.

Algorithms in which bits are flipped in parallel, are capable of avoiding errors in iterative decoding prevalent in algorithms, particularly in lower SNRs. This is because limited number of iterations, although the frequency of such errors may be very low. For instance, consider the received sequence for decoding an LDPC code consists of 2000 bits, and the maximum number of iterations is 100. In this situation, iterative algorithm in which a single bit is flipped in each iteration will be capable of detecting and correcting 100 bits of errors from the received sequence in the best-case scenario. Therefore, the use of such algorithms for received sequences in which the number of errors is greater than the maximum number of iterations is practically unfeasible, because they cannot correct all the errors in such sequences. In the proposed SBWMBF algorithm, the following new equation is used to calculate the number of bits which need to be flipped in each iteration:

$$P^{(k)} = \left\lfloor HW(\mathbf{s}^{(k)}) \big/ \omega_c \right\rfloor \tag{5}$$

Where ω_c is the weight of each column in the parity check matrices. The use of this equation improves the decoding performance, while reducing its complexity in comparison to similar algorithms such as the parallel WBF algorithm [5]. This is because unlike the WBF algorithm, the use of this equation eliminates the need for simulation to determine which bits are to be flipped in each iteration. The idea behind this optimization is based on the observation that for the matrices associated with LDPC codes, the weight of the syndrome vector generally increases as does the number of errors. In other words, there is a strong correlation between the number of bits to be flipped and the hamming weight of the syndrome vector in each iteration. Since the aim of iterative decoding algorithm is to reach an all-zero syndrome vector, therefore the hamming weight of the syndrome vector in each iteration gives a more or less accurate estimation of the number of bits to be flipped.

2.3 SBWMBF Algorithm

The Search-Based Weighted Multi-Bit Flipping (SBWMBF) algorithm is proposed based on the idea that in the case of LDPC matrices, generally the syndrome weight increases with the number of errors until error weights much larger than half the minimum distance [5, 20, 21]. This algorithm flips $P^{(k)}$ bit(s) in k^{th} iteration. These bits are selected in such a way that reduces the weight of the syndrome vector. Obviously, the number of bits must be flipped in each iteration is different. Furthermore, their number in each iteration decrease compared with the previous iteration. The mechanism of this algorithm is that in each iteration first the number of bits that must be flipped is calculated using the hamming weight of the previous iteration syndrome vector with (5). In the next stage, the $P^{(k)}$ bits that must be flipped have to located. For this purpose, among all possible permutations created by the flipping $n_1, n_2, ..., n_{p(k)}$ bits, the algorithm chooses the best case. Best case is that leads to the syndrome vector with minimum weight and flips the bits are in these positions. The SBWMBF algorithm for calculating the hamming weight of the syndrome vector obtained by the flipping $n_1, n_2, ..., n_{p(k)}$ bits in the received vector should utilizes the following equation:

$$\mathbf{s}^{(k)}_{n_1,n_2,...,n_{p(k)}} = HW\left((\mathbf{z}^{(k-1)} + \mathbf{u}_{n_1,n_2,...,n_{p(k)}}).\mathbf{H}^T\right) \tag{6}$$

For the AWGN channel, a simple reliability measure, $|L_i|$, for a received symbol y_i is its magnitude, $|y_i|$. The larger the magnitude $|y_i|$ is equal to the greater reliability of the hard-decision digit z_i. If the reliability of a received symbol y_i is high, our desired is the decoding algorithm prevent from flipping this symbol, because the probability of this symbol being correct is higher than it being wrong. Also algorithm can be achieved this by appropriately increasing the values $\mathbf{s}^{(k)}_{n_1,n_2,...,n_{p(k)}}$ in the decoding procedure. Our proposed solution is to increase the values of $\mathbf{s}^{(k)}_{n_1,n_2,...,n_{p(k)}}$ by the term $\alpha\,\omega_c\left\{\left|L_{n_1}\right| + \dots + \left|L_{n_{p(k)}}\right|\right\}$, where $\alpha > 0$ is a positive weighting factor needed to be determined before decoding. The optimal value of α for a specific LDPC code can be determined through simulation. Therefore, to enhance the algorithm, the Hamming weight of the syndrome vector obtained from flipping bits $n_1, n_2, ..., n_{p(k)}$ in the received vector calculate using the following equation:

$$\mathbf{s}^{(k)}_{n_1,n_2,...,n_{p(k)}} = HW\left((\mathbf{z}^{(k-1)} + \mathbf{u}_{n_1,n_2,...,n_{p(k)}}).\mathbf{H}^T\right) + \alpha\omega_c\left\{\left|L_{n_1}\right| + ... + \left|L_{n_{p(k)}}\right|\right\} \tag{7}$$

