Hindawi
Journal of Electrical and Computer Engineering
Volume 2022, Article ID 3076517, 8 pages
https://doi.org/10.1155/2022/3076517

Research Article

On the Performance of Rate-Compatible LDPC Codes under All-Integer Quantization Decoding

Patinya Muangkammuen, Wongpram Vongjampa, and Puripong Suthisopapan

Department of Electrical Engineering, Faculty of Engineering, Khon Kaen University, Khon Kaen, Thailand

Correspondence should be addressed to Puripong Suthisopapan; purisu@kku.ac.th

Received 3 June 2022; Revised 4 October 2022; Accepted 8 October 2022; Published 8 November 2022

Academic Editor: B. Rajanarayan Prusty

From a practical point of view, quantization is necessary for the design of any low-complexity LDPC decoder. However, the quantization scheme with the minimum computational complexity called all-integer quantization has not been well investigated since, intuitively, it seems to significantly degrade the decoding performance. It is found in this work that the utilization of 4-bit all-integer quantization for MSA-based LDPC decoder is adequate to achieve the decoding performance of the nonquantized case which can be considered as the lower bound on the decoding performance. Moreover, this simple quantization scheme can also be applied to the hardware and cost-efficient rate-compatible LDPC codes.

1. Introduction

Nowadays, it is well known that low-density parity-check (LDPC) codes can provide great error correction performance over many types of communication channels [1–3]. Therefore, LDPC codes are chosen for various communication standards, such as DVB-S2 [4], optical fiber [5], and 3GPP [6]. Their decoding algorithm, known as the sum-product algorithm (SPA), is the important factor that yields the LDPC codes with high decoding performance [7]. However, one might consider that the check node processing of SPA is a bottleneck of LDPC decoding [8]. In order to solve this problem, a simplified version of SPA, called the min-sum algorithm (MSA), has been proposed to greatly reduce the computational complexity [9].

Unlike theoretical studies that assume infinite precision decoding, the implementation of any LDPC decoder has to rely on finite precision, i.e., quantization [10]. It is known that the number of quantization bits relates to both the decoding performance and the computational complexity of decoding [10]. Specifically, LDPC decoding with a sufficient number of quantization bits can provide good error-correcting performance [11]. In contrast, a smaller number of quantization bits can incur a loss in decoding performance but a lower computational complexity can be expected [11]. Therefore, LDPC decoder based on MSA must be practically designed to trade-off between performance and computational complexity [12].

Note that, for SPA-based algorithms, three major processes involving LDPC decoding are initialization, check node processing, and variable node processing [13]. By using the same quantization bits for all decoding processes, the finite precision effects on LDPC decoding are investigated in [14]. The simulation results in this work show that the SPA is very sensitive to the effect of quantization. Furthermore, it is also demonstrated that, compared with SPA, the MSA is less sensitive to the quantization effect. The decoding performance by using the different quantization bits at variable and check node processing is provided in [15]. With the assumption of finite precision, the performance of the enhanced versions of MSA, including normalized MSA and offset MSA, is studied in [16]. For all works mentioned before, distances for all quantized level are the same, i.e., uniform quantization. It is reported in [16] that more sophisticated quantization for MSA can outperform the uniform quantization. An improved MSA specifically designed for 5G LDPC codes is proposed [17, 18], and their simulation results are carried out over finite precision effect.

Contrary to the conventional quantization scheme that relies on both integer and fractional parts, our work focuses attention on the all-integer quantization decoding, i.e., quantization without decimals, to further minimize the computational complexity of the LDPC decoder. We first demonstrate that, for fixed rate LDPC codes, the decoding performance of MSA with only 4-bit integer quantization is almost identical to that of MSA with infinite precision. After getting this result, our hypothesis is that rate-compatible LDPC codes could have higher sensitivity to the effect of integer quantization than the fixed rate LDPC codes. To the best of our knowledge, the performance of rate-compatible LDPC codes under integer quantization has not yet been reported. So, the performance of this particular LDPC coding scheme is deeply investigated in this work. Surprisingly, our results reveal that rate-compatible LDPC codes, constructed from both puncturing and shortening, can exhibit excellent decoding performance under 4-bit integer quantization.

