Optimized Min-Sum Decoding Algorithm for Low Density Parity Check Codes

Mohammad Rakibul Islam, Dewan Siam Shafiullah, Muhammad Mostafa Amir Faisal, Imran Rahman
Dept. of Electrical & Electronic Engineering, Islamic University of Technology
Boardbazar, Gazipur-1704, Dhaka, Bangladesh

***Abstract* — Low Density Parity Check (LDPC) code approaches Shannon–limit performance for binary field and long code lengths. However, performance of binary LDPC code is degraded when the code word length is small. An optimized min-sum algorithm for LDPC code is proposed in this paper. In this algorithm unlike other decoding methods, an optimization factor has been introduced in both check node and bit node of the Min-sum algorithm. The optimization factor is obtained before decoding program, and the same factor is multiplied twice in one cycle. So the increased complexity is fairly low. Simulation results show that the proposed Optimized Min-Sum decoding algorithm performs very close to the Sum-Product decoding while preserving the main features of the Min-Sum decoding, that is low complexity and independence with respect to noise variance estimation errors.**

Keywords* — *LDPC codes; Min-sum algorithm; Normalized min-sum algorithm; Optimization factor.

I. INTRODUCTION

Among the error correction codes, Low Density Parity Check (LDPC) is one of the most efficient techniques. It was first introduced by Robert Gallager in 1962 in his PhD. Dissertation [1]. It is the extreme sparseness of the parity check matrix for LDPC codes that make the decoding particularly attractive. LDPC codes have recently received a lot of attention because they can achieve a remarkable performance near Shannon limit over the binary symmetric channel (BSC) as well as the additive white Gaussian noise (AWGN) channel [2]. The decoding of an LDPC code allows a high degree of parallelism, which makes it very suited for high data rate applications such as wide-band wireless multimedia communications and magnetic storage systems [3], [4]. The low-density nature of the parity check matrix thus contributes both to good distance properties and the relatively low complexity of the decoding algorithm [5]. Well-designed irregular LDPC codes demonstrate better performance than regular ones [6].

Among a variety of decoding algorithms, the well-known Sum Product (SP) algorithm [7] achieves a good decoding performance but requires a large hardware complexity. There are alternative methods such as several kinds of Min-Sum (MS) algorithms which can significantly reduce the hardware complexity of SP at the cost of acceptable performance degradation where complex computations at the check nodes are approximated by using simple comparison and summation operations. Recently, the modified MS algorithms using correction factors have been preferred for many practical applications since they offer comparable decoding performance compared to that of SP [7] for regular LDPC codes [8], [9]. Also, for irregular LDPC codes, the improved normalized or offset MS algorithms exhibit small performance degradations [10], [11]. Specifically, the offset MS algorithm has been implemented for several practical applications due to its better performance and simple computations.

The main decoding algorithms of LDPC codes include soft-decision such as Sum Product (SP) algorithm [7] and hard-decision such as Bit flipping. In iterative decoding, a critical tradeoff between "complexity" and "performance" is required. Based on these two issues, LDPC codes may be classified as optimal, sub-optimal or quasi-optimal. The optimal iterative decoding is performed by the Sum- Product algorithm [7] at the price of an increased complexity, computation instability, and dependence on thermal noise estimation errors. The Min-Sum algorithm [12] performs a suboptimal iterative decoding, less complex than the Sum-Product decoding. The sub-optimality of the Min-Sum decoding comes from the overestimation of check-node messages, which leads to performance loss with respect to the Sum-Product decoding. Several correction methods were proposed [13-15] in the literatures in order to recover the performance loss of the Min-Sum decoding with respect to the Sum-Product decoding which are called quasi-optimal algorithms. An example is Normalized min-sum algorithm proposed by Chen and Fossorier [16]. In this paper, we propose an optimized min-sum algorithm which has better performance not only from min-sum algorithm but also from normalized min-sum algorithm.

The rest of the paper is organized as follows. In section II, different LDPC decoding algorithms are discussed, and section III explains our proposed Optimized Min-sum algorithm. Section IV discusses the simulation results, and finally section V concludes the paper.

II. LDPC DECODING ALGORITHMS

Decoding of LDPC codes can be two types: hard decision decoding and soft decision decoding.