Another key point to reduce the run time of this algorithm is to avoid considering all possible combinations of bit positions (locations). Our proposed suggestion is to put two conditions to determine the $n_1, n_2, ..., n_{p(k)}$ positions. Furthermore, it is not necessary to consider all possible positions, but just to inspect those bit combinations satisfying the following two conditions. For each $n_1, n_2, ..., n_{P^{(k)}}$:

1- $n_1 \neq n_2 \neq ... \neq n_{P^{(k)}}$

2- $1 \leq n_1 < n_2 < ... < n_{P^{(k)}} \leq N$

The necessity of applying these conditions in SBWMBF algorithm to check the unique combination of $n_1, n_2, ..., n_{P^{(k)}}$ bits is avoiding the repetitive combinations. For example, suppose that $P^{(k)} = 3$ and $N = 6$. Therefore, we need the algorithm determines $n_1, n_2, n_3 \in \{1, 2, ..., 6\}$. In the meantime, assuming n_1, n_2, n_3 have values 1, 2 and 3, all the following states have the same meaning:

$$\{(1,2,3),(1,3,2),(2,1,3),(2,3,1),(3,1,2),(3,2,1)\}$$

Because all these cases dictate that the first, second and third bits of the received sequence must be flippedAnother major modification to improve the performance of the proposed algorithm concerns the use of a new loop-detection mechanism. The loop-detection mechanism was first developed in [7] for algorithms in which only a single bit is flipped in each iteration. But the loop-detection we have used here is completely different from that, and is especially designed for algorithms in which several bits are flipped in each iteration. The steps of this algorithm are fully presented below.

Search-Based Weighted Multi-Bit Flipping Algorithm (SBWMBF):

- **Initialization:** Set iteration counter $k = 0$ and $k_{\max}$ as the user defined maximum number of iterations and also set the Loop-Exclusion list containing bit positions that cause loop $LE_0 = \varnothing$. calculate $\mathbf{z}^{(0)} = \text{sgn}(\mathbf{y})$ and $\mathbf{s}^{(0)} = HW(\mathbf{z}^{(0)}.\mathbf{H}^T)$.
- **Step 1)** If $\mathbf{s}^{(k)} = \mathbf{0}$, stop the decoding and return $\mathbf{z}^{(k)}$; otherwise calculate $P^{(k)} = \left\lfloor HW(\mathbf{s}^{(k)}) / \omega_c \right\rfloor$.
- **Step 2)** $k \leftarrow k+1$. If $k > k_{\max}$, declare a decoding failure and return $\mathbf{z}^{(k-1)}$; otherwise go to step 3.
- **Step 3)** For each $n_1, n_2, ..., n_{P^{(k)}} \in \{1, 2, ..., N\}$ which satisfies two following conditions:

  1- $n_1 \neq n_2 \neq ... \neq n_{P^{(k)}}$

  2- $1 \leq n_1 < n_2 < ... < n_{P^{(k)}} \leq N$

  calculate:

$$\mathbf{s}^{(k)}_{n_1, n_2, ..., n_{P^{(k)}}} = HW((\mathbf{z}^{(k-1)} + \mathbf{u}_{n_1, n_2, ..., n_{P^{(k)}}}).\mathbf{H}^T) + \alpha\omega_c \left\{ \left| L_{n_1} \right| + ... + \left| L_{n_{P^{(k)}}} \right| \right\}$$

- **Step 4)** Identify the set of bit position(s) to be flipped $j^{(k)}$ with:

$$j^{(k)} = \underset{(n_1,n_2,\ldots,n_{p(k)})\in[1,N],\notin LE_k}{\arg\min}\ \mathbf{s}^{(k)}_{n_1,n_2,\ldots,n_{p(k)}} .$$

- **Step 5)** For each $1 \le k' \le k-1$, if $j^{(k)} = j^{(k-k')}$, set $LE_k = LE_k \cup \{j^{(k)}\}$ and go back to step 4; otherwise go to step 6.

- **Step 6)** Compute $\mathbf{z}^{(k)} = \mathbf{z}^{(k-1)} + \mathbf{u}_{j^{(k)}}$ and $\mathbf{s}^{(k)} = HW(\mathbf{z}^{(k)}.\mathbf{H}^T)$. Then go back to step 1.

2.4 Simplified SBWMBF Algorithm

The proposed SBWMBF algorithm has an excellent performance very close to soft decision algorithms like BP just with little performance penalty. The only weakness of this method is a long time required to run it. Studies on the simulation results show that, this long time required for process decoding operations is related to the third step of the algorithm. In similar algorithms such as the Standard WBF and the other improved versions, the algorithm calculates only N reliability measure in each iteration. But in the SBWMBF algorithm, the number of permutations that $\mathbf{s}^{(k)}_{n_1,n_2,\ldots,n_{p(k)}}$ have to be determined for them is $\binom{N}{P^{(k)}}$. Obviously always $\binom{N}{P^{(k)}} \gg N$ and this amount increases exponentially by increasing the code length.

