COMPARISON OF SIMPLIFIED GRADIENT DESCENT ALGORITHMS FOR DECODING LDPC CODES

Boorle Ashok Kumar[1], G Y. Padma Sree[2]

[1]*PG Scholar, Al-Ameer College Of Engineering & Information Technology, Anandapuram, Visakhapatnam, (India)*

[2]*Assistant Professor, Dept of ECE, Al-Ameer College Of Engineering & Information Technology, Anandapuram, Visakhapatnam, (India)*

ABSTRACT

In this paper it is shown that multi GDBF algorithm exhibits much faster convergence as compared to the single GDBF algorithm. The multi GDBF algorithm require less iterations when compared to the single GDBF algorithm for the search point to closely approach the local maximum point taking into consideration the gradient descent bit flipping (GDBF) algorithms exhibiting better decoding performance than known BF algorithms. A novel class of bit-flipping (BF) algorithm for decoding low-density parity-check (LDPC) codes is presented. The behaviour of the proposed algorithms can be explained from the view point of optimization of the objective function. The GDBF algorithm with escape process performance very well compare with known BF algorithms, such as weighted BF (WBF) algorithms. A natural relationship between weighted bit-flipping (WBF) decoding and message-passing decoding is explored. This understanding can help us develop a dual WBF decoding algorithm from one type of message-passing decoding algorithm and vice versa. The GDBF algorithms can be regarded as a maximization process of the object function using the bit-flipping gradient descent method (i.e., bit-flipping dynamics, which minimizes the energy. The dynamics of the decoder is naturally derived by differentiating the energy function, and the gradient descent formulation introduces an energy landscape of the state-space of the BF-decoder.

Keyword: GDBF, BF, MWBF, LDPC Code.

I. INTRODUCTION

The gradient descent bit flipping (GDBF) algorithms for decoding low-density parity-check (LDPC) codes [1] shows better decoding performance than known BF algorithms, such as weighted BF (WBF) algorithm [2], the modified weighted BF (MWBF) algorithm [3] and other BF variants [4], [5], [6], have been proposed. The first BF algorithm was developed by Gallager [1]. In decoding process of Gallager's algorithm, some reliable bits (in a binary quantized received word) corresponding to unsatisfied parity checks are flipped for each iteration. The successors of Gallager's algorithm, namely, find reliable bits and then flip them. In general, the bit error rate (BER) performance of the BF algorithm is inferior to that of the sum-product algorithm or the min-sum algorithm, in general, the BF algorithm enables the design of a much simpler decoder, which is easier to implement. Thus, bridging the performance gap between BF decoding and belief propagation (BP) decoding is an important technical challenge. In this, a novel class of BF algorithm for decoding LDPC codes is proposed. The proposed algorithms, which are referred to as gradient descent bit flipping (GDBF) algorithms, can be regarded as bit flipping gradient descent algorithms. These algorithms are naturally derived from a simple

gradient descent formulation. The behaviour of the proposed algorithms can be explained from the viewpoint of the optimization of a non-linear objective function. Multi-bits flipping GD-BF algorithm gives much fast convergence. At the expense of error correcting performance, bit flipping methods allow for significantly less complex decoding because they flip one or more bits at a time based on some objective function. A simple bit flipping technique can be described as follows. Let **y** be the soft-decision received vector and **x** be the binary hard decision of the vector **y**, then the *syndrome*, **s**, is defined as

$$\mathbf{s}=\mathbf{x}H^{y}$$

where, **H** is a M × N parity check matrix and **s** = (s_1, s_2, . . . , s_M) represents the M individual parity checks. Bit flipping looks at the parity checks each received bit is involved in, if the total number of check failures the bit is involved in exceeds some threshold, the sign of the received bit is inverted to form an updated received vector. The next iteration of the algorithm will re-calculate the syndrome vector using the updated hard-decision vector. This process is repeated until all checks are error free or the number of iterations reaches some preset threshold. There are many variants to the basic bit flipping algorithm most notably the *Weighted Bit Flipping* (WBF) algorithm, which in formulating a count of failed checks adds a weighting factor based on the magnitude of the received sample. The *Modified weighted* BF (MWBF) algorithm introduces an additional scaling factor, -f(x), for the soft-decision values in improved error correcting performance. Recently, Wadayama formulated a new way of calculating the decision metric based on the gradient descent formulation. The Gradient Descent Bit Flipping (GDBF) algorithms outperform the WBF and MWBF algorithms in error correcting ability and more significantly in the number of average iterations needed for successful decoding. The BF (bit flipping) decoding algorithm is discussed in [7]. It is described clearly about the weighted bit flipping (WBF), modified weighted bit flipping (MWBF) algorithm and gradient descent bit flipping (GDBF) algorithm.

