Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 1398191, 7 pages
https://doi.org/10.1155/2018/1398191

Research Article

An Alternative Approach Obtaining a Normalization Factor in Normalized Min-Sum Algorithm for Low-Density Parity-Check Code

In-Woo Yun, Hee-ran Lee, and Joon Tae Kim

Department of Electronic Engineering, Konkuk University, Seoul, Republic of Korea

Correspondence should be addressed to Joon Tae Kim; jtkim@konkuk.ac.kr

Received 11 June 2018; Revised 29 August 2018; Accepted 23 September 2018; Published 17 October 2018

Academic Editor: Mohammed El-Hajjar

The min-sum algorithm (MSA) for decoding Low-Density Parity-Check (LDPC) code is an approximation algorithm that can greatly reduce the computational complexity of the belief propagation algorithm (BPA). To reduce the error between MSA and BPA, an improved MSA such as normalized min-sum algorithm (NMSA) that uses the normalization factor when updating the check node is used in many LDPC decoders. When obtaining an optimal normalization factor, density evolution (DE) is usually used. However, not only does the DE method require a large number of calculations, it may not be optimal for obtaining a normalization factor due to the theoretical assumptions that need to be satisfied. This paper proposes a new method obtaining a normalization factor for NMSA. We first examine the relationship between the minimum value of variable node messages' magnitudes and the magnitudes of check node outputs of BPA using the check node message distribution (CMD) chart. And then, we find a normalization factor that minimizes the error between the magnitudes of check node output of NMSA and BPA. We use the least square method (LSM) to minimize the error. Simulation on ATSC 3.0 LDPC codes demonstrates that the normalization factor obtained by this proposed method shows better decoding performance than the normalization factor obtained by DE.

1. Introduction

Low-Density Parity-Check (LDPC) codes [1] proposed by Gallager in the 1960s have been used for error correcting codes for many communication standards showing the error correcting capability close to the Shannon limit. However, LDPC codes burdens the hardware since it must perform many floating-point operations repeatedly during decoding. Therefore, numerous researches have been carried out to reduce the computational complexity of LDPC decoder. One of the topics of the researches was to reduce the computational complexity of the belief propagation algorithm (BPA), known as an optimal message passing decoder for LDPC codes in the binary symmetric channel. The BPA involves great number of logarithmic and multiplicative operations when updating the check node. In an effort to reduce the computational complexity of the BPA, min-sum algorithm (MSA) has been proposed [2]. MSA approximates the exponential term of the check node update function to the minimum value selector. However, MSA has large performance degradation due to the approximation. To improve decoding performances of MSA normalized min-sum algorithm (NMSA), which normalizes the minimum value of the variable node messages, has been introduced [3].

When selecting a normalization factor in the NMSA, density evolution (DE) is usually used [4]. The DE is an analytical tool used for LDPC codes with message passing decoders. It is used to calculate the average probability density function (pdf) of messages along the edges for a given code ensemble and decoding algorithm. If the code length and the number of iterative decoding approach infinity, DE predicts the minimum signal-to-noise ratio (i.e., threshold) for convergence of the bit error probability to zero [5]. However, this method requires the assumption that the specific LDPC code is a tree structure and the algorithm used for decoding should satisfy a symmetric condition. Considering that the LDPC code is rarely a tree structure and the approximation algorithm does not satisfy a symmetric

condition, a normalization factor derived from DE may not be optimal. Additionally, performing DE needs a lot of computation when obtaining pdfs of messages [6].

For this reason, we introduce an alternative approach obtaining a normalization factor. The main idea of the proposed method is to find a factor that minimizes errors between check node outputs of BPA and NMSA by observing the distribution of check node messages of BPA with a graph called check node message distribution (CMD) chart. Since the BPA uses all the input messages of a check node when processing a check node output, it is very hard to predict the output using only a minimum input message. However, if the signal-to-noise ratio (SNR) and the number of iteration is fixed, we can easily see the trend of the error by plotting the CMD chart with the minimum value of input messages' magnitudes as the x-axis and the magnitudes of the check node output of BPA as the y-axis. After observing the CMD chart, the LSM is used to find a normalization factor that minimizes the error.

The rest of this paper is organized as follows. In Section 2.1 we briefly introduce the check node update equation for various decoding algorithms of LDPC codes in the LLR domain. In Section 2.2 we explain how to derive a normalization factor using the CMD chart and the least squares method (LSM). Lastly, we show simulation result with some of the code rates on ATSC 3.0 LDPC codes that the normalization factor derived from the proposed methods shows superior decoding performance over factor using the DE method in NMSA decoding.

