International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 05, May-2015

BER Performance Comparison of Bit Flipping Algorithms used for Decoding of LDPC Codes

Pavithra G.
Department of Telecommunication Engineering
Siddaganga Institute of Technology,
Tumakuru-572103, Karnataka, India

Naveen Kumar M.
Department of Telecommunication Engineering
Siddaganga Institute of Technology,
Tumakuru-572103, Karnataka, India

Abstract— **In this paper the Bit Error Rate (BER) performance of different bit flipping algorithms used for decoding of Low Density Parity Check (LDPC) code is compared. These algorithms mainly depend on inversion function. Through simulation results the Noisy Gradient Descent Bit flipping (NGDBF) algorithm is proved to be best till date. This algorithm provides the best BER performance, for Smoothed Noisy Gradient Descent Bit flipping (SM-NGDBF) algorithm we can obtain BER 4.86×10^{-4} at 3.5db. The Multi Noisy Gradient Descent Bit flipping (M-NGDBF) algorithm requires the least number of iterations than the other algorithms proposed for decoding a codeword.**

Keywords- LDPC codes, bit-flipping algorithm, noisy gradient descent algorithm

I. INTRODUCTION

In present days Low Density Parity Check (LDPC) codes have gained attention in research area. LDPC codes are used in different communication standards, due to their powerful decoding performance. Depending on the choice of decoding algorithm the performance and cost of using LDPC codes are determined. LDPC decoding algorithms are iterative in nature. They function by exchanging messages between basic processing nodes called variable node and check node. Among the existing decoding algorithm the min-sum and sum product offer the best performance but they require large number of arithmetic operation [1], [2]. This results in LDPC decoder to be highly complex device.

Another class of LDPC decoding algorithm is bit flipping algorithm [3]. In this paper the bit flipping algorithm proposed make use of both hard decision and soft information from the channel. These bit flipping algorithms provides less Bit Error Rate Performance (BER) than the min-sum and sum product algorithm but they enable the design of much simpler decoder.

In bit flipping algorithm symbol node update depending on the inversion function. The inversion function value is estimated by using the reliability of received channel samples and they also make use of the hard decision syndrome component obtained from the code parity check equation. In single bit flipping algorithm only one bit is flipped at each iteration. In multiple bits flipping algorithm all the bits which are below the given threshold are flipped, this results in faster operation.

In this paper the BER performance of different bit flipping algorithms is compared. Through simulation results the Noisy Gradient Descent Bit flipping (NGDBF) algorithms is proved to be best. As the number of iterations required for decoding a codeword decreases, it facilitates in high-throughput decoding. By simulations it is shown that Multi Noisy Gradient Descent Bit flipping (M-NGDBF) algorithm requires the least number of iterations. The SM-NGDBF algorithm provides the best BER performance but it requires maximum iteration(Lmax) of 300.

II. BIT FLIPPING DECODING ALGORTIHMS

Preliminaries:

Let H be a binary $m \times n$ parity check matrix, where $n > m \geq 1$. The binary linear code C which is associated with matrix H is defined by $C=\{\boldsymbol{c}\ F_2^n : H\boldsymbol{c} = \boldsymbol{0}\}$, where F_2 denotes the binary Galois field. The codeword C is modulated using Binary phase shift key (BPSK) modulation. After modulation the codeword $\hat{C}$ is given by $\hat{C} = \{(1-2c_1), (1-2c_2),\ldots.(1-2c_n):c=C\}$. The codeword C is of binary form C€(0,1) and $\hat{C}$ is a subset of bipolar codeword $\{+1,-1\}^n$.

Later the codeword $\hat{C}$ is passed through Additive White Gaussian noise (AWGN) channel, it is defined by operation $y=c+z$, $c€\hat{C}$ and z is a independent noise vector $z€(z_1,z_2,\ldots.,z_n)$, where z_j (j€1,n]) is a Gaussian random variable with zero mean and variance$N_0/2$, N_0 is noise spectral density. At the receiver y is the vector of samples obtained.

Let N(i) be the parity check neighborhood defined as $N(i)=\{j€[1,n]:h_{ij}=1\}$ for i=1,2,…m and M(j) be the symbol neighborhood defined as $M(j)=\{i€[1,m]:h_{ij}=1\}$for j=1,2,…,n where h_{ij} is the ij element of parity check matrix. The parity check condition using these notation is expressed as $s_i=\prod_{j€N(i)}x(j)$ (i€[1,m]), where the value of $\prod_{j€N(i)}x(j)€(+1,-1)$ is called as ith bipolar syndrome component of x. If bipolar syndrome component $s_i=1$ the corresponding node is said to be satisfied.

