http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 11 (9): 1298-1303, 2012
ISSN 1812-5638 / DOI: 10.3923/itj.2012.1298.1303

Performance Evaluation of Sum Product and Min-sum Stopping Node Algorithm for LDPC Decoding

Nguyen Thi Dieu Linh, Gang Wang, Min Jia and Georgia Rugumira
Communication Research Centre, Harbin Institute of Technology, Harbin,
Heilongjiang 150001, People's Republic of China

Abstract: Because of the excellent error- correcting capability and optional error correct code, Low-Density Parity-Check (LDPC) codes have received great attention recently. LDPC codes can be decoded by an iterative decoding algorithm as Belief Propagation Algorithm or Message Passing Algorithm. In this paper, the extended analysis of Sum-Product algorithm (SPA) and Min-Sum algorithm (MSA) using Stopping node method is introduced. This method based on SPA using stopping node to reduce the computing complexity on LDPC code is presented, and it has also been proposed to be applied to MSA. Moreover, how to remove the effect of short cycles to improve the quality of LDPC decoder and how to make early decisions on the next iteration to reduce the number of calculations are presented in this paper. Simulations show that the stopping node algorithm not only offers a better performance but also decreases processing time by factor of 5 times under BPSK scheme and nearly 10 times under QPSK scheme by comparison with standard MSA and SPA.

Key words: Low-density parity-check code, sum-product algorithm, min-sum algorithm, stopping node, iterative decoding

INTRODUCTION

Low-Density Parity-Check (LDPC) codes were first introduced by Gallager (1962). Due to the complexity of Decoding Algorithm that requires large volume of calculations and because of the technology was not mature enough for their practical implementation, they were forgotten until the 1990s. LDPC codes were reintroduced by MacKay and Neal (1996). Then with the development of computing technology in processing and data storage, the problem of LDPC had been solved.

LDPC codes are among the known codes achieving close to the Shannon limit performance, can achieve very low bit error rates for low signal to noise ratio (SNR) application (MacKay and Neal, 1996). Comparing the Turbo codes decoding algorithm, LDPC decoding algorithm has low implementation complexity, as well as no error- floor at high SNRs. So that, LDPC codes have received great attention by their excellent error- correcting capability and optional error correct coding schemes by many standards have been adopted.

LDPC codes are a special class of linear codes with a low density check matrix which means that it has just few ones in each column and row. LDPC codes can be decoded by an iterative decoding algorithm as Belief Propagation Algorithm (MacKay, 1999) or so call Message Passing Algorithm in which the calculations are done in probability domain. To achieve the better BER performance and reduce computational complexity by decreasing the number of iterations, Sum-Product Algorithm (SPA) and Min-Sum Algorithm (MSA) were introduced and they are done in log domain. The SPA presents the highest BER performance but it is dependent on the noise variance estimation. In contrast, MSA aims for reducing computational complexity, it cause certain degradation in decoding performance but it is simpler to implement and independent on noise variance estimation.

In "Sleeping node decoder" method presented by Bresnan (2004), the reached nodes can make early decisions on the next iteration to reduce the number of calculations. Lin and Ku (2009) had presented the Elimination Node method to make node stops according to Log likelihood. In addition, modifications using Early Stopping (ES) based on Cross Entropy (CE) coefficient (Hagenauer *et al*., 1996) or based on Hard Decision Aided (HDA) (Shao *et al*., 1999) and follow Sign- Change-Ratio (SCR) for Turbo codes have also been proposed to be applied to LDPC decoder.

This study presented a comparison between the standard SPA, standard MSA and SPA using stopping node (SPA-SN), MSA using stopping node (MSA-SN) to apply in LDPC decode.

Corresponding Author: Nguyen Thi Dieu Linh, Communication Research Centre, Harbin Institute of Technology, Harbin, Heilongjiang 150001, People's Republic of China

SUM-PRODUCT ALGORITHM AND MIN-SUM ALGORITHM

LDPC codes: LDPC code is defined by a sparse parity check matrix H of size M×N that consist of M = N-K rows and N column of the parity check matrix. The code rate is represented by K/N.

LDPC code can be represented by the bilateral Tanner graph (Tanner, 1981). A graph contains two kinds of node: Bit nodes and check nodes. Bit nodes are associated with a column of H, check nodes are associated with a row of H. In the Tanner graph, a single bit node can be connected to check nodes where the elements of a column are "1" in the parity check matrix H and a single check node can be connected to bit nodes where the elements of a row are "1" in the parity check matrix H.