The remainder of this paper is organized as follows. Section 2 presents min-sum decoding for rate-compatible LDPC codes. The quantization decoding is described in Section 3. We show simulation results and discussions in Section 4. Finally, Section 5 presents the conclusions.

2. Basic and Background

In order to comprehend our work, the overall design diagram for the LDPC code is described in this section. The system shown in Figure 1 can be explained as follows. Initially, the information vector **m** is generated and then sent to the rate-compatible low-density parity-check (RC LDPC) code. After that, the codeword **c** is produced. The details of RC LDPC codes will be explained below.

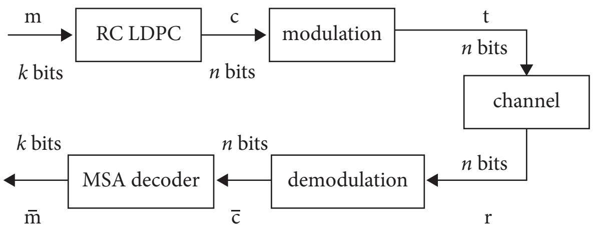

Figure 1: Block diagram of rate-compatible LDPC codes with MSA decoder.

2.1. Rate-Compatible LDPC Codes. LDPC codes are binary linear block codes defined by a sparse parity-check matrix **H** of dimension $m \times n$. This matrix can be represented in terms of the Tanner graph. The number of rows m is related to the number of check nodes in Tanner graph. Similarly, the number of columns n corresponds to the number of variable nodes. The nonzero entries in **H** matrix are represented by the edges connected between the variable node and check node.

To straightforwardly deal with the time-varying channel, many pairs of encoders and decoders must be employed to support various code rates. However, the hardware cost is very high. Rate-compatible (RC) codes are the solution to this problem since they can adapt the code rate according to the channel conditions by using only a single pair of encoder and decoder, i.e., mother code. From a fixed rate mother code, shortening techniques can be used to provide a series of lower rate codes whereas puncturing can be applied to generate a series of higher rate codes.

The idea of shortening is to insert known symbols instead of actual message symbols. Figure 2 shows the distinction between the code rates after encoding and removing. Firstly, the known symbols with length k_s bits are inserted into the actual information with length k_m bits. The information vector **m** with a length $k = k_m + k_s$ bits is sent to the encoding process. Note that the positions and values of the known symbols are known to both the transmitter and receiver. Then, the encoder forms the codeword **c** of length n bits. The code rate obtained after the encoding is $R = k/n$. After that, the known symbols are removed from the codeword and the code rate change to $R_s = k_m/n_s$, i.e., shortened code rate. So, the code rate after shortening depends on the number of known symbols, and the length of the actual codeword is shortened to $n_s = n - k_s$ bits.

By using the (4, 2) shortened code with the (6, 4) mother code as an example, the process of shortening is illustrated in Figure 3. The information vector with a length $k = 4$ bits is the input of the encoder. The known symbols of length $k_s = 2$ bits are inserted instead of the actual message bits. After the encoding process, all the known symbols are removed. Then, the actual codeword with the length $n_s = 4$ bits is produced.

The shortening process at the receiver side is described via Figure 4. The known symbols are inserted back into the predefined positions before decoding. Those known symbols, which can be considered perfect knowledge for a decoder, are fixed through the decoding process. Then, by removing known symbols, the decoder finally produces the estimation.

It is essential to note that the position of the inserted known symbols influences the decoding performance [19, 20]. Therefore, the shortening position should be carefully chosen. The method used to select the position of the known symbols is called the shortening algorithm. There are many shortening algorithms proposed in the literature. The best known shortening algorithm is the uniform shortening algorithm [21]. So, such an algorithm as per employed to per our work.