A. Weighted Bit Flipping (WBF) Algorithm:

WBF is a single bit flipping algorithm. In this algorithm only one bit is flipped at each iteration, the bit to be flipped depend on inversion function value. The inversion function of WBF [2] is given by

$$\Delta_k^{(WBF)}(x) = \sum_{i\in M(k)} \beta_i \prod_{j\in N(i)} x_j \qquad (1)$$

where $\beta_i \triangleq \min_{j€N(i)} |y_j|$

The value β_i (i € [1,m]) defines the reliabilities of bipolar syndrome. The inversion function contains the sum of weighted bipolar syndrome which gives a measure of

invalidness of symbol assignment of x_k. The bit with lower inversion function value will be flipped.
The bit flipping algorithm is summarized as :

```
1. For j=1:n
   $x_j=\text{sign}(y_j)$
2. If $\prod_{j\in N(i)} x_j=+1$ for all $i\in\{1,2,\ldots,m\}$
   Output x and stop
3. Flip bit $x_\ell$ where , $\ell \triangleq \underset{k\in[1,n]}{\text{argmin}}\ \Delta_k^{(WBF)}(x)$
4. If maximum number of iterations is reached output x and stop
   Otherwise go to step 2
```

B. Multi Weighted Bit Flipping (MWBF) Algorithm:

MWBF is also a single bit flipping algorithm. It gives better performance than the WBF and also requires less number of iteration to decode a codeword. The inversion function of MWBF [4] is given by

$$\Delta_k^{(MWBF)}(x) = \alpha|y_k| + \sum_{i\in M(k)} \beta_i \prod_{j\in N(i)} x_j \qquad (2)$$

The inversion function of MWBF is same as that of WBF except that it contains an additional term. The first term in the equation tells about interior bit based message. The second term in the equation give information about only check based message, it comes from check constraints. Thereby in MWBF we consider both the check based and bit based message for each position. A weighting factor α is considered for bit message because for different code with different column weight or for different values of SNR the weight of bit message should not be same. The algorithm for MWBF is same as that of WBF, only the inversion function equation is replaced.

C. Gradient Descent Bit Flipping (GDBF) algorithms:

The performance of WBF and MWBF algorithm is not closer to min sum algorithm. So to improve the BER performance GBBF [5] algorithm was proposed In GDBF algorithm majority logic decoding is used to optimize the gradient descent model. Using this an objective function is derived, it is given as

$$f(x) = \sum_{k=1}^{n} x_k y_k + \sum_{i=1}^{m} \prod_{j\in N(i)} x_j \qquad (3)$$

The objective function is non linear function it has many local maxima. The first term in objective function gives information about the correlation between bipolar codeword and received word, it should be maximized. Only when x is a codeword the second term which is sum of syndrome has maximum value 'm'.

By maximizing f(x) we get an inversion function for GDBF. Maximizing is done by taking partial derivative of f(x) with respect to variable x_k and multiplying the derivative with x_k. Then we can obtain the inversion equation for GDBF defined as

$$\Delta_k^{(GDBF)}(x) = x_k y_k + \sum_{i\in M(k)} \prod_{j\in N(i)} x_j \qquad (4)$$

C(a). Single GDBF:

The inversion function of single GDBF is given by equation (4). In this equation the first term contain the correlation between the hard decision and soft value for bit k and the second term has hard decision .The inversion function value for each bit is calculated, the bit with minimum value is flipped. As at each iteration only one bit is flipped at a time, it results in a larger delay.

C(b). Multi GDBF:

It is a combination of single bit flipping and multi bit flipping algorithm. In multi bit flipping algorithm large step size is used so it converges faster to local maxima but it lead to oscillation around local maxima. In order to avoid this single bit flipping algorithm is used as it provides smaller step size and lead to slower convergence to local maxima.

In this algorithm based on the objective function value switching is done from single bit flipping algorithm to multi bit flipping algorithm to find the local maxima. In multi GDBF algorithm two parameters are used θ and μ. The parameter θ is a negative real number, called as inversion threshold. The variable μ is called as mode flag, it is binary (0 or 1). At the beginning of decoding process μ is set to 0. Step 3 of bit flipping algorithm is replaced with following procedure.

```
3. Calculate the objective function $f_1:=f(x)$. if $\mu=0$ execute 3-1 else execute 3-2
   3-1 flip all bits satisfying $\Delta_k^{(GDBF)}(x) < \theta (k\in[1,n])$
       re-evaluate $f_2:=f(x)$ if $f_1>f_2$ ,then $\mu=1$.
   3-2 flip bit $x_\ell$ where , $\ell \triangleq \underset{k\in[1,n]}{\text{argmin}}\ \Delta_k^{(GDBF)}$
```

C(c). Multi GDBF with escape:

If the search point is captured by local maximum then decoding failure occurs. Since the search point will be having very small weight from the final position even a small perturbation can help in escaping from a local maximum. By this the performance of BF algorithm can be increased.