Sum-product algorithm: Let $L(r_{ji})$ be the messages are sent from check nodes j to bit node i; Let $L(q_{ij})$ be the messages are sent from bit node i to check nodes j. The SPA steps can be summarized as following:

***Initialization:** For the domain of Log-likelihood and for all bit nodes v_i (i=1, 2…, n) the LLR value L_i is:

$$L^{(\gamma)}(q_{ij}) = L(x_i \mid r_i) = \log\frac{\Pr(x_i = 0 \mid r_i)}{\Pr(x_i = 1 \mid r_i)} \quad (1)$$

***Phase 1:** For all check nodes c_j (j = 1, 2…, m), in the corresponding position (I, j), H_{ij} =1, in the γth iterations, calculates $L^{(\gamma)}(r_{ji})$ with the following equation:

$$L^{(\gamma)}(r_{ji}) = \left(\prod_{i' \in V_j \setminus i} \text{sign}\left(L^{(\gamma)}(q_{i'j})\right)\right) \cdot \phi\left(\sum_{i' \in V_j \setminus i} \phi\left(\left|L^{(\gamma)}(q_{i'j})\right|\right)\right) \quad (2)$$

where:

$$\phi(x) = -\log\left[\tanh\left(\frac{x}{2}\right)\right] = \log\left(\frac{e^x + 1}{e^x - 1}\right) \quad (3)$$

***Phase 2:** Calculates $L^{(\gamma)}(q_{ij})$ for all bit nodes v_i (i = 1, 2…, n), for position (i,j), H_{ij} =1, uses the Eq. 4:

$$L^{(\gamma)}(q_{ij}) = \lambda_i + \sum_{j' \in C_i \setminus j} L^{(\gamma)}(r_{j'i}) \quad (4)$$

Exploration: Computing likelihood ratio for bit nodes v_i (i = 1, 2,…, n):

$$L_i^{(\gamma)} = \lambda_i + \sum_{j \in C_i} L^{(\gamma)}(r_{ji}) \quad (5)$$

For the exploration codeword $\hat{y}_i^{(\gamma)}$ where i = 1,2,…,n in the γ^{th} iterations is:

$$\hat{y}_i^{(\gamma)} = \begin{cases} 0, & \text{sign}(L_i^{(\gamma)} = 1 \\ 1, & \text{sign}(L_i^{(\gamma)} = -1 \end{cases} \quad (6)$$

where the exploration codeword $\hat{y}_i^{(\gamma)}$ satisfies:

$$H.(\hat{y}_1^{(\gamma)}, \hat{y}_2^{(\gamma)}, ..., \hat{y}_n^{(\gamma)})^T = [0, 0, ..., 0]^T \quad (7)$$

When the number of iterations achieves the maximum value the iteration stops. Otherwise repeat from phase 1.

Min-sum algorithm: MSA is a simplified version of the SPA. There are only two differences between MSA and SPA. In MSA, the initialization of the input LLR is:

$$L^0(q_{ij}) = y_i \quad (8)$$

Where: Set γ = 1 and γ_{max} be the maximum number iteration.

And in phase 1, $L^{(\gamma)}(r_{ji})$ is calculated by Eq. 9:

$$L^{(\gamma)}(r_{ji}) = \prod_{i' \in V_j \setminus i} \text{sign}\left(L^{(\gamma-1)}(q_{i'j})\right) . \min\left|L^{(\gamma-1)}(q_{i'j})\right| \quad (9)$$

The approximation leads to both computation reductions, MSA may cause certain degradation in decoding performance. The performance of the Min-Sum decoder can be improved in a number ways. In particular, the application of a correction factor and an offset in the check node output messages (Jiang *et al*., 2006) may have beneficial effect on the decoder performance. Using stopping node to make early decisions on the next iteration is one of the methods to improve the performance of the MSA.

STOPPING NODE ALGORITHM

Firstly, set a group of bit nodes as $V^{(s)}=[v_i^{(s)}]$ then the hard decision of codeword in the γ^{th} iterations is $C^{(\gamma)}=(c_1^{(\gamma)}, c_2^{(\gamma)}, \ldots c_n^{(\gamma)})$ that correspond respectively to bit nodes $(v_1, v_2, \ldots v_n)$.

Definition: The Stopping nodes (SN) are a collection of bit nodes $V^{(s)} = [v_i^{(s)}]$ when sign of LLR (Log Likelihood Ratio) corresponding bit codes remain unchanged.