Note again that puncturing is a technique to obtain a series of higher rate codes from a single mother code. The certain parity symbols of the codeword vector will be punctured before transmission. Figure 5 shows a block diagram of the puncturing LDPC code for the encoding process. The information vector, length k bits is encoded to acquire a codeword **c**. After that, some parity symbols will be removed from the codeword. So, the code length is shorter and the code rate is increased.

To readily understand, the example of puncturing from code rate 1/2 into 3/4 is illustrated in Figure 6. The information vector of length $k = 3$ bits is encoded. Then, the encoder produces a codeword of length $n = 6$ in which the

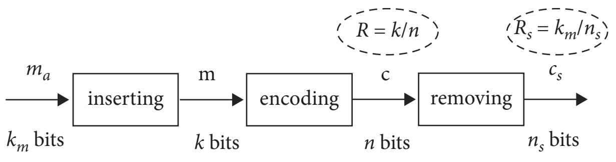

FIGURE 2: Block diagram of the shortening LDPC encoding.

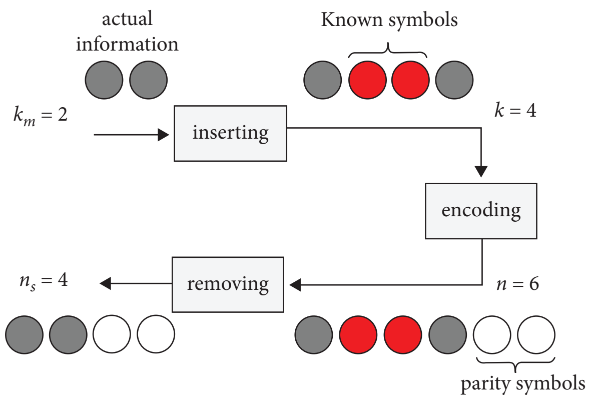

FIGURE 3: The example of the shortening process at the transmitter.

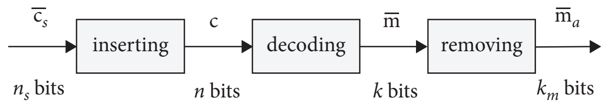

FIGURE 4: Block diagram of shortening LDPC decoding.

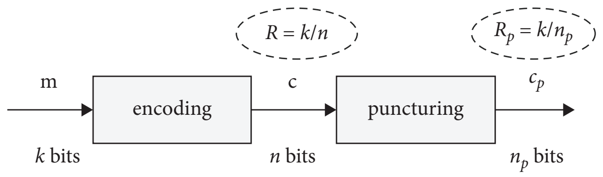

FIGURE 5: Block diagram of puncturing LDPC encoding.

length of parity symbols is 3 bits. After that, two parity symbols are punctured, so the length of the codeword is 4 bits. Finally, the punctured code rate is 3/4.

Like shortening, the selection of puncturing symbols influences the decoding performance. The best known puncturing algorithm is called grouping and sorting and is proposed in [22]. However, the simple puncturing of consecutive parity bits is used for simplicity. For punctured position, the decoder must be initiated with an ambiguous state, e.g., a zero log-likelihood ratio. Then, the punctured position can be recovered by their neighbors that have nonzero values.

2.2. Min-Sum Algorithm. Despite the excellent performance of the traditional LDPC decoder based on the sum-product algorithm (SPA), it requires high decoding complexity. The min-sum algorithm (MSA) is an alternative to SPA with

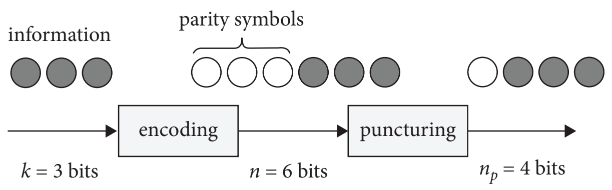

FIGURE 6: The example of the puncturing process at the transmitter.

substantially lower complexity. The MSA is the approximate algorithm of SPA that provides small decoding performance degradation. The procedures of min-sum decoding are described as follows:

At the receiver side, the log-likelihood ratios (LLR) of the received vector are expressed by

$$llr_i = \log \frac{P_r(c_i = 1|r_i)}{P_r(c_i = 0|r_i)}, \quad (1)$$

where log means $\log_e$. c_i denotes the i-th codeword, and r_i is the i-th received. For the initialization step, log-likelihood ratios value for the additive white Gaussian noise (AWGN) channel is calculated by

$$llr_i^{(0)} = \frac{2r_i}{\sigma^2}, \quad (2)$$

where σ^2 is the noise variance, and $i = 1, 2, 3, \ldots, n$. The variable-to-check message $\mathbf{M}$ is given by

$$M_{ji}^{(0)} = llr_i^{(0)}, \quad (3)$$

where $j = 1, 2, 3, \ldots, k$. The check-to-variable message $\mathbf{E}$ can be calculated by

$$E_{ji}^{(l+1)} = \prod_{i' \in N(j), i' \neq i} \operatorname{sign}\left(M_{ji'}^{(l)}\right) \cdot \min_{i' \in N(j), i' \neq i} \left|M_{ji'}^{(l)}\right|, \quad (4)$$

where $l = 1, 2, 3, \ldots, l_{\max}$ and $l_{\max}$ denotes the maximum number of decoding iterations. Let $N(j)$ be the set of all variable nodes connected with j-th check node. The updated LLR value $\mathbf{llr}^*$ is represented by

$$llr_i^{*(l)} = llr_i^{(0)} + \sum_{j \in M(i)} E_{ji}^{(l)}. \quad (5)$$

The value llr_i^* is made the hard decision to get the decoding result as follows:

$$z_i = \begin{cases} 0, & llr_i^* \leq 0, \\ 1, & llr_i^* > 0, \end{cases} \quad (6)$$

where $z^\top = [z_1, z_2, z_3, \ldots, z_n]$. Estimation at each iteration is denoted by $\mathbf{z}$. The decoding terminate when the estimation satisfies $\mathbf{Hz}^\top = 0$ or the maximum number of iterations is reached. Otherwise, the variable-to-check message is updated by

$$M_{ji}^{(l)} = llr_i^0 + \sum_{j' \in M(i), j' \neq j} E_{j'i}^{(l)}, \quad (7)$$

where $M(i)$ is the set of all check nodes connected with i-th variable node. Then, the iteration began with equation (4).

3. Quantization Decoding

It is common to use infinite precision operations in the theoretical study of LDPC decoding. However, it is difficult to execute in hardware implementation. Hence, there is a great need to use a finite precision scheme, i.e., quantization, which constitutes a low-complexity and efficient decoder. For practical LDPC decoding, the reduction of data and complexity in the LDPC decoder leads to a decrease in area, power consumption, and latency. Particularity, the quantization without a decimal part results in a significantly greater reduction of computational complexity.

The notation $(q\colon f)$ are used to represent uniform quantization scheme. Let q denote the total number of quantization bits and f be the bit length of fractional part of the value. To quantize the infinite precision into finite precision, quantization limit is defined as $[-(2^{q-1}-1)/2^f, (2^{q-1}-1)/2^f]$. The interval between every two adjacent quantized values is $1/2^f$. The set of quantized values is expressed as follows:

$$\mathbf{Q} = \left[-Q_J, -Q_{J-1}, \ldots, Q_0, \ldots, Q_{J-1}, Q_J\right], \tag{8}$$

where $Q_J = J/2^f$ and $J = 2^{q-1}-1$.