II. GRADIENT DESCENT BF ALGORITHM

In general, although the multi BF algorithm exhibits faster convergence than the single BF algorithm, the multi BF algorithm suffers from the oscillation behaviour of a decoder state, which is not easy to control. The frame work of the single BF algorithm is summarized as follows:

Step 1 : For $j \in [1, n]$, let $x_j := \text{sign}(y_j)$, where $\text{sign}(x) \triangleq +1$ if $x < 0$.

Otherwise $\text{sign}(x) \triangleq -1$. Let $x \triangleq (x_1, x_2, x_3, \ldots \ldots \ldots, x_n)$.

Step 2 : If the parameter equation $\prod_{j\in N(i)} x_j = +1$ holds for all i ϵ[1,m], output x, and then exit.

Step 3 : Find the flip position given by $\ell := argmin_{k\in[1,n]} \Delta_k(x)$ and then flip the symbol : $x_\ell := -x_\ell$. The function $\Delta_k(x)$ is referred to as an inversion function.

Step 4 : If the number of iterations is less than the maximum number of iterations L_{max}, then return **to** Step 2**;** otherwise output x and exit.

Both the BF algorithms reviewed in the reference [7] and the GDBF algorithm flip only one bit for each iteration. In terms of the numerical optimization, in these algorithms, a search point moves toward a local maximum with a very small step (i.e., 1 bit flip) in order to avoid oscillation around the local maximum (See Fig.1(A)). However, the small step size leads to slower convergence to a local maximum. The multi bit flipping algorithm is expected to converge faster than the single bit flipping algorithm because of its larger step size. If the search point is close to a local maximum, a fixed large step is not suitable for finding a (nearby) local

maximum point and leads to oscillation in a multi-bit flipping BF algorithm (Fig.1(B)). It is necessary to adjust the step size dynamically from a large step size to a small step size in an optimization process (Fig.1(C)).

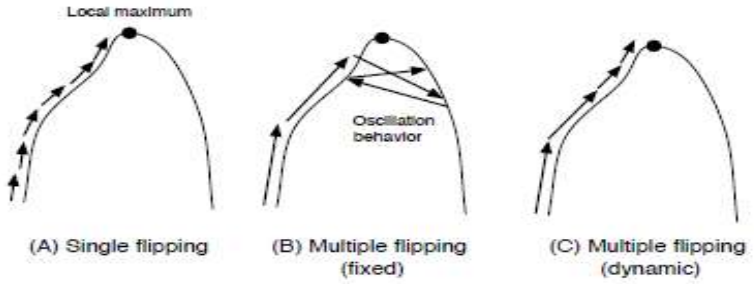


Fig.1. (A) converging but slow, (B) not converging but fast, (C) converging and fast

The objective function can adjust the step size (i.e., number of flipping bits). The multi GDBF algorithm is a GDBF algorithm that incorporates the multi-bit flipping concept. The inversion function of GDBF algorithm (single BF algorithm) $\Delta_k^{(GD)}(x)$ is defined as

$$\Delta_k^{(GD)}(x) \triangleq x_k y_k + \sum_{i \in M(k)} \prod_{j \in N(i)} x_j \qquad (1)$$

To define the multi GDBF algorithm, we used two parameters θ and μ.where the parameter θ is the inversion threshold and the variable μ is the mode flag. At the beginning of the decoding process, the mode flag μ is set to 0 .Step 3 of the BF algorithm should be replaced with the following multi-bit flipping procedure:

Step 3 Evaluate the value of the objective function, and let $f_1 := f(\boldsymbol{x})$. If μ = 0, then execute Sub-step 3-1 (multi-bit mode), else execute

Sub-step 3-2 (single bit mode).

3-1 Flip all bits satisfying $\Delta_k^{(GD)} < \theta$ $(k \in [1, n])$ Re-evaluate the value of the objective function, and let $f_2 := f(\boldsymbol{x})$. If $f_1 > f_2$ holds, then let μ = 1.

3-2 Flip a single bit at the jth position where $j \triangleq argmin_{k \in [1,n]} \Delta_k^{(GD)}$.

