Low Complexity Gradient Descent Weighted Bit Flipping Algorithm for LDPC Codes

J. Jeevitha, D. Monica Marylene, J. Felicita Celestine, R. Joan Kiruba, S.R. Evangelin
Karunya Institute of Technology and Sciences, Coimbatore, India

***Abstract*- Low Density Parity Check (LDPC) codes belong to linear block codes. This paper comprises of low complexity gradient descent bit flipping algorithm for decoding low density parity check codes which deals on hard decision decoding algorithm, soft decision decoding algorithms. The Proposed low complexity Gradient Descent Bit Flipping (LCGDBF) algorithms exhibit decoding performance better than other Bit flipping algorithms for decoding LDPC codes. By varying the check nodes, low complexity gradient descent bit flipping algorithm helps to gain high performance with minimum error rate. The proposed algorithm achieves a coding gain improvement in the range of 1-1.2dB for BER = 10^{-5} with less computational complexity compared with other bit-flipping algorithms.**

Keywords- LDPC, Bit flipping, Bit Error Rate (BER), Signal to Noise ratio (SNR)

I.INTRODUCTION

Low density parity check codes are linear error correcting codes which are used for perfect data transmission in communication network. These codes are found by Robert. G. Gallager. So, it is also called as Gallager codes. These are capacity approaching codes. Turbo codes are also another class of capacity approaching codes which are used in Deep space network and satellite communications. But due to the advancement in low-density-parity-check codes, it surpassed the turbo codes in the performance range for higher code rate and error correction. Whereas turbo codes are used only for lower code rates.

LDPC codes are represented in the graphical mode called Tanner graph [1]. It is a sparse parity check matrix with less number of ones compared to zeros. It is constructed using sparse bipartite graph. LDPC codes are used in 10GBase-T Ethernet, DVB-S2 standard for satellite transmission, Wi-max, Digital video broadcasting etc., LDPC decoding algorithms are iterative in nature. The decoding algorithm which we choose determines the cost and performance of LDPC codes.

In regular LDPC code, every code digit is contained in the same number of equations and every equation contains the same number of code symbols. An irregular LDPC code doesn't follow these conditions. The Tanner graph has two types of nodes: bit nodes and parity nodes. All bit nodes represents a code symbol and all parity node represents a parity equation. The iterating procedure is based on the interchange of messages between the Bit nodes and Check nodes in the Tanner graph. The process of iterating stops when a proper code word is received or maximum number of iterations is completed. A simple iterative decoder can be formed by considering each Bit node and Check node of the Tanner graph as processing units. LDPC has two types of decoding algorithms. They are hard decision decoding algorithm and soft decision decoding algorithm.

LDPC codes can be decoded by brief-propagation (BP) and bit flipping (BF) algorithms [1]. Whereas Brief propagation can achieve good performance only with large number of calculations. Various Bit flipping decoding are presented to achieve good tradeoff between performance and complexity. The parity check matrix consists of only few number of 1's compared to 0's. LDPC codes are now very much popular because of its error correcting performance and wide number of applications [7]. Gradient descent bit flipping (GDBF) provides better performance than other bit flipping algorithms [5]. It provides better performance and low complexity.

This proposed algorithm consists of power efficient LDPC encoders and decoders with low bit error rate (BER) and low signal-to-noise ratio (SNR). There are different bit flipping algorithms. The Weighted Bit Flipping algorithms (WBF) have good performance and complexity [3]. In decoding iteration the inversion function is implemented on the symbol nodes for the bits which needed to be flipped in weighted bit flipping algorithms. There are various WBF algorithms. One of the varieties is Gradient descent bit- flipping (GDBF) algorithm. The GDBF algorithm provides good performance with very low complexity.

The low complexity gradient descent bit flipping (LCGDBF) has low column weight. LCGDBF uses syndrome weight and syndrome information.