As mentioned earlier, only all-integer quantization is utilized in this work. Therefore, the notation of uniform quantization given above can be represented by $(q\colon 0)$. For example, the $(4\colon 0)$ quantization scheme will provide the set of quantized values as follows:

$$\mathbf{Q} = [-7, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7]. \tag{9}$$

For the sake of completeness, the formula to perform quantization with constraint $(q\colon f)$ is given as follows:

$$y = \begin{cases} \dfrac{\lfloor x \cdot 2^f + 0.5 \rfloor}{2^f}, -\dfrac{\left(2^{q-1}-1\right)}{2^f} \le x \le \dfrac{\left(2^{q-1}-1\right)}{2^f}, \\[2ex] -\dfrac{\left(2^{q-1}-1\right)}{2^f}, x < -\dfrac{\left(2^{q-1}-1\right)}{2^f}, \\[2ex] \dfrac{\left(2^{q-1}-1\right)}{2^f}, x > \dfrac{\left(2^{q-1}-1\right)}{2^f}, \end{cases} \tag{10}$$

where x is the input with infinite precision and $y \in \mathbf{Q}$ denotes the quantized output value. It is worth noting again that, f is always zero since all-integer quantization is only our focus.

4. Simulation Results

The bit error rate (BER) performances of LDPC codes under finite precision decoding are presented in this section. A BPSK-AWGN channel is employed for all the simulations. The sum-product and min-sum algorithms are used as the LDPC decoder. Following [15], the maximum iteration of the LDPC decoder is fixed at 20. Various block lengths and code rates involved in this section are chosen to the 5G New Radio (NR) standard [23].

4.1. Integer Quantization Effects on SPA Decoding. Theoretically, infinite precision, i.e., nonquantization is assumed for all LDPC decoding algorithms. However, the practical implementation of any LDPC decoder relies on quantization. As mentioned before, integer quantization implies the lower decoding complexity and the relationship between the number of quantization bits and decoding performance should be addressed. So, this motivates us to first investigate the traditional and powerful SPA-based LDPC decoder. Figures 7 and 8 illustrate the BER performance of integer decoding for different numbers of quantization bits. It is found that integer quantization causes severe performance degradation in all cases. Since the SPA decoding requires high resolution to calculate, even 6 bits for integer quantization are not sufficient to provide good performance. Therefore, with quite poor performance, it is not exaggerated to state that SPA decoding is very sensitive to integer quantization and integer arithmetic as well.

4.2. Integer Quantization Effects on MSA Decoding. Before showing the decoding performance of MSA under integer quantization, MSA decoding performance with a different maximum number of iterations is depicted in Figures 9 and 10. It is obviously seen that the BER performance gradually reduces when the maximum number of iterations is increased. However, it is observed that the BER performance tends to saturate at 100 decoding iterations.

Importantly, the use of the maximum number of iterations of 20 causes a small performance loss, i.e., 0.3–0.4 dB compared with employing 100 iterations. Therefore, MSA with 20 decoding iterations, which is quite a good compromise between time consuming and BER performance, is selected to exhibit the effect of integer quantization in this work.

To show the robustness of MSA decoding to the integer quantization effect, Figures 11 and 12 present the performance of MSA under this circumstance. It is obviously seen that the decoding with 4 bits integer quantization can provide the performance very close to the infinite precision decoding. Although the performance of MSA decoding is slightly poorer than that of SPA, the MSA is more powerful than SPA for low precision decoding, i.e., integer quantization with 2 or 4 bits. The MSA decoding is an approximation algorithm. In the check-to-variable messages procedure, for quantization decoding, the error probability for the MSA is lower than that of SPA. Hence, it is reasonable to think that the MSA decoder is less sensitive to finite precision.

To justify the robustness of MSA under integer quantization decoding, a simple analysis is carried out by observing the quantized values and nonquantized values for both SPA and MSA. The check-to-variable message or check node processing is focused on mean squared error (MSE) calculating because SPA and MSA are different only at this step. The check node processing for SPA is given by

$$E_{ji}^{(l+1)} = 2\tanh^{-1} \prod_{i' \in N(j), i' \neq i} \tanh\left(\frac{M_{ji'}}{2}\right). \tag{11}$$