The multi GDBF algorithm shows much faster convergence than the single GDBF algorithm. It suffers from oscillations where single GDBF algorithm has no oscillation. It is not easy to control when compare to the single GDBF algorithm. A local maximum captures an untransmitted codeword when decoding failure occurs. Since the weight of the final position of a search point is so small, a small perturbation of a captured search point appears to be helpful for the search point to escape from an undesirable local maximum. Such a perturbation process can be expected to improve the BER performance of BF algorithms. to multi bit mode with an appropriate threshold to a trapped search point arrived at a non codeword local maximum is the simplest way for adding perturbation in the escape process. In general, the escape process reduces the object function value. After the escape process, the search point again begins to climb a hill, which may be different from that of the trapped search point.

Here, we modify the multi GDBF algorithm with the reference of GDBF algorithm [7].

III. PROCESS OF ESCAPE FROM A LOCAL MAXIMUM

A decoding failure occurs when a search point is captured by a local maximum, which is not a transmitted codeword. Since the weight of the final position of a search point is so small, a small perturbation of a captured

search point appears to be helpful for the search point to escape from an undesirable local maximum. Such a perturbation process can be expected to improve the BER performance of BF algorithms. One of the simplest ways to add a perturbation to a trapped search point is to forcibly switch the flip mode from the single-bit mode to the multi-bit mode with an appropriate threshold when the search point arrives at a non codeword local maximum. This additional process is referred to as the *escape process*. In general, the escape process reduces the object function value, i.e., the search point moves downward in the energy landscape. After the escape process, the search point again begins to climb a hill, which may be different from that of the trapped search point. Here, we modify the multi GDBF algorithm by incorporating two thresholds: $\theta 1$ and $\theta 2$. The threshold $\theta 1$ is the threshold constant used in the multi-bit mode at the beginning of the decoding process. After several iterations, the multi-bit mode is changed to the single-bit mode, and the search point may then eventually arrive at the non codeword local maximum. In such a case, the decoder changes to the multi-bit mode (i.e., $\mu = 0$) with threshold $\theta 2$. Thus, the threshold $\theta 2$ can be regarded as the threshold for *downward movement*. Although $\theta 2$ can be a constant value, in terms of the BER performance, it is advantageous to choose randomly6. In other words, $\theta 2$ can be a random variable. After the downward move (only one iteration), the decoder changes the threshold back to $\theta 1$. The above process continues until the parity check condition holds or the number of iterations becomes $Lmax$. Figure 2 illustrates the concept of the escape process.

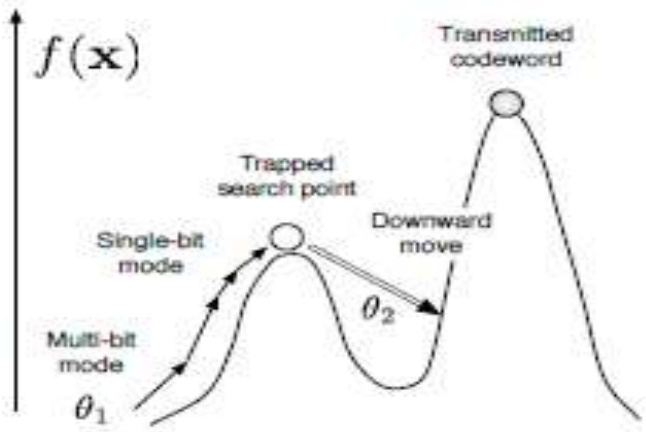


Fig.2. Idea of Escape Process

IV. SIMULATED RESULTS

The behaviour and decoding performance of (single and multi) GDBF algorithms obtained from computer simulations are presented.