Considering regression decoding algorithms, the SN node is determined by comparing the sign of $c_1^{(\gamma)}$ under iterations. If $c_i^{(\gamma)} = c_i^{(\gamma-1)} = \ldots = c_i^{(\gamma-L)}$ where L is the number of bits observed, $v_i^{(s)}$ are nodes SN, then:

$$\sum_{j=\gamma-L}^{\gamma} |c_i^{(j)}| = L;\ c_i^{(\gamma)} = \pm 1 \quad (10)$$

When determining a stop node, the calculation process iteration at node SN, Eq. 2 (in SPA or Eq. 9 in

MSA) and Eq. (4), Eq. (5) does not need to perform calculations and only receives the value of the γ^{th} iteration by:

$$L^{(\gamma)}(r_{ji})|_{i\in V'} = L^{(\gamma)}(r_{ji})|_{i\in V'} \quad (11)$$

$$L^{(\gamma)}(q_{ij})|_{i\in V'} = L^{(\gamma)}(q_{ij})|_{i\in V'} \quad (12)$$

$$L_i^{(\gamma)}|_{i\in V'} = L_i^{(\gamma)}|_{i\in V'} \quad (13)$$

So $L(q_{ij})$ and $L(r_{ji})$ matrix are not updated at SN nodes, the amount of decoder computation can be reduced in every iteration $\gamma>\gamma'$.

When using the stopping node method, if all nodes are SN bit nodes, the Early Stopping (ES) can be used to reduce the number of iteration. So a new target of implementing ES is:

$$\sum_{j=\gamma-L}^{\lambda} |c_i^{(j)}| = L, \ \forall i, i = 1,2,...n \quad (14)$$

LDPC decoding algorithm modified SPA, MSA using stopping node by checking sign the bit code is called Sum-Product Stopping node Algorithm (SPA-SN) and Min-Sum Stopping node Algorithm (MSA-SN).

In the decoding process, with a certain code, the quality achieved depends on the number of iterations performed. The decoding processing time and the quality increase when the number of iterations increases. But this relationship is not linear. When performing building code in random or structure mode it cannot avoid the appearance of finite cycles. That means that when decoding on the graph, node inter-messages transmission depends solely on the time taken by message depending nodes, but updating information at any node depends on the node itself. Error floor appears when the quality graph of the code is in regression.

The decoding quality of LDPC iterative decoding algorithm having finite length is limited by the short cycles on bilateral tanner graph. In iterative decoding algorithms, when transmitting messages from bit node to check node and contrary, often error handling appear within short period. So when building codes, the case of short cycles must be avoided, especially in case of 4 cycles. Using Stopping Node method can improve the quality of LDPC decoder iteration by removing the effect of short cycles.

Using stop node to remove the effect of short cycles is described in Fig. 1.

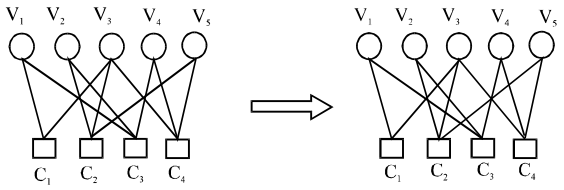

Fig. 1: Excluding the impact of short-cycle by stopping node

The shortest cycles of the code is 4 because of the set of nodes v_3, c_2, v_5 and c_4. If v_5 is the stop node at certain iteration, it do not need to perform calculation at node v_5, the shortest of code will increase to 6 by the set of nodes v_1, c_3, v_4, c_4, v_3 and c_1.

SIMULATION RESULTS

Using Monte-Carlo simulation techniques, assuming that the modulation and demodulation is ideal BPSK and QPSK, the additive noise in AWGN channel is with zero mean and variance is $\sigma^2 = N_0/2$ (N_0 is the power spectrum density bilateral). Effectiveness evaluation of the decoding LDPC algorithm using Stopping node methods can achieved with four parameters: Bit error rate, iterations average number, Stopping Node rate (the number of SN at the last iteration of the bit node) and processing time simulation at different values of bit energy rate on the noise power spectral density E_b/N_0.

MathWorks MATLAB software installed in a computer equipped with Intel(R) Core(TM) 2 Duo CPU T5670@ 1.80 GHz and 3 GB physical memory was been used to run simulations of the SPA, MSA, SPA-SN, MSA-SN algorithms.