In order to accelerate the run time of the SBWMBF algorithm, we have proposed an innovative idea. This idea concerns with the using a new mechanism to reduce the number of cases to be investigated in the third step. The resulting algorithm is named the SSBWMBF decoding Algorithm. The SBWMBF algorithm investigates all possible permutations for determining $P^{(k)}$ bits that must be flipped in k^{th} iteration. Actually the algorithm searches among N location of code length in order to find the case leading to the syndrome vector with minimum hamming weight. The total number of these cases is equal to $\binom{N}{P^{(k)}}$. However, not all these permutations are considered in the simplified version of the algorithm. In fact, by choosing arbitrary number $1 \le \beta \le P^{(k)}$ and determining the $P^{(k)} + \beta$ bit from minimum values of $\mathbf{s}_i^{(k)}$ and storing them in a set $CF^{(k)}$, we investigate the total permutations of $P^{(k)}$ bits that must be flipped in k^{th} iteration among the $P^{(k)} + \beta$ locations of $CF^{(k)}$ set. The total number of these cases is equal to $\binom{P^{(k)}+\beta}{P^{(k)}}$. Obviously, given that in the case of LDPC codes $P^{(k)} + \beta \ll N$, and as for choose of any value for β can be concluded that always $\binom{P^{(k)}+\beta}{P^{(k)}} \ll \binom{N}{P^{(k)}}$.
Also in the simplified version of algorithm in order to improve the error performance a new loop detection mechanism is utilized.

Simplified Search-Based Weighted Multi-Bit Flipping Algorithm (SSBWMBF):

- **Initialization:** Set iteration counter $k = 0$ and $k_{\max}$ as the user defined maximum number of iterations and also set the Loop-Exclusion list containing bit positions that cause loop $LE_0 = \varnothing$. calculate $\mathbf{z}^{(0)} = \operatorname{sgn}(\mathbf{y})$ and $\mathbf{s}^{(0)} = HW(\mathbf{z}^{(0)}.\mathbf{H}^T)$.

- **Step 1)** If $\mathbf{s}^{(k)} = \mathbf{0}$**,** stop the decoding and return $\mathbf{z}^{(k)}$; otherwise calculate $P^{(k)} = \left\lfloor HW(\mathbf{s}^{(k)}) \big/ \omega_c \right\rfloor$.

- **Step 2)** $k \leftarrow k+1$. If $k > k_{\max}$, Declare a decoding failure and return $\mathbf{z}^{(k-1)}$; otherwise go to step3.

- **Step 3)** For each $i = 1, 2, \ldots, N$ compute:
$$\mathbf{s}_i^{(k)} = HW((\mathbf{z}^{(k-1)} + \mathbf{u}_i).\mathbf{H}^T) + \alpha\omega_c \{|L_i|\}$$

- **Step 4)** for each arbitrary $1 \le \beta \le P^{(k)}$, determine the $P^{(k)} + \beta$ bits from minimum values of $\mathbf{s}_i^{(k)}$ and stored them in a set $CF^{(k)}$.

- **Step 5)** For each $n_1, n_2, \ldots, n_{P^{(k)}} \in CF^{(k)}$ which satisfy the following two conditions:
$$n_1 \ne n_2 \ne \ldots \ne n_{P^{(k)}}$$
$$1 \le n_1 < n_2 < \ldots < n_{P^{(k)}} \le N$$
calculate:
$$\mathbf{s}^{(k)}_{n_1,n_2,\ldots,n_{P^{(k)}}} = HW((\mathbf{z}^{(k-1)} + \mathbf{u}_{n_1,n_2,\ldots,n_{P^{(k)}}}).\mathbf{H}^T) + \alpha\omega_c \left\{ \left|L_{n_1}\right| + \ldots + \left|L_{n_{P^{(k)}}}\right| \right\}$$

- **Step 6)** Identify the set of bit position(s) to be flipped with
$$j^{(k)} = \underset{(n_1,n_2,\ldots,n_{P^{(k)}}) \in [1,N], \notin LE_k}{\arg\min}\ \mathbf{s}^{(k)}_{n_1,n_2,\ldots,n_{P^{(k)}}}.$$

- **Step 7)** For each $1 \le k' \le k-1$, if $j^{(k)} = j^{(k-k')}$, set $LE_k = LE_k \cup \{ j^{(k)} \}$ and go back to step 6; otherwise go to step 8.