2. Materials and Methods

2.1. Check Node Update of LDPC Codes. BPA, MSA, and NMSA are identical during the decoding procedures but not during the updating of check nodes [7]. Therefore, in this paper, we show only the check node update process for each algorithm. If the n^{th} variable node is V_n, and m^{th} check node is C_m, the check node message of BPA L_{mn}^{BPA} is computed using (1) and (2) where $L_{n'm}$ denotes variable node messages passing from $V_{n'}$ to C_m. The notation n' denotes variable nodes connected to C_m, excluding n, and the notation d_c denotes the number of variable nodes connected to C_m. All algorithms covered in this chapter including BPA calculate the sign of L_{mn}^{BPA} by multiplying all signs of variable node messages except the node that passes the message. The magnitudes of L_{mn}^{BPA} is calculated using $\Phi(x)$ function in (2), and this function consists of a transcendental function. This algorithm needs the $2d_c$ floating point addition and $2d_c\Phi(x)$ operation to calculate the magnitudes of d_c check node output messages per one check node.

$$L_{mn}^{BPA} = \prod_{n' \in C_m \backslash n} \operatorname{sign}\left(L_{n'm}\right) \Phi\left(\sum_{n' \in C_m \backslash n} \Phi\left(\left|L_{n'm}\right|\right)\right) \quad (1)$$

$$\Phi(x) = -\ln\left(\tanh\left(\frac{x}{2}\right)\right) \quad (2)$$

Equation (3) is used to update the check node using MSA. $\boldsymbol{L}_{n'm}$ denotes a column vector whose elements are messages from the variable nodes connected to C_m except the message from V_n. The sign of the check node output is calculated in the same way as BPA. The magnitude of check node output is processed by selecting the minimum value of variable node messages' magnitudes excluding the node that passes the processed message. In contrast to BPA, this algorithm only needs $2d_c$ floating point comparison to calculate the magnitudes of check node outputs for one check node.

$$L_{mn}^{MSA} = \prod_{n' \in C_m \backslash n} \operatorname{sign}\left(L_{n'm}\right) \min_{n' \in C_m \backslash n} \left(\left|\boldsymbol{L}_{n'm}\right|\right) \quad (3)$$

To compensate the large difference between the check node output messages of BPA and MSA, the NMSA scales the selected minimum value as in (4). By using a proper normalization factor, the NMSA can achieve performances very close to the BPA [4]. This algorithm needs one additional floating-point multiplier per one check node compare to MSA, but it can greatly improve the decoding performance.

$$L_{mn}^{NMSA} = \alpha \cdot \prod_{n' \in C_m \backslash n} \operatorname{sign}\left(L_{n'm}\right) \min_{n' \in C_m \backslash n} \left(\left|\boldsymbol{L}_{n'm}\right|\right) \quad (4)$$

2.2. The Method of Obtaining the Normalization Factor Using the CMD Chart. In the previous section, we mentioned that the MSA can significantly reduce the number of computations for obtaining the magnitudes of a check node message by approximating the transcendental function to a minimum selecting function. Also, NMSA can bridge the gap between the check node output of BPA and MSA by scaling the minimum value. In this chapter, we visualize how NMSA bridges the gap by showing an example of a CMD chart that we propose in this paper. This chart can be further used to get a normalization factor. CMD chart is short for check node message distribution chart, which plots the magnitudes of check node output messages as the y axis, and the minimum value of input variable node messages' magnitudes connected to a check node excluding n^{th} variable node ($\min(|\boldsymbol{L}_{n'm}|)$). Figure 1 shows an example of the CMD chart simulated using $(\mathrm{n}, \mathrm{j}, \mathrm{k}) = (n, d_v, d_c) = (20, 3, 4)$ regular LDPC code where n denotes number of encoded bits and d_v denotes the number of check node connected to one variable node. The parity check matrix was constructed using the Gallager method [1]. In this simulation, SNR=-5[dB] AWGN was added to the BPSK mapped signal. After demapping, 20 log-likelihood ratios (LLRs) were set as the initial value of variable node messages and check node output messages are obtained using each decoding algorithm. Data points on Figure 1 are the magnitudes of a check node output messages for the same ($\min(|\boldsymbol{L}_{n'm}|)$). To explain in detail, we pick out one check node then follow the message calculation and plotting process. One check node C_m receives four variable node messages $\boldsymbol{L}_{nm} = [1.49, 0.97, -0.40, 0.52]$. Then, the check node processes output messages using BPA using Eq. (1) and Eq. (2). The four check node output messages are $\boldsymbol{L}_{mn}^{BPA} = [-0.04, -0.06, 0.14, -0.11]$. Meanwhile, the minimum values of the variable node messages are $\min(|\boldsymbol{L}_{n'm}|) = [-0.40, -0.40, 0.52, -0.40]$ and this vector is equal to the check node output message vector of MSA ($\boldsymbol{L}_{mn}^{MSA}$). We can