Syndrome weight is used for detecting the decoding loop and then the bits are flipped. Syndrome information updates the reliability of flipped bit nodes. In the process of decoding the bits with error i.e., the bits which do not satisfy the parity checks are flipped for each iteration. In single bit flipping only one bit is flipped for each iteration whereas in multi bit flipping all bits below the given threshold are flipped which results in fast operation. Low complexity gradient descent is better than GDBF and Noisy GDBF. Low complexity Gradient Descent Bit-Flipping (LCGDBF) algorithm has better error correcting ability than weighted bit flipping (WBF) and Modified WBF and has average number of iterations. It also has better performance than any other bit flipping algorithms.

II.PRELIMINARIES

LDPC codes can be represented by a bipartite Tanner graph which was proposed by Tanner in 1981.The Tanner graph consists of two sets of vertices: n vertices for the codeword bits(called variable nodes) and k vertices for the parity-check equations(called check nodes)

An edge joins a variable node and a check node if that bit is included in the corresponding parity-check equation and so the number of edges in the Tanner graph is equal to the number of ones in the parity-check matrix.

$$\mathbf{H} = \begin{array}{c} \begin{matrix} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 & v_7 \end{matrix} \\ \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \end{array} \begin{matrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{matrix}$$

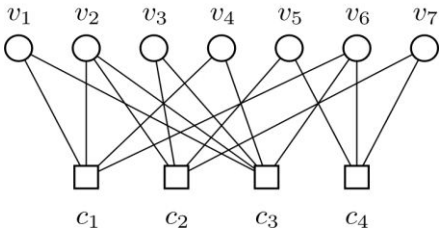

Fig1. Tanner graph

Let the variable nodes that are participated in the mth check node is represented as N(m) and the check nodes that are participated in the nth variable is represented as M(n). The K_{th} iteration notations for the decoding process are as follows:

$H_{m,n}$: mth row and nth column of the parity check matrix.

Z(k)n : nth received sequence (hard decision).

R(k)n : The Reliability measure of nth received sequence.

e(k)n : Total extrinsic information contributed to the received bit *z(k)n* Excluding bits that participate in the checksum.

e(k)m,n : Extrinsic information contributed to the received bit *z(k)n* Including bits that participate in the checksum.

λ : Quantization factor.

dc : Check node degree (row weight).

dv : Variable node degree (column weight).

R*c*: Code rate.

For a (n,k) LDPC code H, where dc and dv is small compared to code's block length N and code's information length, Let c=$(C_0,C_1\ldots,C_{N-1})\in\{GF(2)\}^N$ represent the code word c, Before transmission that is mapped into the bipolar sequence x=$(x_0,x_1,\ldots,x_{N-1})$, where x_n =(2(n-1) with $0 \leq n \leq$ N-1. Let y=$(y_0,y_1,\ldots,y_{N-1})$ at the receiver received as the soft decision. For $0\leq n\leq$N-1, $y_n=x_n+w_n$, where w_n is Gaussian random variable with zero mean and variance σ^2 that is independent of x_n. Let AWGN has N_0 /2 power spectral density is given as σ^2 =(2Rc. E_b / $N_0)^{-1}$. Received sequence corresponding binary hard decision is given by $Z^{(0)}=(Z_0^{(0)},Z_1^{(0)},\ldots,Z_{N-1}^{(0)})$. Then the hard decision sequence $Z^{(0)}$ syndrome is denoted by S=$(s_0,s_1,\ldots,s_{M-1})$.

A. Hard Decision Decoding

For hard decision decoding, Bit flipping algorithm is the best example. In Bit flipping the message which is decoding would be binary that is different from Belief Propagation decoding. In hard decision the error bits are flipped till the given maximum iteration and the output is shown .The probabilities of incidence of the code word bits in Belief Propagation decoding constitute the message. In the Tanner graph the messages and the edges are passed together in Bit flipping Algorithm. The message node sends the message contain the information that the bits are zero which are available at the message node or one to the check node. By parity check equation this response is initiated which is based on the Modular sum of bits available at the check node is equal to zero. The steps involved in hard decision decoding are as follows:

Step1: All message nodes send a message to the check nodes which are connected to it. In this case, the message is the bit they believe to be correct for

the check nodes. Here 0 receives by c_2 and it will send f_2 and f_1. Like this to their corresponding check nodes all message nodes will send the messages.