- **Step 8)** Compute $\mathbf{z}^{(k)} = \mathbf{z}^{(k-1)} + \mathbf{u}_{j^{(k)}}$ and $\mathbf{s}^{(k)} = HW(\mathbf{z}^{(k)}.\mathbf{H}^T)$. Then go back to step 1.

3. Simulation Results

In this section, the error performance as a function of the signal-to-noise ratio (SNR) for the two proposed algorithms is compared with Modified WBF and BP algorithms. The maximum number of iterations have been set at 100, 50, 10, 10 for BP, Modified WBF, SBWMBF and SSBWMBF algorithms, respectively. Also the parameter β is fixed $\beta = P^{(k)}$. For the purposes of simulation, large different types of LDPC codes have been used, but for the sake of brevity, only the results of the simulation a (2048, 1664) PEG-LDPC code with column weight 6 is presented.

3.1 BER performance of the proposed algorithms

Fig. 2 compares the performance of the two proposed algorithms with Modified WBF and BP decoding algorithms. The optimum value of weighting factor has been set $\alpha = 1.5$ for both SBWMBF and SSBWMBF algorithms. This value chosen as the optimal α at $BER = 10^{-5}$.It was observed through simulations of various LDPC codes that the two proposed algorithms always outperform the Modified WBF algorithm. For example in figure 2, it can be observed that, at the BER of 10^{-6}, the SBWMBF and SSBWMBF algorithms outperform the Modified WBF algorithm by about 0.7 and 0.5 dB respectively. Note that the SBWMBF algorithm is slightly more complex than the Modified WBF decoding. Running the SBWMBF algorithm take a long time especially for lengthy LDPC codes. This is because of the large number of choices for flipping their bit positions. Fortunately the BER performance of the SBWMBF algorithm shows a performance gap of 0.2 dB with its simplified version. Since run time of the algorithm and the number of cases to be considered significantly reduced, this performance gap is completely acceptable. Furthermore, it can be observed that, performance of the SBWMBF Algorithm is 0.2 dB worse than the BP algorithm. However the number of required iterations and it's complexity is less than BP. As mentioned earlier, the required run time for SBWMBF is slightly more than BP decoding. In order to eliminate this weakness we offer a simplified version with about 0.4 dB worse performance than the BP algorithm. The SSBWMBF algorithm requires very fewer iterations and has less complexity and run time BP decoding algorithm. Thus the SSBWMBF algorithm is a good option for decoding LDPC codes.

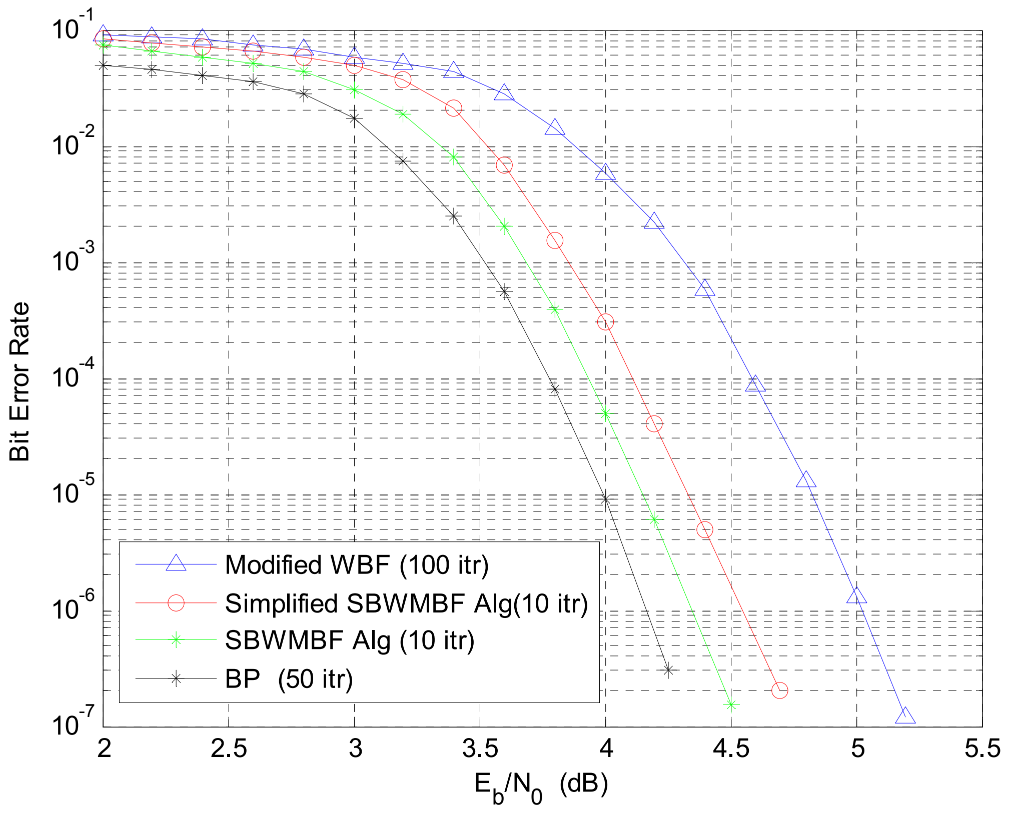

Figure 2. The BER performance of the (2048,1664) PEG- LDPC code with column weight 6 based on Modified WBF, BP and the proposed algorithms.