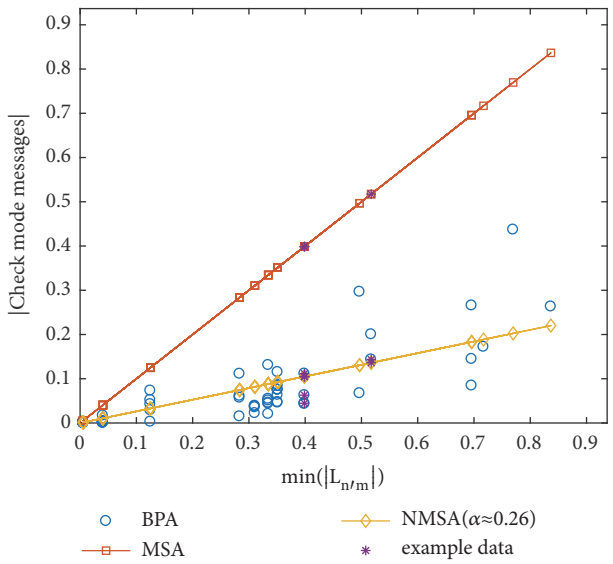

FIGURE 1: CMD chart according to $d_c = 3$, $E_s/N_0 = -5$[dB].

plot the data points $(\min(|\boldsymbol{L}_{n'm}|), |L_{mn}^{BPA}|)$ as shown in Figure 1 (∗). We can easily notice that the check node output messages, obtained using MSA, are very different from the messages obtained using BPA because data points (○, □) are distant from each other. Furthermore, we can guess that this causes a large performance gap between the two algorithms. We can also see how NMSA improves the performance in the CMD chart. In the case of setting the normalization factor as 0.26, the check node output message vector of NMSA is $\boldsymbol{L}_{mn}^{NMSA} = [-0.10, -0.10, 0.14, -0.10]$. NMSA is expected to achieve performances close to that of BPA with the error being reduced from [0.36, 0.34, 0.38, 0.29] to [0.06, 0.04, 0.01, 0.01]. In this way, we find the normalization factor which best approximates the check node output of BPA. To obtain the appropriate normalization factor, we first examine how the appropriate normalization factor varies with dc and SNR in Section 2.2.1. Then we propose a new method to minimize the error between BPA and NMSA by using LSM in Section 2.2.2.

2.2.1. Check Node Message Distribution Chart. The CMD chart shows the magnitudes of check node messages of a specific decoding algorithm against the minimum value of $|\boldsymbol{L}_{n'm}|$. In this chapter, we introduce more examples of CMD chart and explain the characteristics of check node output messages of BPA decoder using the chart. We first show how the distribution of the data points in CMD charts changes in a different SNR. Then, we show how the distribution changes when the number of variable nodes connected to one check node (d_c) changes. The distribution also changes as the iteration proceeds. However, since the processing of iteration can be considered as an increase in SNR, we skip the subject and cover it in detail in Section 2.2.2. These examples will help us to get a sense of, which values of normalization factor will improve the decoding performance. Figure 2 shows CMD chart for different SNR (E_s/N_0). The check node messages are obtained through the Monte Carlo simulation. We first generated the random bits and modulated using QPSK. We then calculated the check node messages, assuming that one check node is connected to three variable nodes ($d_c = 3$). In Figure 2(a), the simulation shows that values of check node output messages of two algorithms are very similar in a high SNR. In the lowest SNR, as shown in Figure 2(c), the differences are relatively large. This change of distribution in the different SNRs is caused by the characteristics of error between BPA and MSA. The check node update function of the BPA can be divided into two terms, which are sign computing and minimum value selecting term, and error term as in Eq. (5) [8]. Since the sign computing and the minimum value selecting term is equal to the check node update function of the MSA, the error between the check node output messages of the BPA and MSA can be calculated using the error term. This error can have values between $\ln((1 + e^{-|\infty|})/(1 + e^{-|0|})) \approx -\ln(2)$ and $\ln((1 + e^{-|0|})/(1 + e^{-|\infty|})) \approx \ln(2)$. In other words, regardless of the values of variable node messages, the error is bound to give a value between $-\ln(2)$ and $\ln(2)$. In conclusion, the initial variable node messages have large values in a high SNR environment, so $\min(|\boldsymbol{L}_{n'm}|)$ is also high, resulting in the bounded error showing a relatively small value compared to $\min(|\boldsymbol{L}_{n'm}|)$. Therefore, the data points of BPA are distributed closely to that of MSA in a high SNR. Using this information, we can understand that relatively small normalization factor will improve decoding performance in low SNR, and MSA shows great performance in high SNR.