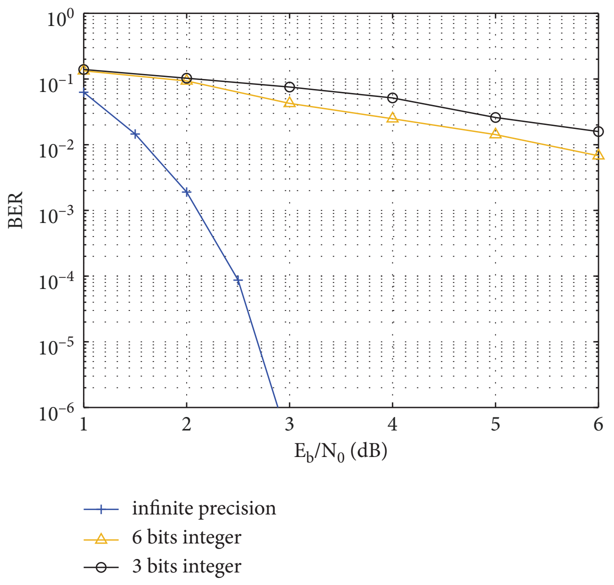

FIGURE 7: Effect of integer quantization on SPA decoding. The block length and code rate of LDPC code are $n = 1008$ bits and $R = 1/2$, respectively.

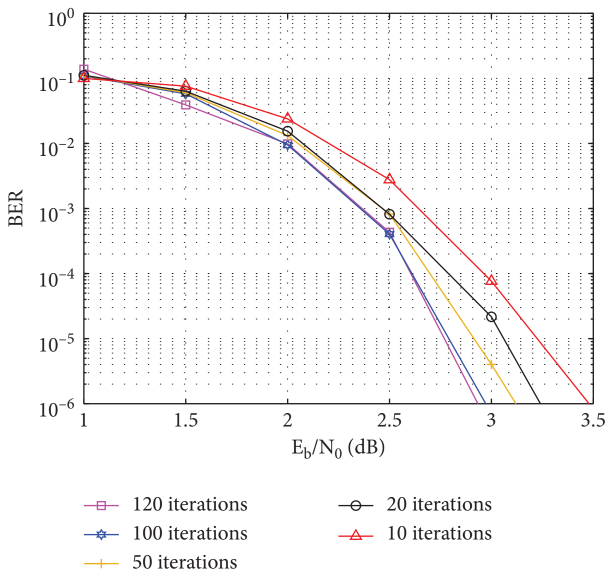

FIGURE 9: Influence of different maximum numbers of iterations on MSA decoding for LDPC codes. $n = 1008$ and $R = 1/2$.

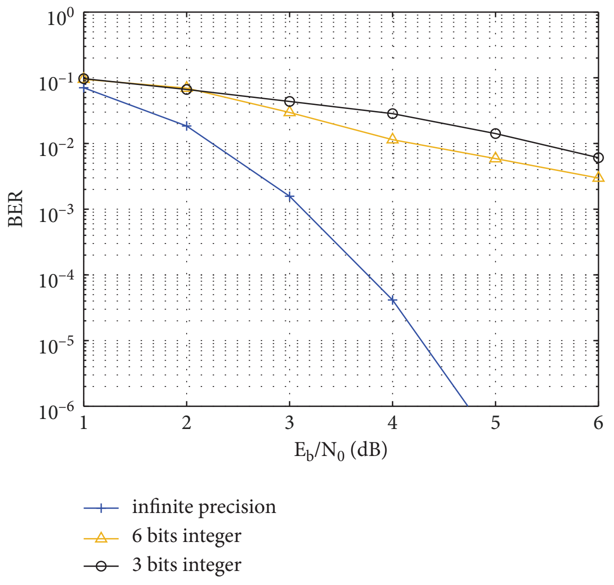

FIGURE 8: Effect of integer quantization on SPA decoding. The block length and code rate of LDPC code are $n = 264$ bits and $R = 2/3$, respectively.