Step2: Every check nodes using the messages they receive from step1 calculate a response to their connected message nodes. By using parity check equations which force all the message nodes to join to the particular check node to sum to 0.if sum of bits received is zero, will send back the same bit which they received from the message nodes. If it is not zero, the bit that is received from the message node will be flipped and send back.

Step3: The message nodes utilize the messages they received from the check nodes and they get from transmitter to determine by the majority rule, if the bit at their position is a 0 or 1. The message nodes to their connected check nodes will send the hard decision.

Step4: Repeat the step2 until either exit at the step2 or a assigned number of iterations has been passed.

B .Soft Decision Decoding

In decoding process of LDPC codes, soft decision gives good performance which is based on the theme of belief propagation. In soft scheme, the messages in the given received bit are a 1 or a 0 are the conditional probability. In soft decision all the error bits are used to flipped and the output is shown. The soft decision message-passing algorithm is the Sum-product algorithm. Priori probabilities for the received bits are the input probabilities as here because they were known in advance before the LDPC decoder has run . The bit probabilities that has returned by the decoder are called as the posterior probabilities.

The sum-product algorithm is the soft decision message-passing algorithm that is equal to the bit flipping algorithm but the variation is that the messages representing each decision in SPA with probabilities. On the received bit as input in bit-flipping decoding which accepts the sum product algorithm as a soft decision algorithm and accepts an initial hard decision that accepts the probability of each received bit as input.

C. Brief Review on Bit Flipping algorithms

A number of variants of Bit Flipping algorithms have been introduced. The BF algorithms have been divided to two classes (1) Single bit flipping algorithm (2) Multiple bit flipping algorithms. In the single Bit Flipping algorithm decoding process, according to bit flipping rule, only one bit can flipped. Other side, Multiple Bit Flipping iteration are allowed by the multi Bit Flipping algorithm in the decoding process. Multi Bit Flipping algorithm have faster convergence then single Bit Flipping algorithm. The multi Bit Flipping algorithm affect by the oscillation behavior of decode state that is not easy to control. The frame work of Single Bit Flipping algorithm as follows:

Step1: For j ∈ [1.n] , let X_j :=sign(Y_j) .let x ≜(x1,x2,….,xn)

Step2: If the parity equation $\pi_{j\in N(i)}$xj=+1 holds for all i∈[1,m] , output x , then exit.

Step 3: Find the flipping position given by

L:= arg $min_{k\in[1,n]}\Delta_{k(x)}$.

Then flip the symbol: xl :$_=$-xl. The function Δ_k(x) is called as an inversion function.

Step4: If the number of iteration is under the maximum number of iterations L_{max} , return to the step2, otherwise output x and then exit.

The function sign (.) is defined by

$$\text{Sign(x)} \triangleq \begin{cases} +1, x \geq 0 \\ -1, x < 0 \end{cases}$$

In a decoding process of the signal BF algorithm, for a given y hard decision is first performed and to the hard decision result x is initialized. The minimum of the inversion function for k∈ $[1, n]$ is Δ_k(x) is then found. An inversion function Δ_k(x) can be seen for the bit assignment on x_k as a measure of the in validness. The bit xl, where l of the inversion function, gives the smallest value.

The inversion functions of WBF are defined by

$$\Delta_k^{(WBF)}(x) \triangleq \sum_{i\in M(k)} \beta_i \pi x j_{j\epsilon N(i)}$$

The values β_i (i ∈ $[1, m]$) defined by $\beta_i \triangleq min_{j\in N(i)}|y_j|$ is the reliability of bipolar syndromes. In this case, the inversion function $\Delta^{(WBF)}$(x) gives the symbol assignment's in validness on x_k which is given by the sum of weighted bipolar syndromes.