1) Hard Decision Decoding

For each bit c_n, compute the checks for those checks that are influenced by c_n. If the number of nonzero checks exceeds some threshold (say, the majority of the checks are nonzero), then the bit is determined to be incorrect. The erroneous bit is flipped, and correction continues. This simple scheme is

capable of correcting more than one error. Suppose that c_n is in error and that other bits influencing its checks are also in error. Arrange the Tanner graph with c_n as a root considering no cycle in the graph. In Fig. 1, suppose the bits in the shaded boxes are in error. The bits that connect to the checks connected to the root node are said to be in tier 1. The bits that connect to the checks from the first tier are said to be in tier 2. Then, decode by proceeding from the "leaves" of the tree (the top of the figure). By the time decoding on c_n is reached, other erroneous bits may have been corrected. Thus, bits and checks which are not directly connected to c_n can still influence c_n.

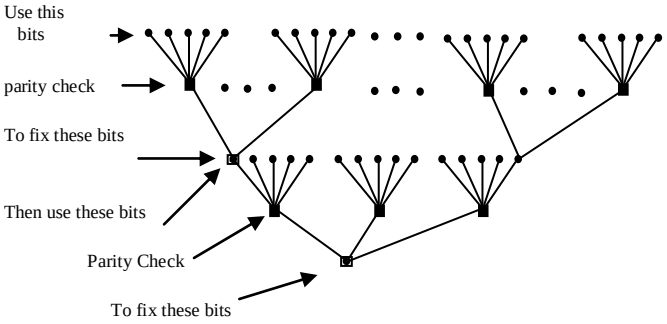

Figure 1. A parity check tree associated with the Tanner graph [18]

2) Soft Decision Decoding

In the Soft decision decoding, rather than flipping bits (a hard operation), we propagate probabilities through the Tanner graph, thereby accumulating evidence that the checks provide about the bits. The optimal (minimum probability of decoding error) decoder seeks a codeword $\hat{\boldsymbol{c}}$ which maximizes $P(\hat{\boldsymbol{c}} \mid \boldsymbol{r}, A\hat{\boldsymbol{c}} = 0)$. So, it seeks the most probable vector which satisfies the parity checks, given set of received data $r = [r_1, r_2 \ \ldots, r_N]$.

However, the decoding complexity for the true optimum decoding of an unstructured (i.e., random) code is exponential in K, requiring an exhaustive search over all 2^k codewords. Instead, the decoder attempts to find a codeword having bits c_n which maximize P ($c_n \mid \boldsymbol{r}$, all checks involving bit c_n are satisfied), it is the posterior probability for a single bit given that only the checks on that bit are satisfied. As it turns out, even this easier, more computationally localized, task cannot be exactly accomplished due to approximations the practical algorithm must make. However, the decoding algorithm has excellent performance and the complexity of the decoding is linear in the code length.

LDPC decoding is based on the parity check matrix which can also be represented using a bipartite graph. Columns in the parity check matrix represent variable nodes and rows in the matrix represent check nodes. Each variable node corresponds to one bit of the codeword and each check node corresponds to one parity check equation. Edges in the graph connect variable nodes to check nodes and represent the nonzero entries in *H* matrix. The term "low density" conveys the fact that the fraction of nonzero entries in *H* is small, in particular it is linear in the block length *n*, Parity check matrix can be of regular and irregular types. In this paper, we use the regular codes. For regular codes, the corresponding *H* matrix has d_c ones in each row and d_v ones in each column. It means that every codeword bit participates in exactly d_c parity check equations and that every such check equation involves exactly d_v codeword bits. Low density parity check codes have been constructed mostly using regular random bipartite graphs, here is an example of a regular parity check matrix with $d_c = 3$ and $d_v = 3$.

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

A graph associated with a parity check matrix *A* is called the Tanner graph and it contains two sets of nodes. The first set consists of *N* nodes which represent the *N* bits of a codeword; nodes in this set are called "bit" nodes. The second set consists of *M* nodes, called "check" nodes representing the parity constraints. The graph has an edge between the nth bit node and the *m*-th check node if and only if nth bit is involved in the mth check, that is, if $A_{mn} = 1$. Thus, the Tanner graph is a graphical depiction of the parity check matrix. The bipartite graph corresponding to this parity check matrix is shown in Fig. 2.

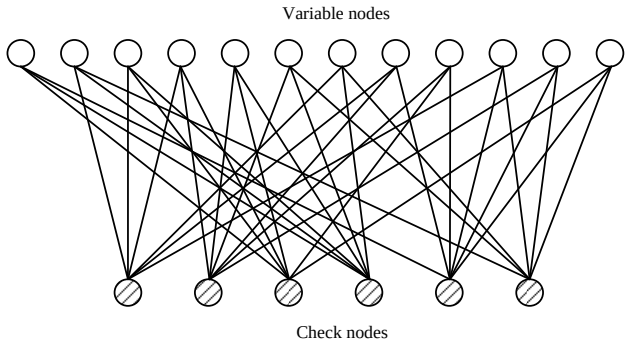

Figure 2. Bipartite graph corresponding to a regular parity check matrix

Let C be a regular LDPC code of length N and dimension K whose parity-check matrix A with $M = N - K$ rows and N columns contains exactly d_v 1's in each column (column weight) and exactly d_c 1's in each row (row weight)

. A_{mn} is the value of the m_{th} row and n_{th} column in *A*.The set of bits that participate in check is denoted: $N_m = \{n: A_{mn} = 1\}$. The set of checks that participate in bits $M_n = \{m: A_{mn} = 1\}$.