3.2 Impact of SNR on the optimal value of α in proposed algorithms

For LDPC code with given column weight, at a given SNR, the performances of two proposed Algorithms vary with the value of the weighting factor α . The effect of α on the BER of the (2048, 1664) PEG-LDPC code with column weight 6 is shown in figure 3.

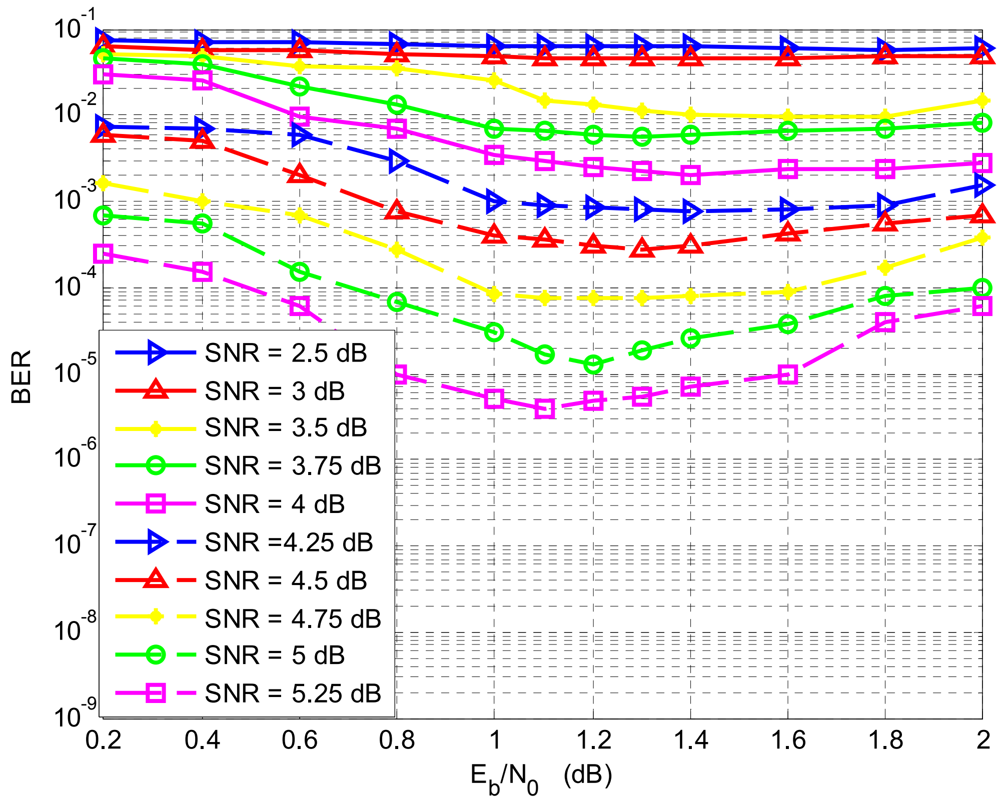

Figure 3. Impact of the weighting factor α in the proposed Algorithms on the BER performance for the decoding of the (2048, 1664) PEG-LDPC code with column weight 6.

It can be seen that:

1- The optimal value of α decreases slowly as the SNR increases.

2- In low SNRs, the performance is not very sensitive to the value of α .

The same observations were made for the different LDPC codes studied. Accurate studies on simulation results for different LDPC codes have shown that the optimal weighting factor of α depends on the SNR and the code structure. However, choosing the α at $BER = 10^{-5}$ and keeping constant it as optimal value of α for all values of SNRs does not cause significant degradation in performance. Therefore, the optimal weighting factor α can be considered to be only code-dependent.

3.3 Impact of the number of iterations on the proposed algorithm's performance

Another prominent feature of the two proposed algorithms is that they require much fewer iterations compared to the BP algorithm in regions of both high and low SNRs. Therefore, their convergence rates are much faster than that of the BP algorithm. This is because in these algorithms a large portion of the operations are carried out during the initial iterations, particularly the first one. This is caused by the use of the equation (5) to calculate the number of bits to be flipped in each iteration. In figure 4, SSBWMBF with different iteration numbers is shown. As illustrated in figure 4, the performance of the SSBWMBF is only about 0.2 dB weaker than its original version. Also the performance difference between SSBWMBF with 10 and 20 iterations is very negligible. Therefore, the number of required iterations and parameter β for running the SSBWMBF in order to reach the optimal performance are 10 and $\beta = P^{(k)}$ respectively.