In order to avoid this multi GDBF with escape process is used. In this when search point arrives at non transmitted codeword mode flag is switched from single bit to multi bit mode. This is called as escape process. After this search point again begin to climb up the hill, which is different than that of trapped search point.

In multi GDBF with escape algorithm we use two threshold θ_1 and θ_2. At the beginning of decoding process θ_1 is used as threshold for multi bit mode. After some iterations the multi bit mode is changed to single bit mode when objective function $f_1>f_2$. During single bit mode operation the search point may be captured by local maxima. In order to avoid this single bit mode is again changed to multi bit mode with threshold θ_2, it is called as threshold for downward movement. Only one iteration is performed with this threshold later it is switched back to multi bit mode with threshold θ_1. This process continues till parity check condition is satisfied or the number of iteration reaches its maximum value.

D. Adaptive Threshold Bit Flipping (ATBF) algorithm:

ATBF provides high speed decoding and low power operation. In all the other algorithms described above we have to locate the minimum value over the whole block length by calculating inversion function for each bit. This result in decrease of speed of decoding, by using ATBF decoding algorithm speed can be increased.

The inversion function of ATBF [6] is given by

$$\Delta_k^{(ATBF)}(x) = x_k y_k + \sum_{i\in M(k)} \prod_{j\in N(i)} x_j \qquad (5)$$

The equation is same as that of GDBF. In this algorithm two parameter is used θ and λ, λ_k $k \in (1,n)$ be a negative threshold value associated with each received bit. In order to modify λ_k a constant scaling factor is used $\theta \in [0,1]$.

The benefit of using this algorithm by thresholding on a per bit level is at beginning multiple bit will be flipped, as the number of iteration increases the number of bit flipped decreases as most checks are satisfied. Thus ATBF move from multi bit flipping to single bit flipping as required. Therefore this algorithm require only localized operation and eliminate the need to locate a minimum value over entire block length.

The bit flip algorithm is modified in following step:

```
Step 0: initialize $\lambda_k = \lambda_0$, $k \in \{1,2,..n\}$
Step 3: for k=1:n
        If $\Delta_k^{(ATBF)}(x) < \lambda_k$
            flip bit $x_k$
        otherwise $\lambda_k = \theta \lambda_k$
```

The number of iteration required to decode a codeword is less for ATBF algorithm when compared to other to other algorithm like WBF, MWBF and GDBF but BER performance is less.

Noisy Gradient Descent Bit Flipping (N-GDBF) algorithms:

The performance of GDBF algorithm is increased by escaping from the local maxima, but it leads to increase in complexity. The complexity can be reduced by introducing a random perturbation in the inversion function. This give rise to algorithm called Noisy GDBF [7].

S-NGDBF:

The inversion function of single N-GDBF is equation

$$\Delta_k^{(NGDBF)}(x) = x_k y_k + w \sum_{i\in M(k)} s_i + q_k \qquad (6)$$

In the equation (6) q_k is Gaussian distributed random variable with zero mean and variance $\sigma^2 = \eta^2 N_0/2$, where $0<\eta\leq 1$ and w is a syndrome weighting parameter, it is motivated by local maximum likelihood analysis. Numerical optimization is used to find η and w, they are close to one. The optimal value of w and η are code independent and in certain cases found to be weakly SNR dependent. It is single bit flipping algorithm so only one bit is flipped at each iteration. The algorithm used is same as that of BF algorithm only inversion function is replaced by equation (6).

M-NGDBF:

The convergence of multi bit flipping algorithm can be improved by using threshold adaptation method. Therefore in M-NGDBF algorithm instead of switching method threshold adaptation method is used. Due to this performance increased and number of iterations decreased. The algorithm for M-NGDBF can be obtained by modifying following steps in BF algorithm.

```
Step 0: initialize $\lambda_k = \lambda_0$, $k \in \{1,2,..n\}$

Step 3: for k=1:n
        if $\Delta_k^{(NGDBF)}(x) < \lambda_k$
            flip bit $x_k$
        otherwise $\lambda_k = \theta \lambda_k$
```

The term θ is called as adaptation parameter, the value of θ is less than one for adaptive case and for non-adaptive it is equal to one.