Assume codeword, $c = [c_1, c_2, c_3 \ldots\ldots, c_N]^T$. Before transmission, it is mapped to a signal constellation to obtain the vector, $t = [t_1, t_2, t_3 \ldots\ldots, t_N]^T$,

where

$$t_n = 2 * c_n - 1,$$

which is transmitted through an AWGN channel with variance

$$\sigma^2 = {}^{N_0}/_2,$$

$$r = [r_1, r_2, r_3 \ldots\ldots\ldots, r_N]^T$$

where

$$r_n = t_n + v_n.$$

Here, v_n is the Additive White Gaussian Noise (AWGN) with zero mean. Let hard decision vector,

$z = [z_1, z_2, z_3 \dots\dots\dots, z_N]^T$ Be $z_n = sgn(r_n)$

Where $\text{sgn}(r_n) = \begin{cases} 1 & r_n > 0 \\ 0 & otherwise \end{cases}$

The following notations concern bipartite graphs and message-passing algorithms running on these graphs and will be used throughout the paper.

L_n: A priori information of bit node, n

$\overline{L_n}$: A posteriori information of bit node, n

$E_{m,n}$: The check to bit message from m to

$F_{n,m}$: The bit to check message from n to m

A. Sum Product Algorithm:

The Sum Product Algorithm [17] can be summarized in the following four steps.

Step 1: Initialization

A priori information, $L_n = -r_n$

Bit to check message initialization, $F_{n,m} = L_n$

Step 2: Horizontal Step

Check node Processing:

$$E_{m,n} = \log \frac{1 + \prod_{n' \in N(m) \backslash n} \tanh\left(F_{n',m}/2\right)}{1 - \prod_{n' \in N(m) \backslash n} \tanh\left(F_{n',m}/2\right)}$$

Step 3: Vertical Step

A posteriori information:

$$\overline{L_n} = L_n + \sum_{m \in M(n)} E_{m,n}$$

Bit node Processing:

$$F_{n,m} = \overline{L_n} + \sum_{m' \in M(n) \backslash m} E_{m,n}$$

Step 4: Decoding Attempt

$\overline{L_n} > 0, \overline{c_n} = 0$, else $\overline{c_n} = 1$

If $A\overline{c_n} = 0$ then the algorithm stops and $\overline{c_n}$ is considered as a valid decoding result.

Otherwise, it goes to next iteration until the number of iteration reaches its maximum limit.

B. Log Likelihood Decoding Algorithm for Binary LDPC codes

Step 1: Initialization:

Set $\eta_{m,n}^{[0]} = 0$ for all (m, n) with $A(m, n) = 1$.

Set $\lambda_{mn} = L_c r_n$

Set the loop counter 1 = 1.

Step 2: Check node update:

For each (m, n) with $A(m, n) = 1$, Compute

$$\eta_{m,n} = 2 \tanh^{-1}\left(\prod_{j \in \mathcal{N}_{m,n}} \tanh\left(\frac{\lambda_{mj}}{2}\right)\right)$$

Step 3: Bit node update: For each (m, n) with $A(m, n) = 1$, Compute

$$\lambda_{mn} = L_c r_n + \sum_{m \in \mathcal{M}_{n,m}} \eta_{m,n}$$

Log pseudo posterior probabilities:

For $n = 1,2 \dots, N$ Compute,

$$\lambda_n = L_c r_n + \sum_{m \in \mathcal{M}_n} \eta_{m,n}$$

Step 4: Make a tentative decision: Set $\widehat{c_n} = 1$ if $\lambda_n > 0$, else set, $\widehat{c_n} = 0$

If $A\widehat{c_n} = 0$ then stop, otherwise, if the number of iteration < maximum number of iteration, loop to check node update. Otherwise, declare decoding failure and stop.

C. Min Sum Decoding

The sum–product algorithm can be modified to reduce the implementation complexity of the decoder.