$$L_{mn_3}^{BPA} = \operatorname{sign}\left(L_{n_1m}\right)\operatorname{sign}\left(L_{n_2m}\right)\min\left(\left|L_{n_1m}\right|, \left|L_{n_2m}\right|\right) + \ln\left(\frac{1 + e^{-|L_{n_1m}+L_{n_2m}|}}{1 + e^{-|L_{n_1m}-L_{n_2m}|}}\right) \tag{5}$$

Figure 3 shows the CMD chart according to d_c. As shown in the figure, when d_c becomes larger, the output value of the check node becomes smaller. This can be explained by the characteristics of $\Phi(x)$ function of (2) which is used to update the check node with BPA. The function $\Phi(x)$ has characteristics, which is that input and output of the function are defined only for positive real values and that they are inversely proportional. Since $\Phi(|L_{n'm}|)$ from (1) is always positive, when they are summed, the large d_c yields a small output of the check node. Therefore, for the NMSA to approximate the BPA well, in lower SNR or larger d_c conditions, the normalization factor should have a small value.

2.2.2. Obtaining Normalization Factors Using the Least Square Method. The LSM is used to derive the parameters of the curve, which minimizes the square sum (L2 norm) of the shortest distance between the curve and the given data points. It is used to remove noise from data points or to characterize the data set. We use this method to obtain a normalization factor that minimizes the error between the check node messages of BPA and NMSA. We use data points $(\min(|\boldsymbol{L}_{n'm}|), |L_{mn}^{BPA}|)$ for $n' \in C_m \setminus n$, and $m \in [1, M]$ where M is the number of the check node. If we define a

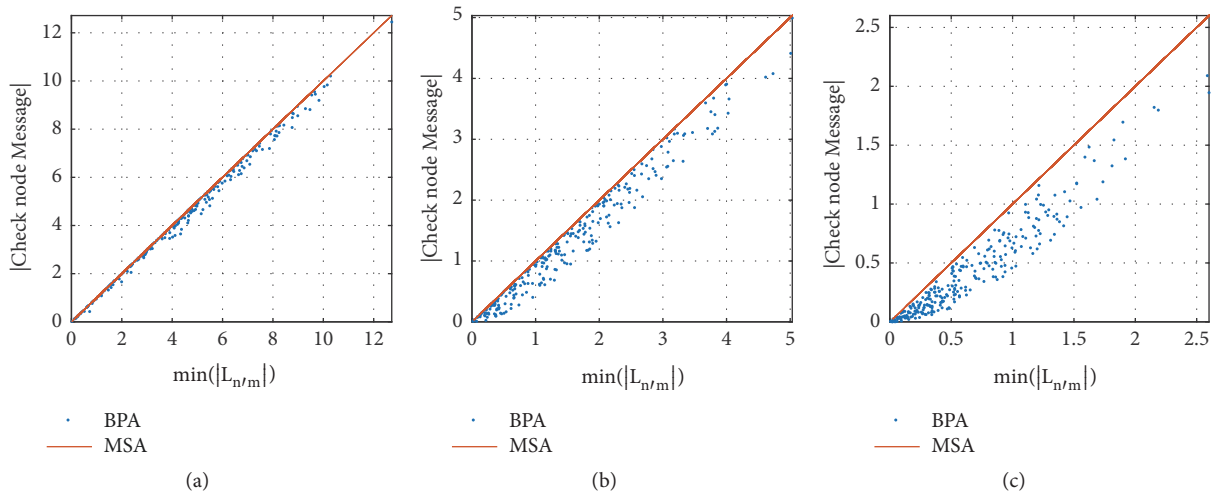

Figure 2: CMD charts according to E_s/N_0. (a) $E_s/N_0 = 5$ [dB]. (b) $E_s/N_0 = 0$ [dB]. (c) $E_s/N_0 = -5$ [dB].