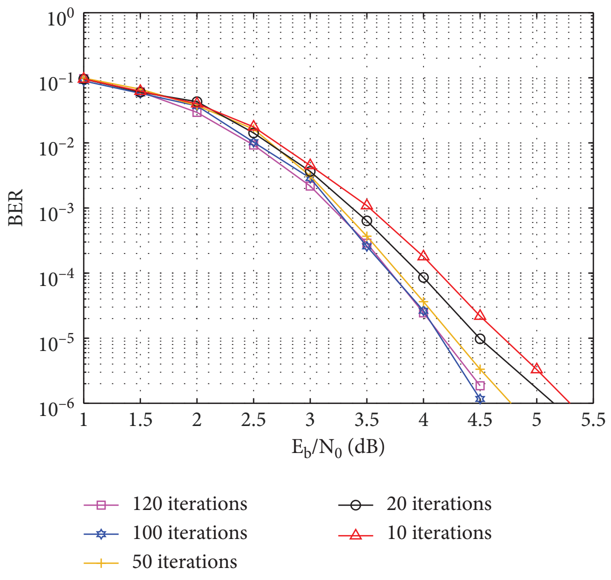

FIGURE 10: BER performance of MSA decoding for LDPC code under different maximum number of iterations. $n = 264$ and $R = 2/3$.

While this step for MSA is defined by (4).

Specifically, it is known that check node processing for MSA is an approximation of that of SPA in order to achieve lower decoding complexity. Therefore, it is worth to note that the values involved in MSA decoding are already based on approximated values or truncated values. So, it is intuitive to think that the MSA under integer decoding, i.e., further approximation of decoding value, is less sensitive to SPA with integer decoding.

To clearly show this, let y_v and x_v be the v-th quantized value and v-th nonquantized value at check node processing, respectively, where $v = 1, 2, \ldots, N$ and N is the total number of observation. The MSE between quantized value and nonquantized value can be expressed by

$$\text{MSE} = \frac{1}{N}\sum_{v=1}^{N}\left(y_v - x_v\right)^2. \qquad (12)$$

This parameter is defined to quantify the quantization error between quantized values and nonquantized values for both the MSA and SPA decoding algorithms. The MSE analysis is shown in Table 1. The E_b/N_0 used for block length

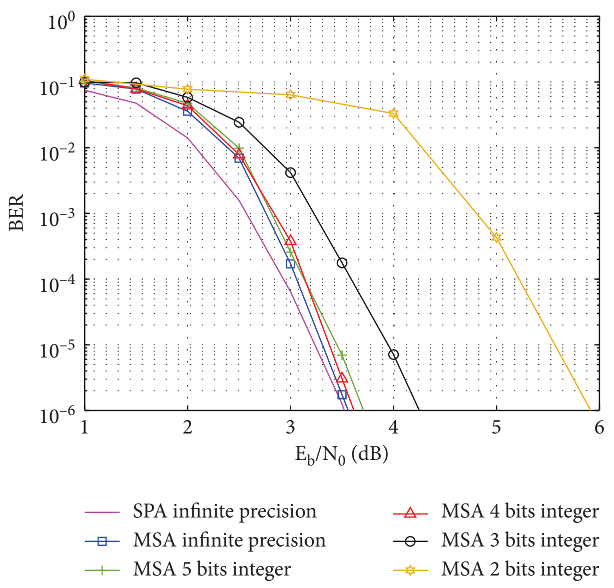

FIGURE 11: BER performance of MSA LDPC decoding with various quantization schemes. $n = 1008$ and $R = 2/3$.