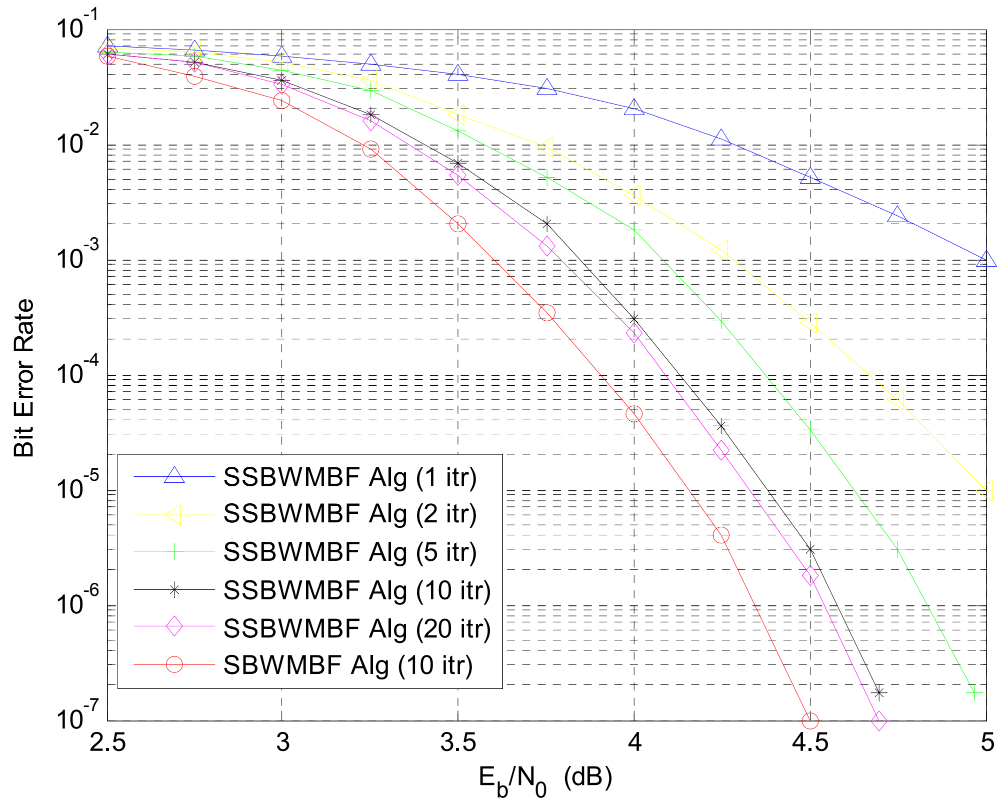

Figure 4. The impact of the number of iterations on the performance of SSBWMBF algorithm with column weight 6 and optimal α an. $\beta = P^{(k)}$.

4. Discussion

In this section, we discuss about the results. Important parameters in evaluating a decoding algorithm include error performance, the complexity and the number of required iterations. In the previous section, we have evaluated these parameters for the proposed algorithms, comparing them with the other algorithms. It is necessary to mention that the soft decision algorithms like BP have the best performance. However, numerous computations make the decoding process complicated and the memory in the decoder required may be prohibitive in practical application. We have compared the performance of the two proposed algorithms with BP and Modified WBF algorithms in figure 2. Looking at this figure, it can be seen that the performance of the proposed algorithms is very close to BP. Also both of the proposed algorithms outperform Modified WBF. This is due to the fact that reliability of received symbol in our proposed methods is higher than the other hybrid decoding like Modified WBF. The solution we proposed is to increase the values of $\mathbf{s}^{(k)}_{n_1,n_2,\dots,n_{p(k)}}$ by the term $\alpha\omega_c\left\{\left|L_{n_1}\right|+\dots+\left|L_{n_{p(k)}}\right|\right\}$.This solution, which is very simple, happens to be very efficient as we have shown by the simulation results. Our intuitive approach is justified by the following facts:

- Generally larger $|L_i|$ implies that the hard decision $\mathbf{z}_i$ is more reliable.
- If there is no overlap between the "non zero" elements of the different columns of **H**, the contribution of each column to the syndrome weight is ω_c .
- For different values of the signal-to-noise ratio (SNR), the values of $|L_i|$ are not the same. The presence of the weighting factor is due to this fact.

According to figure 3, optimal weighting factor of α depends on the SNR and the code structure. Actually, there is strong correlation between optimal weighting factor and code type, length of the code, column weight, BER and SNR. For the sake of clarity, in table 1 we illustrate the impact of these parameters on α_{opt} .