In this study, LDPC encoding with code rate of 1/2 is set, the block size (204.102) and (504.252) have been chosen. According to experience, the value of L can be selected to L = 9. The maximum number of iterations of LDPC decoding algorithm was set to $\gamma_{max} = 30$.

Decoder using SPA-SN, MSA-SN and common SPA, MSA where, codewords error number corresponds to $E_b/N_0 = 1$ dB, 1 dB$<E_b/N_0=$ 2 dB, $E_b/N_0>2$ dB are, respectively 300, 200 and 100.

This paper had achieved the following results:

Figure 2-5 show BER performance comparisons between the SPA-SN, MSA-SN and common SPA, MSA under BPSK scheme and QPSK scheme in the (204,102) LDPC code and (504.252) LDPC code.

With BPSK scheme when $E_b/N_0>2$ dB, BER quality of the SPA-SN losses are insignificant for the (204,102) LDPC code and about 0, 1 dB for (504,252) LDPC code

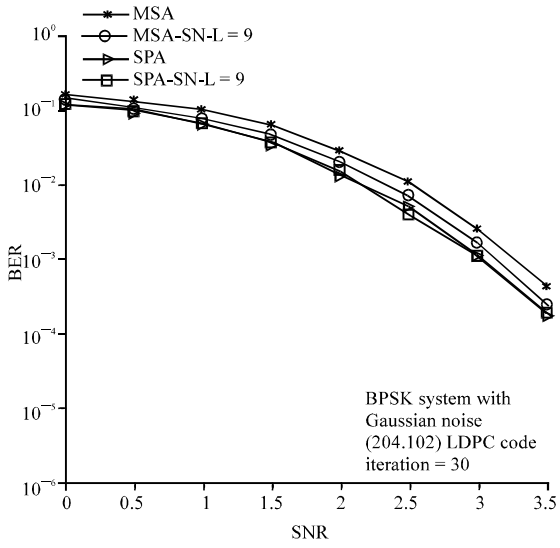

Fig. 2: BER performance under BPSK scheme comparison for (204,102) LDPC code with maximum iteration = 30

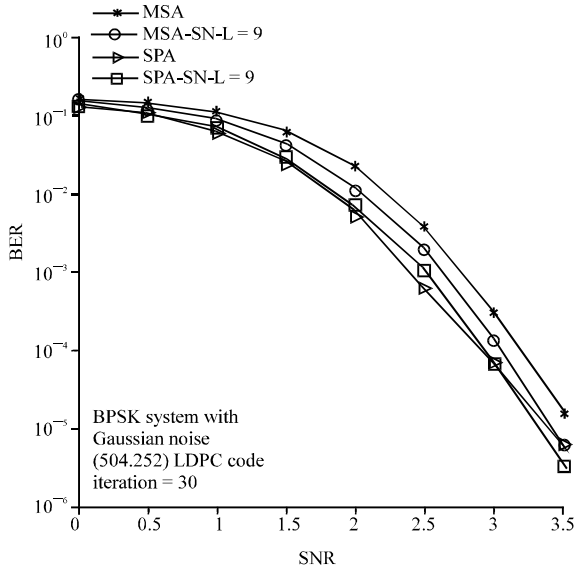

Fig. 3: BER performance under BPSK scheme comparison for (504.252) LDPC code with maximum iteration =30

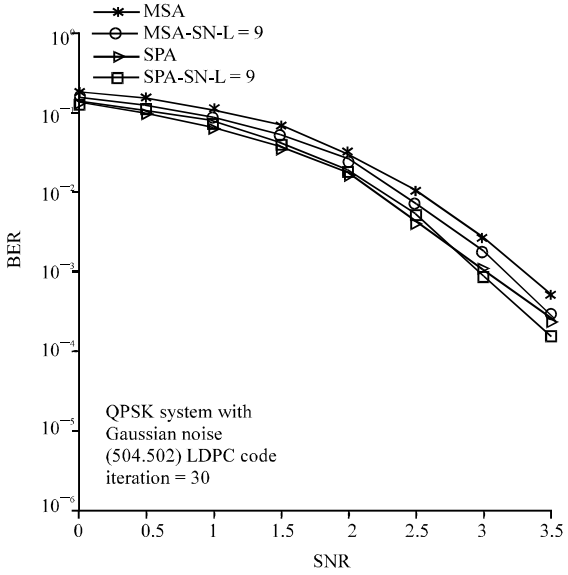

Fig. 4: BER performance under QPSK scheme comparison for (204,102) LDPC code with maximum iteration = 30