SM-NGDBF:

Due to excessive flipping of low confidence symbol convergence failure occur in M-NGDBF algorithm. This is due to stochastic distribution term. To avoid this up and down counter is used at output of every x_k. The counter is initialized to zero at start of decoding. After each decoding iteration the counter is updated using the equation

$$X_k(t+1) = X_k(t) + x_k(t) \qquad (7)$$

The equation (7) implies that the counter consist of running sum X_k for each of N output decision. If all parity check condition is satisfied before the maximum number of iteration completed then output the x_k directly. If output is not decoded even after the maximum number of iterations then smoothen the decision x_k=sign X_k. The summation is taken only for the interval t=Lmax-64 upto Lmax. This is referred as smoothed NGDBF.

III. SIMULATION RESULTS:

A. BER Performance

In this section the simulation results obtained for different bit flip algorithms is presented. For simulation a regular LDPC code m=504, n=1008 (called PEGReg504×1008 in [8]) is used. The column weight of LDPC code is 3. In all the algorithms the same noise AWGN is added and it output the correct codeword. Thus the decoding was successful.

Figure 1 presents the comparisons between BER curves for WBF(Lmax=100), MWBF (Lmax=100,α=0.2), single GDBF(Lmax=100), Multi GDBF(Lmax=100, θ=-0.6), multi GDBF with escape(Lmax=300, θ_1=-0.7, θ_2=1.7+α) α is a Gaussian random variable with zero mean and variance 0.01 and Lmax is the maximum number of iterations. In the figure 1 the multi GDBF with escape gives the best performance.

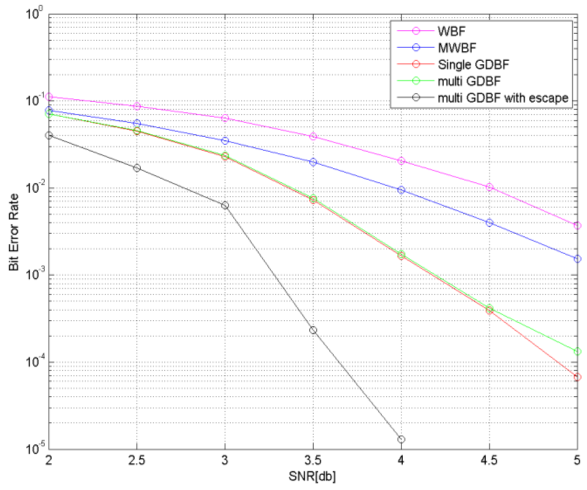

Fig 1.BER performance of bit flipping algorithms: regular LDPC code (PEGReg504 × 1008 [8])

In figure 2 the best performed multi GDBF with escape is compared with NGDBF algorithm like SNGDBF (Lmax=100), MNGDBF (Lmax=100,θ=0.9,λ_0=-0.9) and ATBF (Lmax=100, θ=0.25,λ_0=-10). From the figure 2 it can be seen that SNGDBF performance is less than that of multi GDBF with escape but the MNGDBF perform better.

In figure 3 the SM-NGDBF gives better performance than the MNGDBF but it requires 300 iterations. For this algorithm initial threshold λ_0=-0.9, for SNR<3.5db adaptation parameter θ=0.99, SNR=3.5db θ=0.97 and 0.94 at 4db. The performance of SM-NGDBF is very close to sum product algorithm.

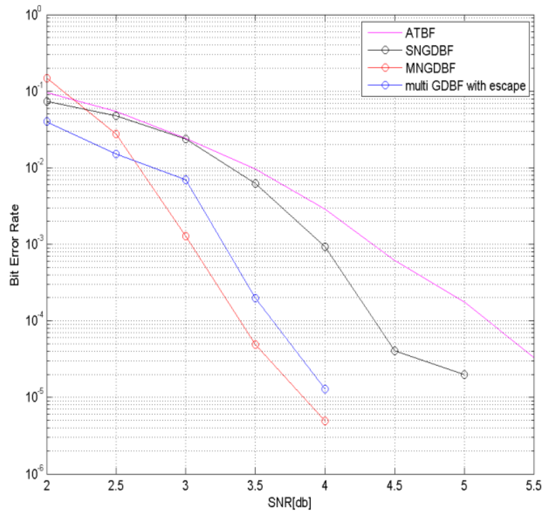

Fig 2.BER performance of bit flipping algorithms: regular LDPC code (PEGReg504 × 1008 [8])