The inversion function of MWBF is same like the inversion function of WBF but it has a term respect to a received symbol. The inversion function of MWBF is given by

$\Delta_k^{(MWBF)}(x) \triangleq \propto |y_k| + \sum i \in M(k)\, \beta_i \pi_{j \epsilon N(i)} x_j$

Where α is the positive real number.

1. Weighted Bit Flipping algorithm (WBF)

WBF algorithm considers the influence of parity information on the error metric. The weighted BF decoding (WBF) algorithm uses both the check relationships and the reliability of receive message, therefore obtains a better decoding performance when compared with BF algorithm. The performance loss due to loop oscillation occurs in weighted bit-flipping algorithm. In order to reduce this further a hybrid algorithm is proposed.

At the position n , for each message node , the WBF algorithm calculates the following :

$E_n = \sum_{m \in M(n)} (2sm - 1)\, y_m^{min}$

Where E_n is error term, which is used for probability evaluation that the bit position n would flipped. And on the next step of WBF algorithm, E_n *is* the highest error term and it will be regarded the least reliable bit, so it will be flipped.

2. Gradient descent bit flipping Algorithm

The decrease in the transistor size and the increase in integration factor throughout the past four decades, has led to the invention of semiconductor memories, and also very small and fast memory chips. Increase in the urge for higher memory capacity, read/write speed has led to the, wide acceptance that a sustainable progress can be only attained through very high energy-efficient transistors. We are proposing a new method with developing advanced error rectifying projects for fault-tolerant memories. This is based on codes, graphs and iterative decoding. That is this method is based on the gradient descent bit flipping Algorithm. The Gradient Descent Bit Flipping (GDBF) algorithms perform well than the WBF and MWBF algorithms in error correcting ability and more significantly in the number of average iterations which are required for successful decoding.

The partial derivative of f(x) respect to the variable X (K)∈[1,n] will be derived from the term f(x) :

$$\partial / \partial x_k \text{ f(x)} = y_k + \sum i \in M(k)\, \pi_{j \epsilon N(i) \backslash k} \text{xj}$$

Consider partial derivative of X_K and the product X_K in X given as:

$$X_K \partial / \partial x_k \text{f(x)} = x_k y_k + \sum i \in M(k)\, \pi_{j \epsilon N(i)} \text{xj}$$

In order to finding the position of flipping function, we have to choose the position where the absolute value of partial derivative is large.

III. GRADIENT DESCENT FORMULATION

The Gradient Descent Bit Flipping (GDBF) algorithms perform well than the WBF and MWBF algorithms in error correcting ability and more significantly in the number of average iterations which are required for successful decoding.

A. Objective function

The dynamics of a BF algorithm is the minimization process of a hidden objective function. This observation finally grows out to be a gradient descent formulation of BF algorithms.

The objective function is defined as follows,

$\text{f(x)} \triangleq \sum_{\text{i=1}}^{\text{n}} \text{x}_\text{j}\, \text{y}_\text{j} + \sum_{\text{i=1}}^{\text{m}} \prod_{\text{j} \epsilon \text{N(i)}} \text{x}_\text{j}$

This objective function is a non-linear function and also it has many local maxims, which later becomes the sub-optimality of GDBF algorithm.