Table 1. Optimum weighting factor for (2048,1664) PEG -LDPC code with different parameters

Code Type	(N,K)		SNR (dB)	BER	
PEG LDPC	(2048,1664)	3	4.0		1.4
			4.5		1.3
			5.0		1.2
			5.25		1.1
	(2048,1664)	6	4.0		1.7
			4.25		1.6
			4.5		1.5
			4.75		1.2

It was observed in figure 2 that the two proposed algorithms always outperform the Modified WBF algorithm. SSWMBF algorithm with loop detection offers a gain of about 0.5 dB over the Modified WBF, and is less than 0.4 dB worse than the BP algorithm. Performance of SMWMBF and SSBWMBF algorithms with different iterations compared in figure 4. It can be observed that, SSBWMBF has weaker performance about 0.2 dB in comparison with SBWMBF. In return, SSBWMF has faster convergence rate and require less run time for decoding. The number of required iterations for two proposed algorithms in comparison with the other methods is much fewer. To clarify this, the average number of iterations in terms of E_b/N_0 for these algorithms is compared in figure 5.

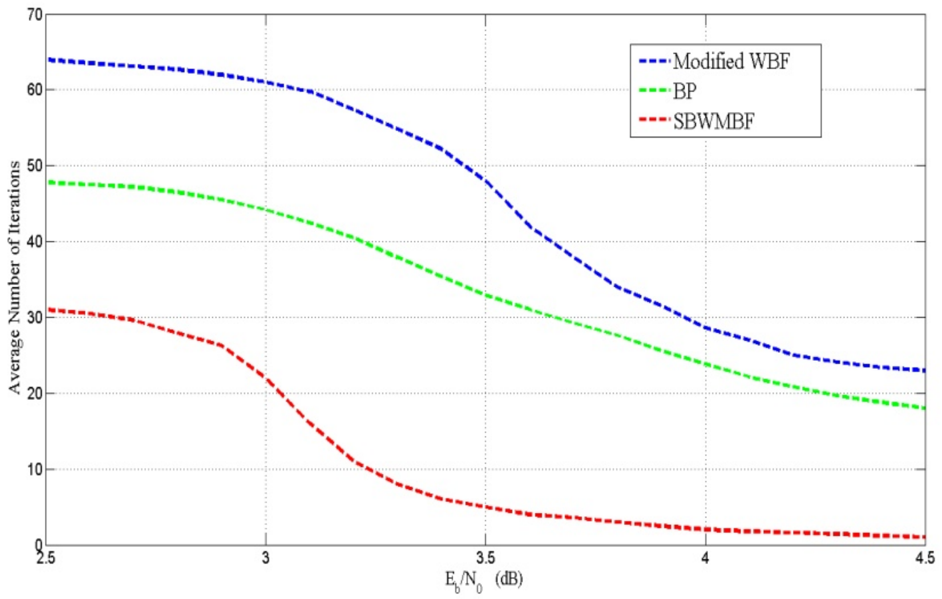

Figure 5. Average number of iterations for the Modified WBF, BP and SBWMBF algorithms for decoding the (2048, 1664) PEG-LDPC code

According to this figure, the SBWMBF algorithm needs approximately one iteration at $E_b/N_0 = 4.5dB$, while the BP and modified WBF algorithm require less and more than 20 iterations. The SSBWMBF algorithm has a fewer complexity than BP in each iteration. Also as illustrated in figure 5, this algorithm requires much fewer iterations compared to other algorithms in regions of both high and low SNRs. Therefore, its convergence rate is much faster than the other two algorithms. This is obviously because in proposed algorithms a large portion of the operations do during the initial iterations, particularly the first one. This is caused by the use of the equation (5) to calculate the number of bits to be flipped in each iteration.

5. Conclusion

In this paper, we propose two search based hybrid decoding algorithms for regular LDPC codes. Our simulation results indicate that compared to the Modified WBF decoding, the two proposed algorithms converge very much faster and outperform it. Furthermore, for different types of LDPC codes it was shown that these proposed algorithms perform very close to standard BP decoding, but with faster convergence rates. In order to reduce the cases to be considered, we have offered a simplified version of the algorithm which is very simple and competitive for designing high-performance decoders.

References

[1] R. G. Gallager, “Low Density Parity-check codes,” IRE Trans. Inform. Theory , vol. IT-8, no. 1, pp. 21–28, Jan. 1962.

[2] D. J. C. MacKay and R. M. Neal, “Near Shannon limit performance of low density parity-check codes,” Electro. Lett., vol. 32, pp. 1645–1646, Aug. 1996.