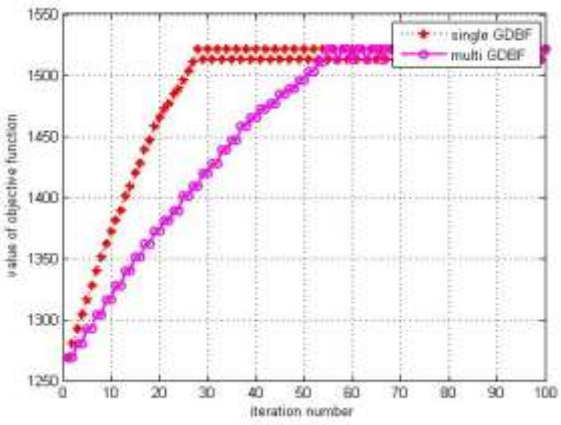
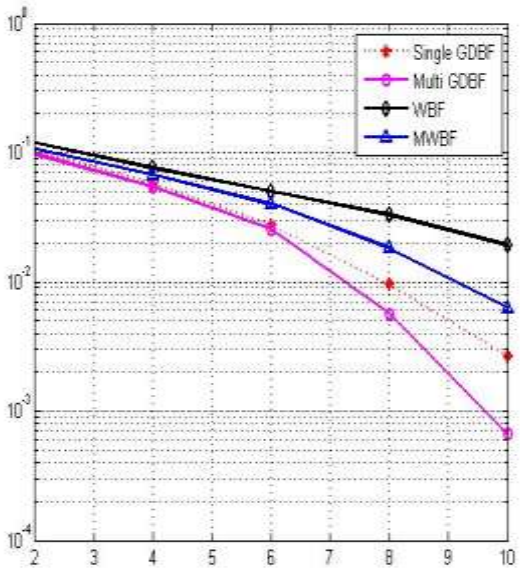


Fig.3. Objective function values in GDBF processes as a function of the number of iteration (single GDBF v.s. multi GDBF)



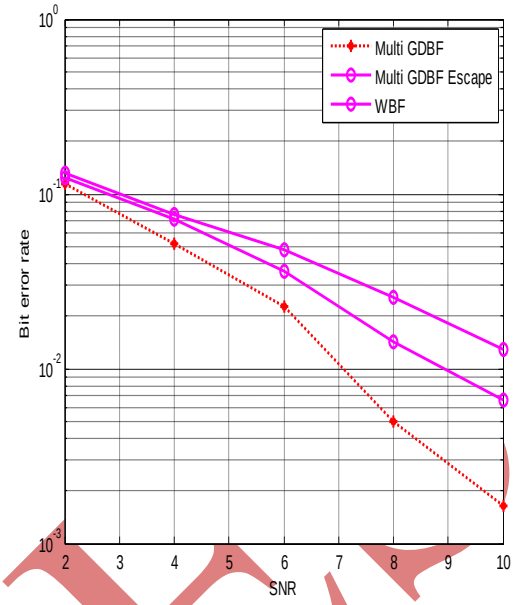


Fig.4.Bit Error Rate Of GDBF Algorithms And Other BF Algorithms

Fig.5. Bit Error Rate Of The GDBF Algorithm With The Escape Process

Figure 5 shows the BER curve for such a decoding algorithm (labelled 'Multi GDBF with escape').

V. ANALYSIS AND CONCLUSION

The BER and SNR status of multi GDBF, multi GDBF escape and WBF. This paper presents multi GDBF algorithm much faster convergence as compared to single GDBF algorithm. In this multi GDBF algorithm requires less iterations to search local maximum. The GDBF algorithm shows better decoding performance than known BF algorithms for decoding LDPC codes. The gradient descent formulation introduces an energy landscape of the state-space of the BF-decoder. The viewpoint obtained by this formation brings us a new way to understand convergence behaviour of BF algorithms. Further this viewpoint is also useful to design improved decoding algorithm such as multi GDBF algorithm and the GDBF algorithm with escape process from an undesired local maximum. The GDBF algorithm with escape process performance very well compared with known BF algorithms. One lesson we learned from this result is that fine control an flipping schedule is indispensable to improve decoding performance of BF algorithms. The GDBF algorithms show better decoding performance than known BF algorithms, such as the weighted BF algorithm and the modified weighted BF algorithm for several LDPC codes.

REFERENCES

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

[2] 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. Inf. Theory, pp. 2711–2736, vol. 47, Nov. 2001.

[3] 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.

[4] M. Jiang, C. Zhao, Z. Shi, and Y. Chen, "An improvement on the modified weighted bit flipping decoding algorithm for LDPC codes," IEEE Commun. Lett., vol. 9, no. 9, pp. 814-816, 2005.

[5] 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.

[6] C. H.Lee, and W.Wolf, "Implmentation-efficient reliability ratio based

[7] weighted bit-flipping decoding for LDPC codes," IEEE Electron. Lett., vol. 41, no. 13, pp. 755-757, 2005

[8] T. Wadayama, K. Nakamura, M. Yagita, Y. Funahashi, S. Usami, and I. Takumi, "Gradient descent bit flipping algorithms for decoding LDPC codes," in Proc. International Symposium on Information Theory and Its Applications ISITA 2008, 2008, pp. 1–6.