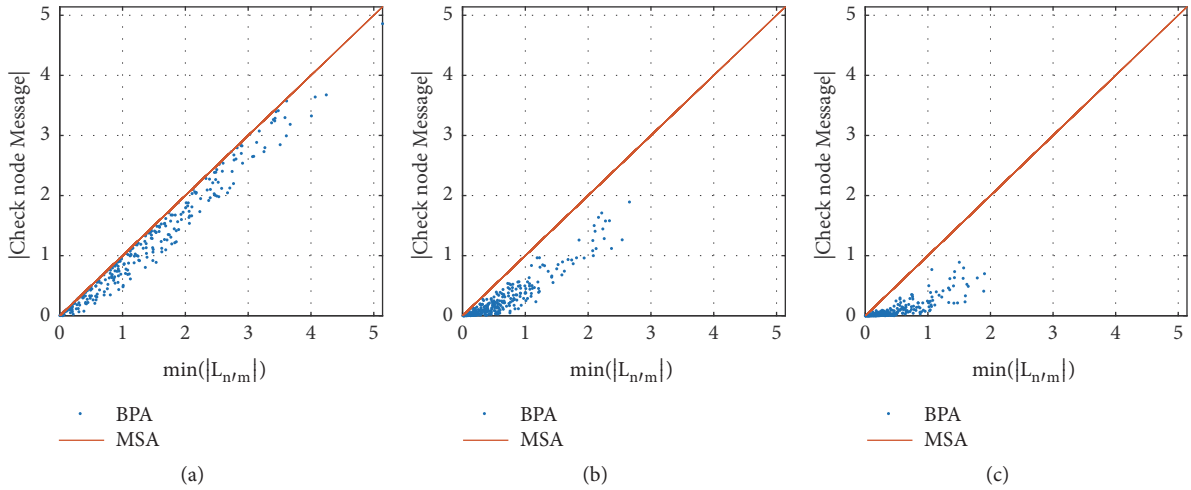

Figure 3: CMD charts according to d_c. (a) $d_c = 3$, (b) $d_c = 6$, and (c) $d_c = 9$.

set of $\min(|\boldsymbol{L}_{n'm}|)$ as the column vector $\boldsymbol{X}$ and a column vector $|\boldsymbol{L}_{mn}^{BPA}|$ as the vector $\boldsymbol{Y}$, the normalization factor can be obtained from equation (6) as follows.

$$\alpha = \left(\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{Y} \tag{6}$$

Figure 4 is a graph showing the bit error rate of four different decoding algorithms when the maximum number of iterations is set to 1. The simulation is conducted on the ATSC 3.0 LDPC code rate of 2/15 and the frame size of 64800 while using QPSK mapping. An average value for the normalization factor is derived through repetitive simulation and used for decoding in each SNR. As shown in Figure 4, the decoding performance of the proposed method is close to that of BPA than the DE method [9], especially in low SNR.

When the number of iterations is higher than one, we need to obtain an appropriate α for each iteration. So we rewrite (6) to (7) where $\alpha_{(l)}$ denotes a normalization factor for l^{th} iteration and $\boldsymbol{X}_{(l)}$ and $\boldsymbol{Y}_{(l)}$ denote vectors for data points $(\min(|\boldsymbol{L}_{n'm}|), |L_{mn}^{BPA}|)$ at l^{th} iteration.

$$\alpha_{(l)} = \left(\boldsymbol{X}_{(l)}^{\mathrm{T}}\boldsymbol{X}_{(l)}\right)^{-1}\boldsymbol{X}_{(l)}^{T}\boldsymbol{Y}_{(l)} \tag{7}$$

Figure 5 shows a computer simulation block diagram for obtaining a set of normalization factors $\alpha_{(l)}$. In the simulation, we get vectors $\boldsymbol{X}_{(l)}$ and $\boldsymbol{Y}_{(l)}$ assuming that the $(l-1)^{th}$ iteration is correctly decoded using BPA and find $\alpha_{(l)}$ that has the least square sum with the given data points. Since the decoding tends to proceed well, irrespective of the decoding algorithm in high SNR, the SNR to get $\alpha_{(l)}$ is set to a minimum error free SNR for the BPA algorithm so that the check node messages can be well approximated in a low SNR. The simulation result of obtaining the set of $\alpha_{(l)}$ for each iteration is shown in Figure 6. The maximum iteration number is set as 40 in the simulation. In early iterations, the magnitudes of LLR values of the variable node messages have small values, which means that the messages have high