This can be done by altering the Horizontal step:

$$E_{m,n} = \log \frac{1 + \prod_{n' \in N(m) \backslash n} \tanh\left(F_{n',m}/2\right)}{1 - \prod_{n' \in N(m) \backslash n} \tanh\left(F_{n',m}/2\right)} \quad (1)$$

using the relationship:

$$2 \tanh^{-1} p = \log \frac{1 + p}{1 - p}$$

Equation (1) can be rewritten as,

$$E_{m,n} = 2 \tanh^{-1} \prod_{n' \in N(m) \backslash n} \tanh\left(F_{n',m}/2\right) \quad (2)$$

Equation (2) can be further modified as,

$$E_{m,n} = 2\tanh^{-1}\prod_{n'\in N(m)\backslash n}\operatorname{sgn}\left(F_{n',m}\right)\prod_{n'\in N(m)\backslash n}\tanh\left(\left|F_{n',m}\right|/2\right)$$

$$= \prod_{n'\in N(m)\backslash n}\operatorname{sgn}\left(F_{n',m}\right)2\tanh^{-1}\prod_{n'\in N(m)\backslash n}\tanh\left(\left|F_{n',m}\right|/2\right) \quad (3)$$

The Min-sum algorithm simplifies the calculation of (3) even further by recognizing that the term corresponding to the smallest $F_{n',m}$ dominates the product term and so the product can be approximated by a minimum:

$$E_{m,n} = \prod_{n\prime\in N(m)\backslash n}\operatorname{sgn}(F_{n',m})\min_{n\prime\in N(m)\backslash n}\left|F_{n',m}\right| \quad (4)$$

D. Normalized Min Sum Decoding

Normalized Min sum algorithm [16] further modifies the min sum algorithm by multiplying a normalizing factor (say υ) where $o < \upsilon \leq 1$ in the horizontal step to achieve a better error performance closer to sum product algorithm.

$$E_{m,n} = \upsilon\prod_{n\prime\in N(m)\backslash n}\operatorname{sgn}(F_{n',m})\min_{n\prime\in N(m)\backslash n}\left|F_{n',m}\right| \quad (5)$$

A flow chart of Normalized Min Sum algorithm is given below:

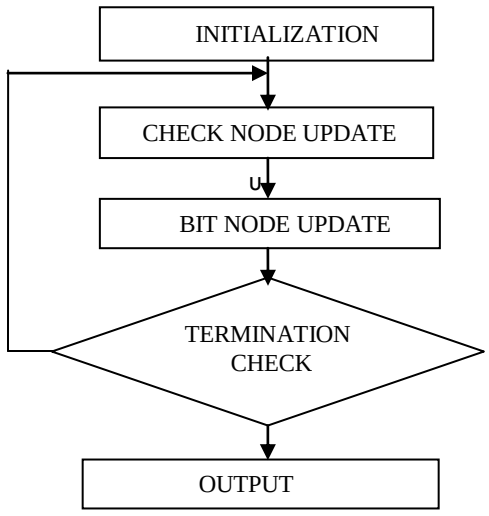

Figure 3. Flow chart of Normalized Min Sum Algorithm.

III. PROPOSED OPTIMIZED MIN-SUM ALGORITHM

1) Motivation

The foundation of our work is based on the improvement in error performance of normalized min sum algorithm [16]. From descriptions in previous sections, we have seen that the sum product decoding [7] has been reduced to different forms to reduce the complexity and through some compromise in performance. Min sum decoding [12] algorithm is one of them. Different works have been done on min sum decoding to improve its performance to get closer to sum product algorithm performance like normalized min sum decoding algorithm[16], adaptive min sum decoding algorithm[14], self-corrected min sum decoding algorithm[15] etc. In these papers, they proposed different factors which modifies and improves the error performance in different ways. In the normalized min-sum algorithm, a normalizing factor was proposed to be multiplied in check node. But, the error performance using normalized min-sum algorithm can be further modified to get closer to the error performance of sum product algorithm.

2) Optimization Factor, $\boldsymbol{\alpha}$

The value of optimization factor α varies for different Signal to Noise Ratio (SNR). For a particular SNR, we took the value of α that causes the minimum Bit Error Rate (BER).