[3] D. J. C. MacKay, “Good error-correcting codes based on very sparse matrices,” IEEE Trans. Inform. Theory, vol. 45, no. 2, pp. 399–432, Mar. 1999.

[4] A. Blanksby and C. Howland, “A 690-mW 1-Gb/s 1024-b, rate-1/2 low-density parity-check code decoder,” IEEE Journal of Solid-State Circuits, vol. 37, no. 3, pp. 404-412, Mar 2002

[5] M. Rovini, N. E. L’Insalata, F. Rossi and L. Fanucci, “VLSI design of a high-throughput multi-rate decoder for structured LDPC codes,” In Proc. 8th euromicro conf. on digital system design (DSD 2005), pp. 202–209, Aug 2005.

[6] J. Luand J. M. F. Moura, “Structured LDPC codes for high-density recording: Large girth and low error floor,” IEEE Transactions on Magnetics, vol. 42, no. 2, pp. 208-213, Feb 2006.

[7] Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications, Draft EN 302 307 V1.1.1 ed., ETSI, June 2004.

[8] IEEE P802.11 Wireless LANs TGn Sync proposal: TgnSync proposal Technical Specification, May 2005, IEEE 11-04-0889-06-000n.

[9] IEEE P802.11 Wireless LANs WWiSE proposal: High throughout extension to the 802.11 Standard. Aug. 2004, IEEE 11-04-0886-00-000n.

[10] LDPC coding for OFDMA PHY. 802.16REVe Sponsor Ballot Recirculation comment, Jul. 2004, IEEE C802.16e-04/141r2.

[11] C. Berrou and A. Glavieux, “Near optimum error correcting coding and decoding: turbo-codes,” IEEE Trans Commun, vol. 44, pp. 1261-1271, 1996.

[12] Y. Fengfan and Y. Ming, “High-Performance simple-encoding generator-based systematic irregular LDPC codes and resulted product codes, ” Journal of electron(china), vol.24, no. 5, pp. 613-621, Sep 2007.

[13] V. K. Kidambi, C. K. singh and P. T. balsara, “A generic scalable architecture for Min-Sum/offset-Min-Sum unit for irregular/regular LDPC decoder,” IEEE trans on very large scale integration (VLSI) systems, vol. 18, no. 9, Sep 2010.

[14] S. Lin and D. J. Costello, “Error Control Coding: Fundamentals and Applications,” Englewood Cliffs, NJ: Prentice Hall, 1983.

[15] J. L. Massey, “ Threshold Decoding,” Cambridge, MA: MIT Press, 1963.

[16] V. D. Kolesnik, “Probability decoding of majority codes,” Prob. Peredachi Inform., vol. 7, pp. 3-12, July 1971.

[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 Trans. Inform. Theory, vol. 47, pp. 2711-2736, Nov. 2001.

[18] Z. Liu and D. A. Pados, “A decoding algorithm for finite geometry LDPC codes,” IEEE Trans. Commun., vol. 53, pp. 415-421, Mar. 2005.

[19] J. Zhang and M. P. C. Fossorier, “A Modified Weighted Bit-Flipping Decoding of Low-Density Parity-Check Codes,” IEEE Comm Lett, vol. 8, pp. 165-168, 2004.

[20] D. J. C. MacKay, “Good error-correcting codes based on very sparse matrices,” IEEE Trans. Inform. Theory, IT-45, pp. 399-432, Mar. 1999.

[21] N. Wiberg, “Codes and decoding on general graphs,” Ph.D. dissertation, Department of Electrical Engineering, Linkoping University, Linkoping, Sweden, 1996.

[22] M. Fossorier, M. Mihaljevic and H. Imai, “Reduced complexity iterative decoding of low density parity check codes based on belief propagation,” IEEE Trans on Comm, vol. 47, pp. 673-680, 1999.

[23] J. Chen and M. Fossorier, “Density evolution of two improved BP-based algorithms for LDPC decoding” IEEE Comm Lett, vol. 6, pp. 208-210, 2002.

[24] J. Chen, “Reduced complexity decoding algorithms for low-density parity check codes and turbo codes,” Ph.D. dissertation, University of Hawaii, Dept. of Electrical Engineering, 2003.

[25] E. O. Torshizi, H. Sharifi and M. Seyrafi, “A New Hybrid Decoding Algorithm for LDPC Codes Based on the Improved Variable Multi Weighted Bit-Flipping and BP Algorithms,” 21st IEEE Iranian Conference on Electrical Engineering (ICEE), p.1-6, Mashhad, 2013.

[26] T. M. N. Ngatched and M. Bossert, “Two Bit-Flipping Algorithms for Low-Density Parity-Check Codes,” IEEE Trans Commun, vol.57, no. 3, pp. 591-596, mar 2009.