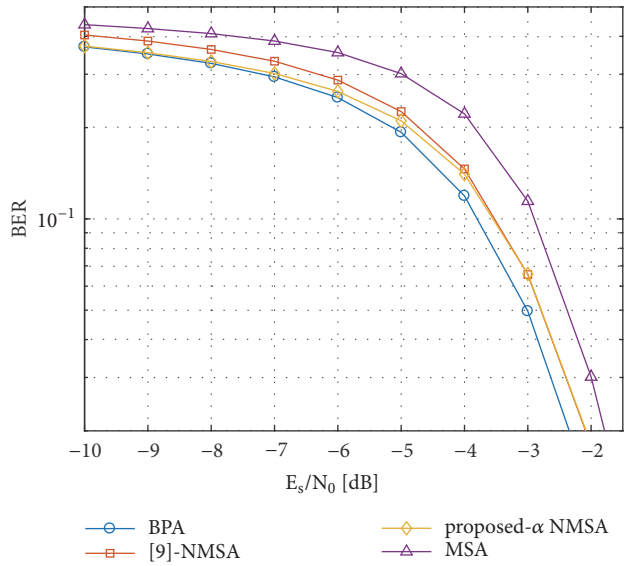

Figure 4: Bit error rate when the number of maximum iterations is set as 1.

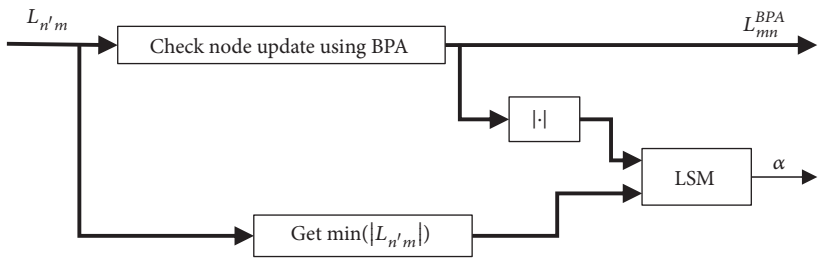

Figure 5: Block diagram of a procedure obtaining α with LSM.

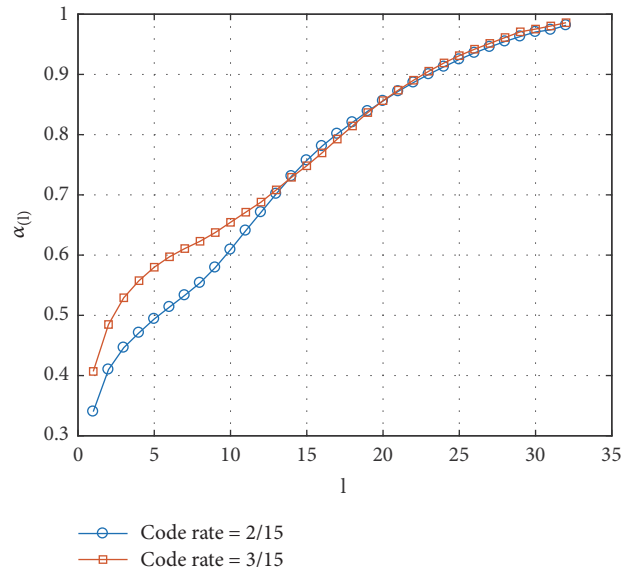

Figure 6: Trend of appropriate normalization factors over the number of iterations.