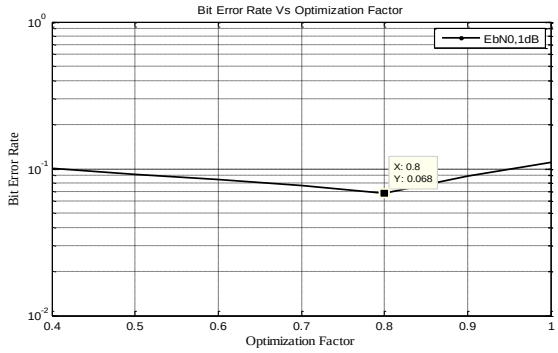

Figure 4. The impact of the optimization factor in the Optimized Min-Sum algorithm on the BER for the (2000, 1000) LDPC codes

Fig. 4 shows the variation of BER with respect to optimization factor, α for 1dB Signal to Noise Ratio. Here α= 0.8 is selected for which the BER is minimum. This same procedure is followed to calculate α for different SNRs.

3) Proposed Algorithm

In line with our motivation and the previously explained normalized min-sum algorithm, we propose the optimized min-sum algorithm. The main feature of our proposed algorithm is the use of the optimization factor. Multiplication of α both in check node and bit node update is the basic difference between optimized min-sum algorithm and Normalized Min-sum algorithm. In the Normalized Min-sum algorithm, normalizing factor was used for check node update only [16]. Also in 2 Dimensional Normalized Min Sum algorithm [19], two different factors for check and bit node updates are used and multiplied in 3 different places, check node processing, A posteriori information and bit node processing. The advantage of the proposed algorithm is that only the optimization factor is used for both bit node and check node updates. Also, the Optimization factor is not multiplied in a posteriori information which reduces complexity of the algorithm. The proposed algorithm is explained in Fig. 5 where a flow chart is shown.

First we initialize the bit to check message. Then we update the check message in the horizontal step. In this step, we multiply the Optimization factor α with the check message. After that, we proceed to the vertical step. In this step, we update the posteriori information with the help of check message and then we update the bit node. Here, we multiply the Optimization factor α with the check message. The last step is the decision making process. If the decoded codeword is correct, we stop there and take it as the output or otherwise repeat the whole decoding process until the iteration number reaches its maximum limit.

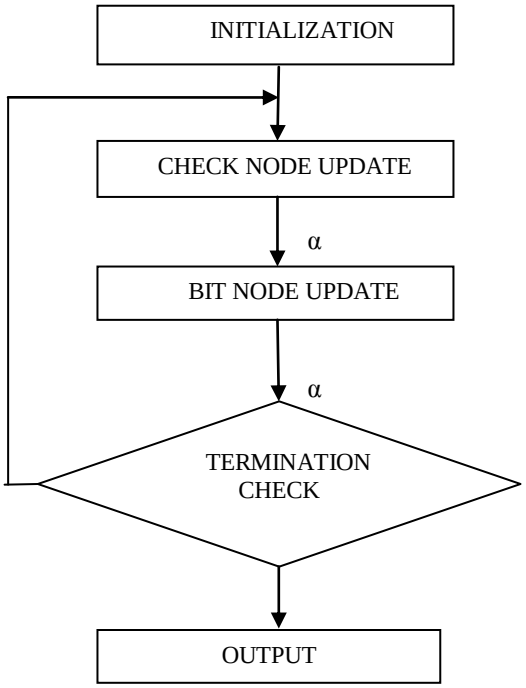

Figure 5. Flow chart of Optimized Min-Sum Algorithm

The detailed version of the algorithm is shown in the following steps. A is the optimization factor whose range is $0 < \alpha \leq 1$.

Step 1: Initialization

A priori information, $L_n = -r_n$

Bit to check message initialization, $F_{n,m} = L_n$

Step 2: Horizontal step

Check node processing:

$$E_{m,n} = \alpha \prod_{n' \in N(m)\backslash n} \operatorname{sgn}(F_{n',m}) \min_{n' \in N(m)\backslash n} |F_{n',m}| \quad (6)$$

Step 3: Vertical step

A posteriori information:

$$\overline{L_n} = L_n + \sum_{m \in M(n)} E_{m,n} \quad (7)$$

Bit node processing:

$$F_{n,m} = \overline{L_n} - \alpha E_{m,n} \quad (8)$$

Step 4: Decoding Attempt

If $\overline{L_n} > 0$, $\overline{c_n} = 0$,

else $\overline{c_n} = 1$

If $\overline{Ac_n} = 0$

Then the algorithm stops and $\overline{c_n}$ is considered as a valid decoding result. Otherwise, it goes to next iteration until the number of iteration reaches its maximum limit.