The partial derivative of f(x) can be immediately derived from the definition of

$$\frac{\partial}{\partial x_k} f(x) = y_k + \sum_{i \in M(k)} \prod_{j \in N(i) \backslash k} x_j$$

Let us consider the product of x_k and the partial derivative of x_k in x, namely
X

$$x_k \frac{\partial}{\partial x_k} f(x) = x_k y_k + \sum_{\text{i} \epsilon \text{M(k)}} \prod_{\text{j} \epsilon \text{N(i)}} \text{x}_\text{j}$$

B. Behavior of Modified GDBF Algorithms

1. Low complexity GDBF Decoding

This single improved decoding is generally denoted as S-MGDBF. The performance is better in this decoding process. The loop detection based on the syndrome weight, and it also iteratively updates the mechanism based on the syndrome information. The stages involved in this method are defined by:

Step 0: At first the iteration number is set as k=0, the maximum iteration number is set as k_{max}, syndrome weight factor is set as α , excluding set B=∅,Maximum size of the excluding set is $\Delta = 2$

Step 1: Syndrome vector is calculated using the formula $s^{(k)} = z^{(k)} H^T$. If the value of $wt\left(s^{(k)}\right) =$

0 or $k = k_{max}$ stop the procedure and extract the output of $z^{(k)}$.Or else take the value of $k = k + 1$.

Step 2: Calculate for each variable node n∈ v , $E*_n$= $\sum m \in M(n)$(1-2$S^{(k)}$m) and E_n= (2$Z_n^{(K)}$-1)y_n+$E*_n$.

Step 3: Find the value of the variable node n* by using the formula n*=arg $\min_{n \in V/B} E_n$, then flip this bit n* and update the y_n* by using the formula y_n*=y_n*+$\alpha(2Z_{n*} - 1)E*_{n*}$

Step 4: Record the position of n^* if wt ($s^{(k)}$)≤wt($S^{(k-1)}$) . If the position of n^* is the same with the previous one , set B=B∪ $\{n^*\}$. If |B|≥ Δ , set B=∅ and Δ=Δ + 1 . Then return to step1.

IV. Simulation Results

The decoding performance of the various bit flipping algorithms are illustrated in Fig. 2. Comparison of different algorithms with MacKay (1008,504) regular LDPC code is performed for 100 decoding iterations. The decoding performance of Weighted Bit Flipping algorithm, Gradient Descent bit flipping algorithm and Low Complexity Gradient Descent Weighted Bit Flipping algorithm are compared and was found that the proposed low complexity gradient descent weighted bit flipping decoding algorithm has higher performance and low complexity compared to other bit flipping algorithms.

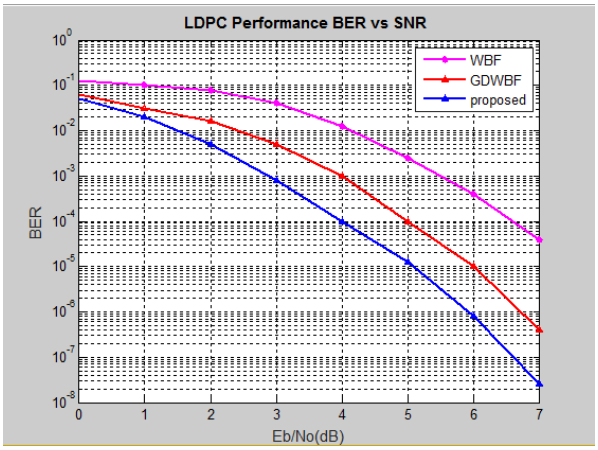

Fig 2. Comparision of different algorithms with MacKay (1008,504) regular LDPC codes for 100 decoding iterations

Figure 2 shows the decoding performance of rate 0.5 MacKay (1008, 504) code with Bit Error Rate of 10^{-8} and signal to noise ratio of 7.2dB. The simulation results shows the proposed decoding algorithm which exhibits better decoding performance in 100 iterations compared to other bit flipping algorithms.