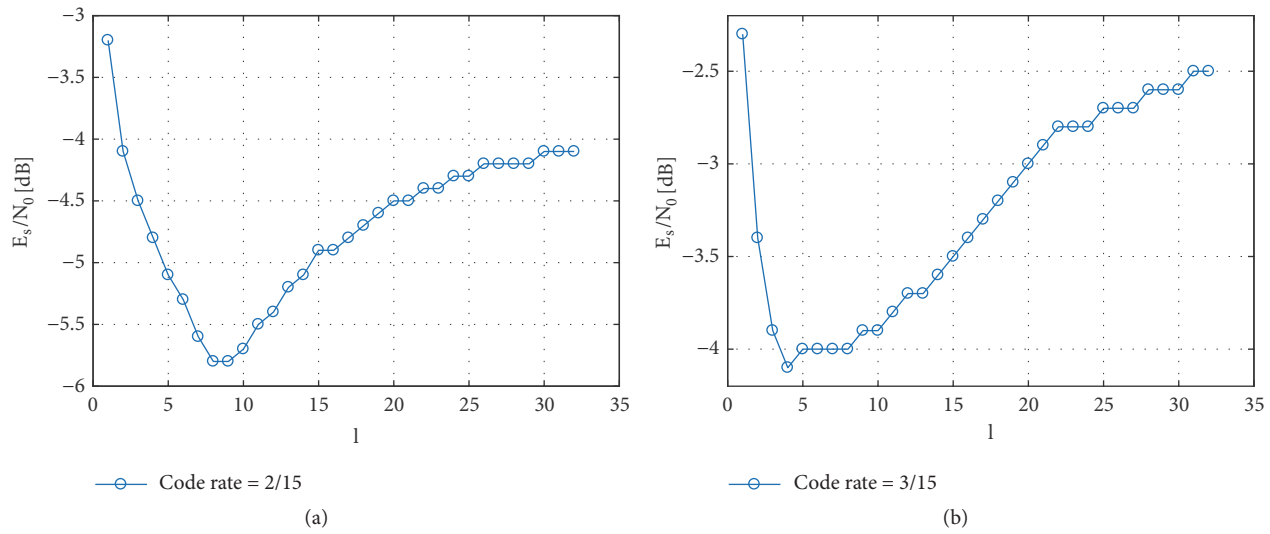

FIGURE 7: Error free E_s/N_0 when $\alpha_{(l)}$ used for decoding. (a) 2/15 and (b) 3/15.

a bit error probability. So, the CMD chart shows similar distribution as Figure 2(c), and the appropriate normalization factor is comparatively small. As the iteration progresses, the magnitudes of LLR values of the variable node messages get larger and their distribution becomes closer to the CMD chart of Figure 2(a). At the completion of iterative decoding, the normalization factor converges to 1 and the NMSA becomes equal to the MSA.

By using the set of normalization factors derived from this simulation, we can obtain good decoding performance by changing the normalization factor for each iteration. However, for the sakeof computational efficiency, we need to derive a single normalization factor for all iterations. When the $\alpha_{(l)}$ obtained from early iterations is used as a single normalization factor, the error free SNR is high because it does not properly approximate BPA as iteration proceeds. On the other hand, if the $\alpha_{(l)}$ obtained from the end of iterations is selected as a single factor, the message is exaggerated at the beginning of decoding, and as a result the error free SNR is high. Therefore, it is appropriate to use an intermediate normalization factor. So, we derived the best normalization factor for the entire iteration through simulation. The simulation result is shown in Figure 7. In the case of code rate 2/15, Figure 7(a), the normalization factor obtained from the eighth iteration $\alpha_{(8)} \approx 0.5538$ showed the lowest error free SNR. Likewise, in the case of code rate 3/15, Figure 7(b), the normalization factor obtained from the fourth iteration $\alpha_{(4)} \approx 0.5576$ showed the lowest error free SNR. As a result, we specify the normalization factor $\alpha_{(8)} \approx 0.5538$ as a single coefficient at code rate 2/15 and $\alpha_{(4)} \approx 0.5576$ as a single coefficient at code rate 3/15.

The proposed method can efficiently obtain normalization factor for each iteration by using CMD chart and LSM. This method is much easier to implement than DE method since it does not require infinitely repeated pdf calculation to get a single normalization factor [10, 11]. Also, the proposed method needs fixed number (i.e., the maximum iteration number) of α verifications to derive a single alpha for the entire iteration, whereas the DE method needs full search of alpha in range from 0 to 1 [11].

3. Results and Discussion

In this section, we show computer simulation results on ATSC 3.0 LDPC codes that compare the BPA, NMSA with the normalization factor proposed in [9] ([9]-α NMSA), and NMSA with the single normalization factor proposed in this paper.

Figure 8 shows the simulation result, when the maximum iteration is set as 40 on both ATSC 3.0 LDPC 2/15 and 3/15 codes for a code length of 64800, when layered decoding and QPSK mapping are used. As shown in Figure 8(a), the BPA showed the best performance that -6.35 [dB] of E_s/N_0 is required to have BER under 10^{-6}. The NMSA which uses the normalization factor derived in [9] ($\alpha = 0.63$) (□) has a performance gap of 0.75 [dB] compared to BPA. Whereas the NMSA, using the proposed $\alpha(\approx 0.5538)$, showed a performance gap of 0.50 [dB] (◇). The simulation on code rate 3/15 also showed that the NMSA using the proposed α ($\approx$ 0.5576)showed better performance by 0.15 [dB] compared to the NMSA that uses the normalization factor derived in [9] ($\alpha \approx 0.63$) (□) as shown in Figure 8(b).