IV. Simulation Results

A. Error Performance Analysis

In total, we observed 4 simulations. The first one is regular (1944, 972) LDPC codes for IEEE 802.16e with code rate 1/2, row weight 7 and column weight 11. The codes are transmitted on AWGN channel after BPSK modulation. We set the maximum number of iteration to 50. The comparison among Sum Product (SP) algorithm [7], Min-sum (MS) algorithm [12], Normalized Min-sum (NMS) algorithm [16], 2 Dimensional Normalized Min Sum (2D NMS) algorithm and proposed Optimized Min-Sum (OMS) algorithm are shown in the Fig. 6. Simulation results show that the Optimized Min-sum algorithm obtains much better performance than Min sum algorithm, comparatively better performance than Normalized min sum algorithm,2 Dimensional Normalized Min Sum (2D NMS) algorithm and closer to that of Sum Product algorithm. Fig. 6 shows that for the BER value, 10^{-3}, our algorithm can achieve 0.05dB decoding gain over 2Dimensional Normalized Min Sum algorithm and 0.1dB gain over Normalized Min Sum algorithm.

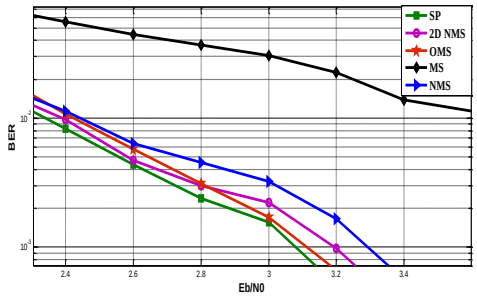

Figure 6. Bit Error Rate of LDPC codes (1944, 972) for SP, MS, NMS, and OMS

The second one is regular (1944, 1296) LDPC codes with code rate 2/3, row weight 11 and column weight 8 are used. The codes are also transmitted on AWGN channel after BPSK modulation and we set the maximum number of iteration to 50. The comparison among Sum Product (SP) [7], Min-sum (MS) [12], Normalized Min-sum (NMS) [16] and proposed Optimized Min-Sum (OMS) algorithms are shown in the Fig. 7.

Simulation results in Fig. 7 show that for the BER value, 10^{-2}, our algorithm can achieve around 0.3dB decoding gain over Normalized Min sum algorithm which depicts that Optimized Min-sum algorithm significantly better than Normalized Min sum algorithm in error performance.

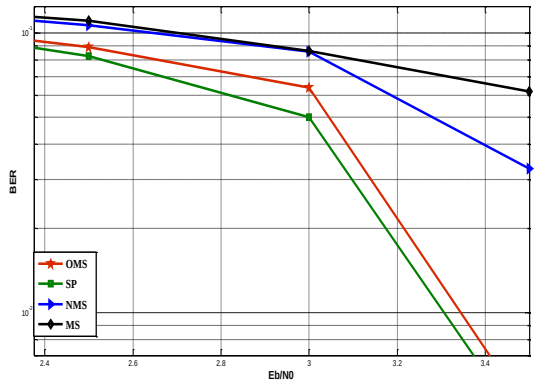

Figure 7. Bit Error Rate of LDPC codes (1944, 1296) for SP, MS, NMS, and OMS

The third one is regular (1944, 1458) LDPC codes with code rate 3/4, row weight 14 and column weight 6 are used. The codes are transmitted again on AWGN channel after BPSK modulation and we set the maximum number of iteration to 50. The comparison among Sum Product (SP) algorithm [7], Min-sum (MS) algorithm [12], Normalized Min-sum (NMS) algorithm [16] and proposed Optimized Min-Sum (OMS) algorithms are shown in the Fig. 8. Simulation results show that the Optimized Min-sum obtains much better performance than Min sum algorithm, comparatively better performance than Normalized Min-sum algorithm and closer to that of Sum Product. Fig. 8 shows that for the BER value, 10^{-2}, our algorithm can achieve 0.2dB decoding gain over Normalized Min-Sum algorithm.

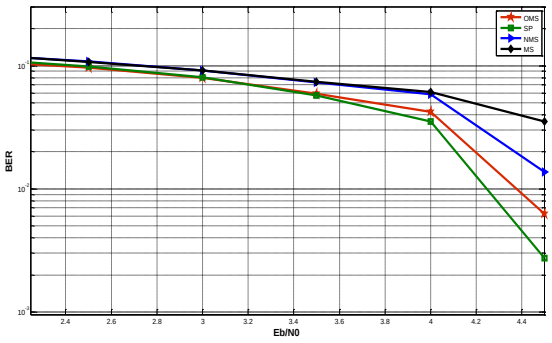

Figure 8. Bit Error Rate of LDPC codes (1944, 1458) for SP, MS, NMS and OMS

From the figures, it is clear that Optimized Min-Sum algorithm consistently shows better performance from Normalized Min-Sum algorithm and Min-Sum algorithm for different rates. We can also notice that for 1/2 rate code, Optimized Min-Sum algorithm has only -.03dB gain over Sum Product algorithm and for 2/3 and 3/4 code rates around -.05dB gain and -1dB gain respectively, So, if we take in account the reduction of complexity, it can be said that Optimized Min-Sum algorithm is almost comparable to Sum Product decoding algorithm.