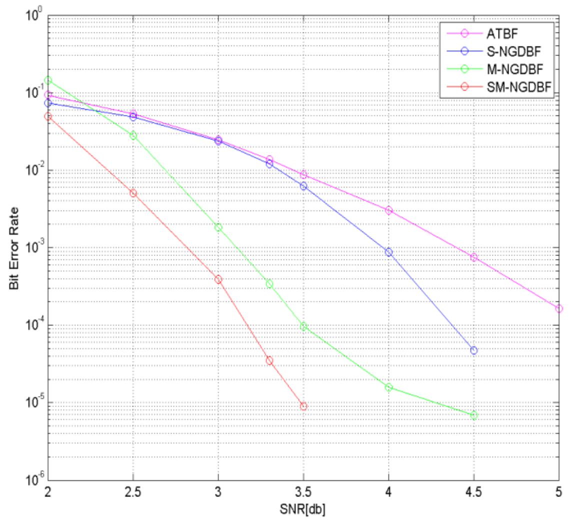

Fig 3.BER performance of bit flipping algorithms: regular LDPC code (PEGReg504 × 1008 [8])

B. Average Number of Iterations:

Figure 4 shows the average number of iterations as a function of SNR. For algorithm SM-NGDBF and multi GDBF with escape the maximum number of iterations Lmax is 300 and for other algorithm Lmax=100. From the figure4 it can be shown that M-NGDBF algorithm require least number of iteration than the other algorithms to decode a codeword. Thus by using threshold adaptation method in M-NGDBF algorithm it facilitates in high throughput decoding and its BER performance is also good.

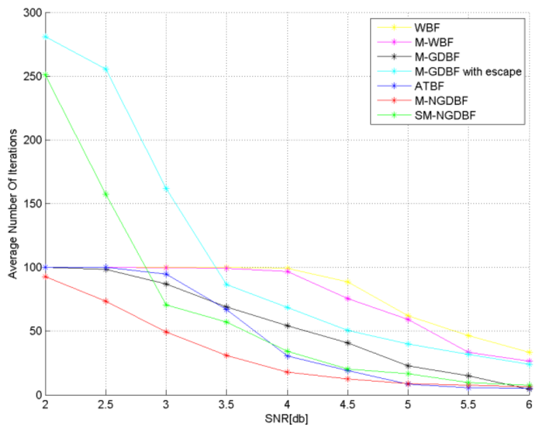

Fig 4.Average number of iterations of bit flipping algorithms: regular LDPC code (PEGReg504×1008 [8])

IV.CONCLUSION:

The present paper proposed the comparison between different bit flipping algorithms. From the simulation results it was shown that the SM- NGDBF performed the best. In NGDBF a random perturbation is added to each symbol metric at each iteration. Due to this random perturbation the algorithm escape from undesirable local maxima and results in improved performance.

REFERENCES

(1) D. J. C .MacKay and R.M. Neal, "Near Shannon limit performance of low density parity check codes," *IEEE Electron. Letter.*, vol. 33, no. 6, pp. 457–458, Mar. 1997.

(2) Y. Kou, S. Lin, and M. Fossorier , "Low-density parity-check codes based on finite geometries: A rediscovery and new results," *IEEE Trans. Inf. Theory*, vol. 47, no. 7, pp. 2711–2736, Nov. 2001.

(3) R. G. Gallager, "Low-density parity-check codes," in Research Monograph Series. Cambridge, MA: MIT Press, 1963.

(4) J. Zhang, and M. P. C. Fossorier, "A modified weighted bit-flipping decoding of low-density parity-check codes," *IEEE Commun. Lett.*, pp. 165-167, vol. 8, Mar. 2004.

(5) T. Wadayama *et al.*, "Gradient descent bit flipping algorithms for decoding LDPC codes," *IEEE Trans. Commun.*, vol. 58, no. 6, pp. 1610–1614, Jun. 2010.

(6) M. Ismail, I. Ahmed, J. Coon, S. Armour, T. Kocak, and J. McGeehan, "Low latency low power bit flipping algorithms for LDPC decoding," in *Proc. 2010 IEEE Int. Personal Indoor and Mobile Radio Communications Symp.*, pp. 278-282.

(7) Gopalakrishnan Sundararajan, Chris Winstead, and Emmanuel Boutillon, "Noisy Gradient Descent Bit-Flip Decoding for LDPC Codes," IEEE transactions on communications, vol. 62, no. 10, october 2014

(8) D. J. C. MacKay, "Encyclopedia of sparse graph codes." [Online].Available:http://www.inference.phy.cam.ac.uk/mackay/codes/data.html