4. Conclusions

In this paper, we introduced the CMD chart and proposed a new method to obtain a single normalization factor using the chart and LSM. In addition, we showed that the NMSA using the normalization factor derived from the proposed method showed far superior decoding performance than the NMSA employing the normalization factor of the conventional scheme [9]. Furthermore, the proposed method can be

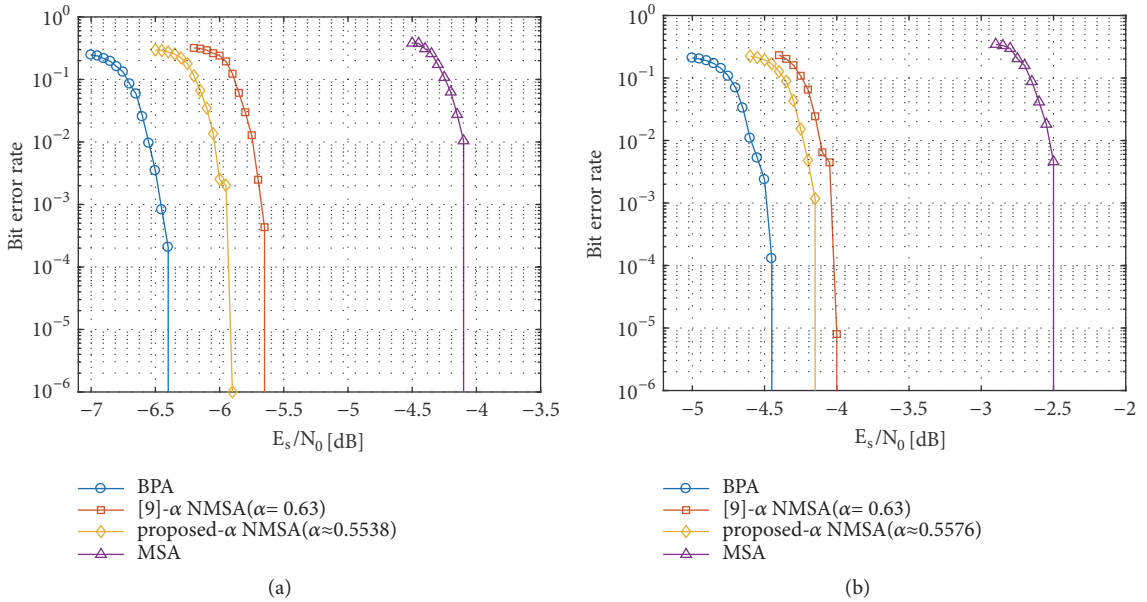

Figure 8: Bit error rate of code rate (a) 2/15 and (b) 3/15.

used to obtain the correction coefficients of other decoding algorithms that approximate the BPA algorithm.

Data Availability

No data were used to support this study.

Disclosure

In-Woo Yun and Hee-ran Lee are co-first authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the KIAT (Korea Institute for Advancement of Technology) grant funded by the Korea Government (MOTIE: Ministry of Trade Industry and Energy) (No. N0001884, HRD program for Embedded Software).

References

[1] R. G. Gallager, "Low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 8, pp. 21–28, 1962.

[2] M. P. C. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *IEEE Transactions on Communications*, vol. 47, no. 5, pp. 673–680, 1999.

[3] J. Chen and M. P. C. Fossorier, "Near optimum universal belief propagation based decoding of low-density parity check codes," *IEEE Transactions on Communications*, vol. 50, no. 3, pp. 406–414, 2002.

[4] J. Chen and M. P. C. Fossorier, "Density evolution for two improved BP-based decoding algorithms of LDPC codes," *IEEE Communications Letters*, vol. 6, no. 5, pp. 208–210, 2002.

[5] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 619–637, 2001.

[6] A. Anastasopoulos, "A comparison between the sum-product and the min-sum iterative detection algorithms based on density evolution," in *Proceedings of the IEEE Global Telecommunicatins Conference, GLOBECOM'01*, pp. 1021–1025, San Antonio, TX, USA, November 2001.

[7] R. W. Munn, "An introduction to LDPC codes," in *CRC Handbook for Coding and Signal Processing for Recording Systems*, B. Vasic, Ed., vol. 4, CRC, Boca Raton, FL, USA, 2004.

[8] F. Guilloud, E. Boutillon, and J. Danger, "λ-Min Decoding Algorithm of Regular and Irregular LDPC Codes," in *Proceedings of the 3nd International Symposium on Turbo Codes & Related Topics*, pp. 451–454, Brest, France, 2003.

[9] S. Myung, S.-I. Park, K.-J. Kim, J.-Y. Lee, S. Kwon, and J. Kim, "Offset and Normalized Min-Sum Algorithms for ATSC 3.0 LDPC Decoder," *IEEE Transactions on Broadcasting*, vol. 63, no. 4, pp. 734–739, 2017.

[10] X. Wei and A. N. Akansu, "Density evolution for low-density parity-check codes under Max-Log-MAP decoding," *IEEE Electronics Letters*, vol. 37, no. 18, pp. 1125-1126, 2001.

[11] J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Transactions on Communications*, vol. 53, no. 8, pp. 1288–1299, 2005.