The earlier simulations were run for AWGN channels. Practically, fading exists in channels. There are various types of fading channel models i.e. Rayleigh, Weibul, Log Normal etc. So to get a more practical view of Optimized Min-Sum Algorithm, error performance of the algorithm in AWGN, Rayleigh, Weibul and Log Normal channels are compared.

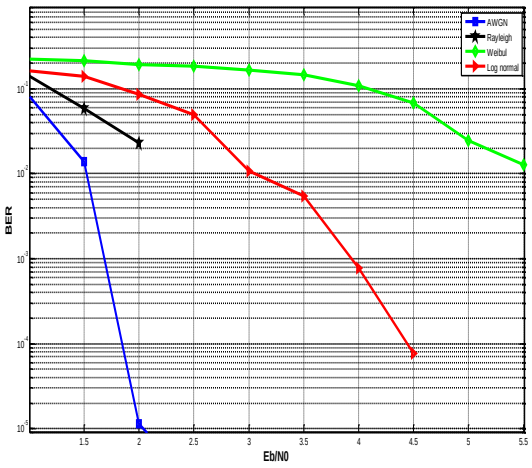

Figure 9. Comparison of OMS in AWGN, Rayleigh, Weibul and Log normal channels.

Fig 9 shows that the error performance varies with variation in fading. Error performance is best when fading is ignored in AWGN channel. In case of other channel models with fading, error performance degrades according to the degree of fading.

B. *Complexity Analysis*

TABLE I. COMPLEXITY CALCULATION

Name of the Algorithms	Calculations	
	Addition	***Multiplication***
Sum Product Decoding	150	2150
Min-Sum Algorithm	150	1100
Normalized Min-Sum Algorithm	60	1250
Optimized Min-Sum Algorithm	60	1400
2D Normalized Min-Sum	60	1650

Optimized Min-Sum algorithm is a quasi optimal decoding algorithm which improves error performance from Min-Sum decoding algorithm through slight increase in complexity. A (6, 3) regular LDPC code of code rate 1/2 was used to compare the complexity among different algorithms. Optimal Sum Product (SP) Algorithm has highest complexity and the Sub Optimal Min Sum (MS) algorithm has the lowest complexity. For quasi optimal codes, a good tradeoff between 'complexity' and 'Error Performance' is required, Normalized Min Sum (NMS) algorithm improves error performance from Min-Sum (MS) algorithm but, complexity increases as the table shows an increase in multiplication. For the proposed Optimized Min-Sum Algorithm table shows slight increase in multiplication

because of using the Optimization factor α for two updates but the tradeoff between error performance and complexity is an attractive as can be seen from the previous section. 2D Normalized Min-Sum algorithm has further increase in complexity than Optimized Min-Sum algorithm due to additional multiplication in the a posteriori information as can be seen from the table.

V. CONCLUSION

Through the introduction of Optimization factor we have obtained a better tradeoff between 'performance' and 'complexity'. We achieved much better performance than Min Sum algorithm and Normalized Min Sum algorithm in exchange of a slight increase in complexity. Using the same factor for both the nodes has reduced the complexity of calculating two different factors.

The optimization factor is determined before the decoding process which causes no additional complexity in the decoding algorithm. Thus, the proposed algorithm is a very competitive decoding technique for regular LDPC codes. Further analysis can be done for irregular LDPC codes and hardware implementation is also possible.

REFERENCES

[1] R. G. Gallager, "Low-density parity check codes," IRE Trans. on Information Theory, vol. IT-8, pp.21-28, Jan. 1962.

[2] O. J. C. Mackay, "Good error correcting codes based on very sparse matrices," IEEE Trans. on Inform. Theory, vol. 45, pp.399-43I, Mar. 1999.

[3] 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, pp. 202-209, Aug. 2005

[4] J. Lu and J.M.F. Moura, "Structured LDPC codes for high-density recording: large girth and low error floor," IEEE Trans. Magnetics, vol. 42, no. 2, pp. 208-213, Feb. 2006.

[5] Todd K. Moon, "Error correction coding- Mathematical Methods and Algorithms," WILEY- INDIA,2005 page- 653

[6] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density paritycheck codes," IEEE Trans. Inf. Theory, vol. 47, pp. 619- 637, Feb. 2001.