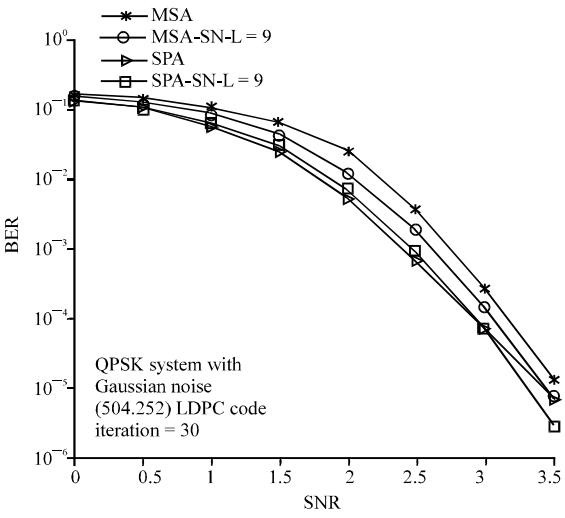

Fig. 5: BER performance under QPSK scheme comparison for (504.252) LDPC code with maximum iteration =30

compared with the SPA. But when E_b/N_0>3 dB, SPA-SN have a BER quality 60% higher than SPA for (504,252) LDPC code. In contrast, MSA-SN has 0,1 dB to 0,3 dB BER lower than common MSA in both (204,102) LDPC code and (504,252) LDPC code.

Using SPA-SN and MSA-SN under QPSK scheme can achieve the better BER quality compared-with SPA SN and MSA-SN under BPSK scheme for the (204, 102) LDPC code and (504, 252) LDPC code.

Evaluating the decoding algorithm's complexity is done by comparing processing time to assess BER on the same computer for the same codewords error number and the same values of E_b/N_0. The processing time within SPA-SN, MSA-SN compared to SPA, MSA are

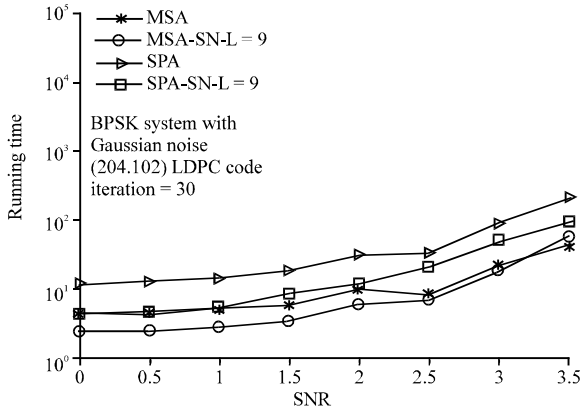

Fig. 6: Running time comparison under BPSK scheme comparison for (204, 102) LDPC code, max iteration = 30

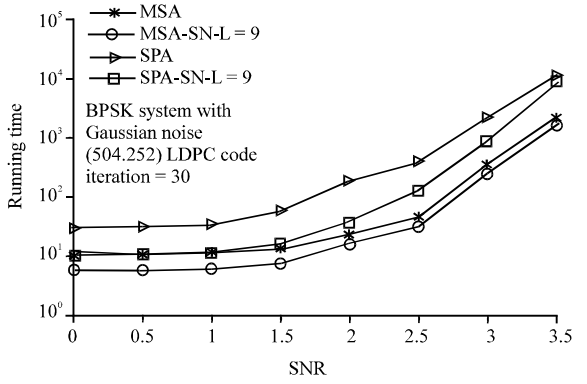

Fig. 7: Running time comparison under BPSK scheme comparison for (504.252) LDPC code, max iteration=30

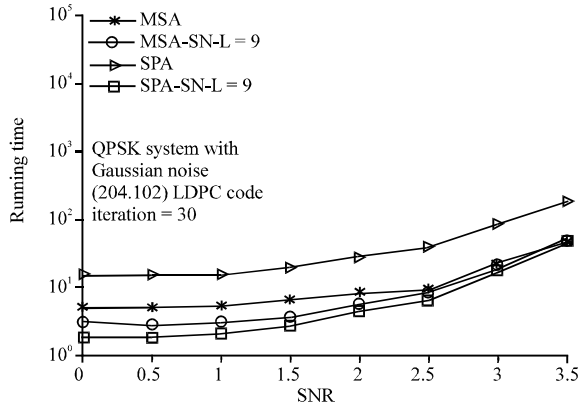

Fig. 8: Running time comparison under QPSK scheme comparison for (204,102) LDPC code, max iteration = 30