Table 1. Overall decoding performance comparison between various BF algorithms

CODE LENGTH	Comparison of SNR required to achieve BER of 10−5		
	PROPOSED	GDBF	WBF
(1008,504)	5	6	7.2

Table 2. Coding gain improvement of the proposed algorithm with WBF algorithm and its variants at BER of 10^{-5}

CODE LENGTH	Code gain achieved by the proposed algorithm for a BER of 10^{-5}	
	GDBF	WBF
(1008,504)	1	1.2

The overall comparison of the decoding performance and code gain of the proposed decoding algorithm with various Bit Flipping algorithms are tabulated in table 1 and 2. It is noted that the proposed algorithm requires comparatively lesser signal to noise ratio to achieve a BER of 10^{-5} and also table 2 proves that the proposed decoding algorithm can achieve over all code gain in the range of 1 to 1.2 db when compared with other bit flipping algorithms.

V. Conclusion

This paper deals with the performance of a low complexity gradient descent weighted bit-flipping algorithm. This algorithm is studied and analyzed based on comparing the decoding performance of different bit flipping algorithms with the proposed algorithm and noting their BER and SNR. It is found that the proposed decoding algorithm has less BER and SNR compared with other bit flipping algorithms and also it has better performance with low complexity when compared with others. Simulation results proves that the proposed algorithm achieves a coding gain improvement in the range of 1-1.2dB for BER = 10^{-5} with less computational complexity compared with other bit-flipping algorithms. This proposed algorithm takes only less decoding time which helps to maintain reliability and high speed decoding of data transmitted in a noisy channel. Because of its less decoding time it has wide range of applications.

References

[1] Hua Li,Hong Ding,Linhua Zheng, “Modified Gradient Descent Bit-Flipping Decoding for Low- Density Parity-Check Codes,”

Wireless Pers Commun, Springer Science+Business Media New York, 201796:6459–6472 DOI 10.1007/s11277-017-4486-7, May. 2017.

[2] Tadashi Wadayama, Keisuke Nakamura, Masayuki Yagita, Yuuki Funahashi, Shogo Usami, and Ichi Takumi, " Gradient Descent Bit Flipping Algorithms for Decoding LDPC Codes," IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. 58, NO. 6, JUNE 2010.

[3] Danilo P. Mandic, "A Generalized Normalized Gradient Descent Algorithm," IEEE SIGNAL PROCESSING LETTERS, VOL. 11, NO. 2, FEBRUARY 2004.

[4] Tharathorn Phromsa-ard, Jiratchaporn Arpornsiripat, Jutaphet Wetcharungsri, Paramin Sangwongngam and Keattisak Sripimanwat, Pisit Vanichchanunt, "Improved Gradient Descent Bit Flipping Algorithms for LDPC decoding ".

[5] Boorle Ashok Kumar, G Y. Padma Sree, "COMPARISON OF SIMPLIFIED GRADIENT DESCENT ALGORITHMS FOR DECODING LDPC CODES," International Journal of Advanced Technology in Engineering and Science, Volume No.02, Issue No. 12, pp. 2348 – 7550 ,December 2014.

[6] Pavithra G, Naveen Kumar M, "BER Performance Comparison of Bit Flipping Algorithms used for Decoding of LDPC Codes," International Journal of Engineeering Research &Technology(IJERT), Vol. 4 Issue 05, pp. 2278-0181,May-2015.

[7] Ranjitha CR, Jeena Thomas, Chithra KR, "A brief study on LDPC codes," International Journal of Engineering Research and General Science Volume 4, Issue 2, pp. 2091-2730, March-April, 2016.

[8] Tasnuva Tithi, Chris Winstead and Gopalakrishnan Sundararajan, "Decoding LDPC codes via Noisy Gradient Descent Bit-Flipping with Re-Decoding," arxiv:1503.08913v1 [cs.IT], 31 Mar 2015.

[9] R. G. Gallager, "Low-density parity-check codes," in Research Monograph Series. Cambridge, MA: MIT Press, 1963.

[10] F. Guo and H. Henzo, "Reliability ratio based weighted bit-flipping decoding for low-density parity-check codes," IEEE Electron. Lett., vol. 40, no. 21, pp. 1356-1358, 2004.