[7] N. Wiberg. Codes and decoding on general graphs. PhD thesis, Likoping University, 1996. Sweden.

[8] J. Chen, A. Dholakia, E. Eleftheriou, M.P.C. Fossorier, and X. Hu, "Reduced-complexity decoding of LDPC codes," IEEE Trans. Comms., vol. 53, no. 8, pp. 1288- 1299, Aug. 2005.

[9] J. Zhao, F. Zarkeshvari, and A.H. Banihashemi, "On implementation of min-sum algorithm and its modifications for decoding low-density Parity-check (LDPC) codes," IEEE Trans. Comms., vol. 53, no. 4, pp. 549-554, Apr.

[10] J. Zhang, M. Fossorier, and D. Gu, "Two-dimensional correction for min-sum decoding of irregular LDPC codes," IEEE Comm. Lett., vol. 10, pp. 180-182, Mar. 2006.

[11] J. Chen, R.M. Tanner, C. Jones, and Y. Li, "Improved min-sum decoding algorithms for irregular LDPC codes", IEEE ISIT 2005 Proc., pp. 449-453, Sep. 2005.

[12] M.P.C.Fossorier, M. Mihaljevic, and H.lmai, Reduced Complexity Iterative Decoding of Low-Density Parity Check Codes Based on Belief Propagation, IEEE Trans. on Comm. May 1999, vol. 47, no. 5, pp. 673-680.

[13] Han, Wei; Huang, Jianguo; Fangfei Wu; , "A modified Min-Sum algorithm for low-density parity-check codes," Wireless Communications, Networking and Information Security (WCNIS), 2010 IEEE International Conference on , vol., no., pp.449-451, 25-27 June 2010.

[14] Xiaofu Wu; Yue Song; Long Cui; Ming Jiang; Chunming Zhao; , "Adaptive-normalized min-sum algorithm," Future Computer and Communication (ICFCC), 2010 2nd International Conference on , vol.2, no., pp.V2-661-V2-663, 21-24 May 2010

[15] Savin, V.; , "Self-corrected Min-Sum decoding of LDPC codes," Information Theory, 2008. ISIT 2008. IEEE International Symposium on, vol., no., pp.146-150, 6-11 July 2008 doi: 10.1109/ISIT.2008.4594965

[16] J. Chen and M. P. Fossorier. Near optimum universal belief propagation based decoding of low density parity check codes. IEEE Trans. on Comm., 50(3):406–414, 2002.

[17] Sarah J.Johnson, "Iterative Error Correction: Turbo, Low-Density Parity-Check and Repeat–Accumulate Codes", Cambridge Ubiversity Press, 2010 , page : 64-69

[18] Moon, T.K. (2005) "Error Correction Coding. Mathematical Methods and Algorithms", Wiley Inter-science, ISBN 0-471-64800-0

[19] Juntan Zhang; Fossorier, M.; Daqing Gu; Jinyun Zhang; , "Improved min-sum decoding of LDPC codes using 2-dimensional normalization," Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE , vol.3, no., pp. 6 pp., 28 Nov.-2 Dec. 2005.

[20] Hrishikesh Sharma Sachin Patkar. "FPGA Design for Decoder of Projective Geometry(PG)-based Low Density Parity Check(LDPC) Codes".

AUTHORS PROFILE

Mohammad Rakibul Islam is currently working as an Associate Professor in the Department of Electrical and Electronic Engineering at Islamic University of Technology, Bangladesh. He received the B.Sc.Engg. and M.Sc.Engg. degree in Electrical and Electronic Engineering from Bangladesh University of Engineering and Technology (BUET), Bangladesh in 1998 and 2004 respectively. He also received MBA degree in Marketing from the Institute of Business Administration (IBA) under the University of Dhaka. He received his PhD degree in 2010 from Kyung Hee University, SouthKorea. His research interests include cooperative technique for wireless sensor networks, LDPC and QC-LDPC codes, secrecy capacity and other wireless applications.

Dewan Siam Shafiullah received his Bachelor of Science in Engineering from Department of Electrical and Electronic Engineering from Islamic University of Technology (IUT), Bangladesh in the year of 2011. Email: siamshafiullah@yahoo.com

Muhammad Mostafa Amir Faisal achieved his Bachelor of Science in Engineering from Department of Electrical and Electronic Engineering from Islamic University of Technology (IUT), Bangladesh in the year of 2011. Email: oranta68@yahoo.com

Imran Rahman obtained B.Sc.Engg. from Islamic University of Technology (IUT), Bangladesh in the year of 2011. Email: imran.iutoic@gmail.com