Table 1: Comparisons of processing time with maximum iteration =30

	(204,102) LDPC code		(504,252) LDPC code	
	BPSK scheme	QPSK scheme	BPSK scheme	QPSK scheme
Full processing time (Sec)	1081	884	47013	44236
SPA	550	513	23899	30207
SPA-SN	254	99	16033	3559
MSA	126	129	4116	6185
MSA-SN	118	112	2929	4258

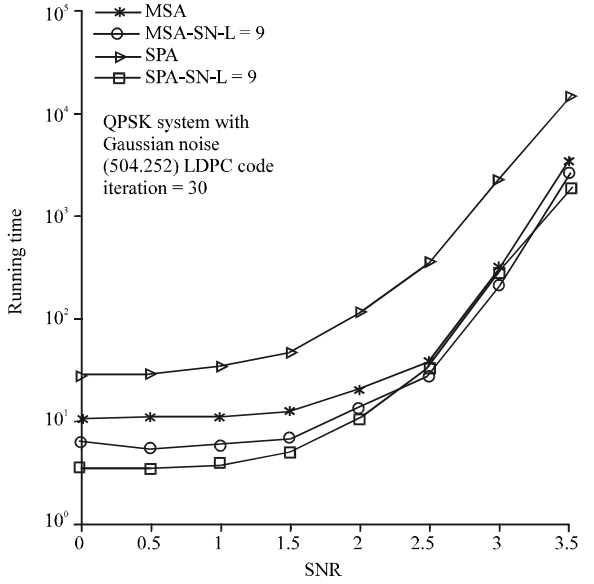

Fig. 9: Running time comparison under QPSK scheme comparison for (504.252) LDPC code, max iteration= 30

significantly decreased in all values of E_b/N_0. Figure 6 to 9 show full processing resulting time for SPA, MSA, SPA-SN and MSA-SN together under BPSK scheme and QPSK scheme. Table 1 shows the comparisons of processing time for (204,102) LDPC code and (504,252) LDPC code which is used SPA, MSA, SPA-SN and MSA-SN for decode LDPC.

Means that using the BPSK scheme, SPA-SN, MSA-SN decreases processing time by factor of 5 times compared to the common SPA, MSA algorithm and nearly 10 times under QPSK scheme.

CONCLUSION

An enhanced and more efficient algorithm for decoding LDPC codes using Stopping node method has been proposed. The performance of simulation shows that computation time of this proposed method is significantly reduced, about 5 times than the conventional

algorithms under BPSK scheme and nearly 10 times than the conventional algorithms under QPSK scheme. Moreover, the proposed method has a 0.3 dB BER lower than common MSA for MSA-SN under QPSK scheme for (504.252) LDPC code, with maintaining processing quality where losses are insignificant and acceptable for SPA-SN. Therefore, the proposed algorithm can save power, reduce transmission delay and processing cost.

REFERENCES

Bresnan, R., 2004. Novel Code Construction and decoding techniques for LDPC codes. Masters Thesis, National University of Ireland, Cork, Ireland.

Gallager, R.G., 1962. Low density parity check codes. IRE Trans. Inform. Theory, 8: 21-28.

Hagenauer, J., E. Offer and L. Papke, 1996. Iterative decoding of binary block and convolutional codes. IEEE Trans. Inform. Theory, 42: 429-445.

Jiang, M. and C. Zhao, L. Zhang and E. Xu, 2006. Adaptive Offset min-sum algorithm for low-density parity check codes. IEEE Commun. Lett., 10: 483-485.

Lin, C.Y. and M.K. Ku, 2009. Node operation reduced decoding for LDPC codes. Proceedings of the International Symposium on Circuits and Systems, May 24-27, 2009, Taipei, pp: 896-899.

MacKay, D.J.C. and R.M. Neal, 1996. Near Shannon limit performance of low density parity check codes. Electron. Lett., 33: 457-458.

MacKay, D.J.C., 1999. Good error-correcting codes based on very sparse matrices. IEEE Trans. Inform. Theory, 45: 399-431.

Shao, R.Y., S. Lin and M.P.C. Fossorier, 1999. Two simple stopping criteria for turbo decoding. IEEE Trans. Commun., 47: 1117-1120.

Tanner, R., 1981. A recursive approach to low complexity codes. IEEE Trans. Inform. Theory, 27: 533-547.