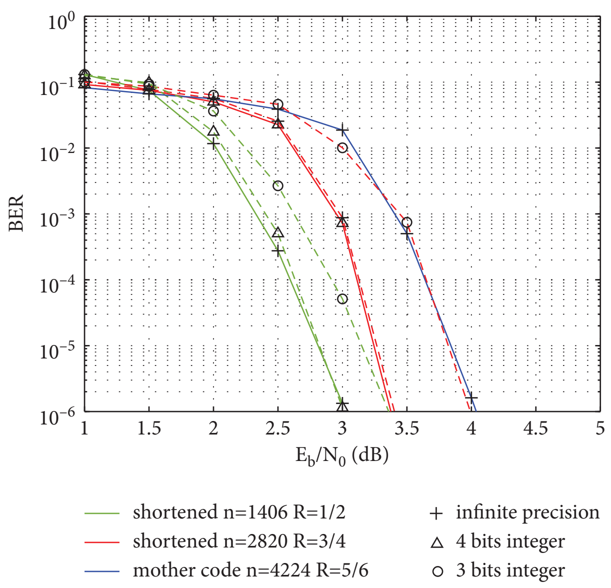

FIGURE 13: BER performance of MSA decoding for shortening LDPC codes. The solid lines represent infinite precision and the dashed lines mean integer quantization.

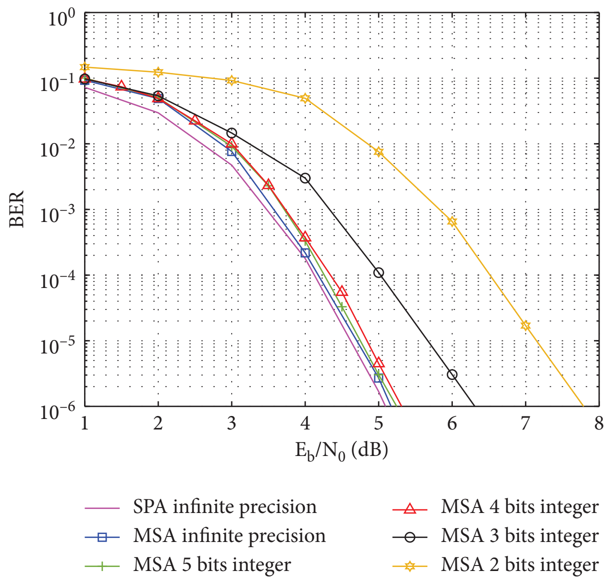

FIGURE 12: BER performance of MSA LDPC decoding with various quantization schemes. $n = 264$ and $R = 3/4$.

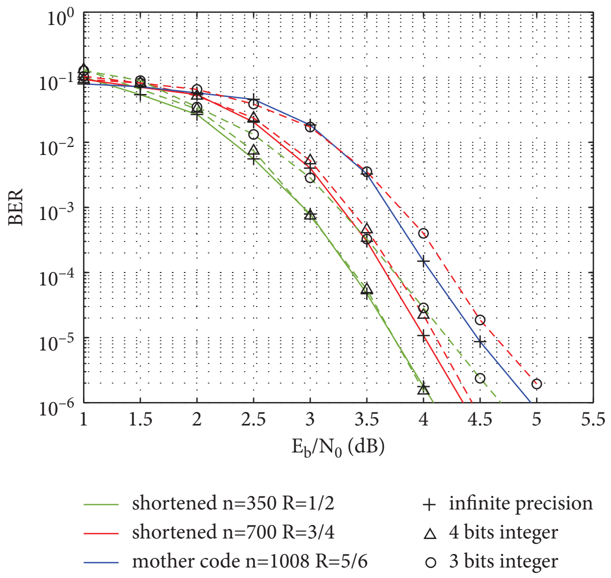

FIGURE 14: BER performance of MSA decoding for shortening LDPC codes. The solid lines represent infinite precision and the dash lines mean integer quantization.

TABLE 1: MSE between the nonquantized values and the quantized values for check nodes processing.

Algorithms	R and n	q bits 4	q bits 6
Average MSE of MSA	3/4 and 264	≈ 2.2	≈ 2
	2/3 and 1008	≈ 2.1	≈ 1.9
Average MSE of SPA	3/4 and 264	≈ 10	≈ 10
	2/3 and 1008	≈ 22	≈ 22

of 1008 bits and 264 bits is 3.5 and 5 dB, respectively, to achieve a BER of 10^{-6}. Additionally, the observed values of about 10^6 samples are used. It is clear that, in all cases, the MSE values of SPA are much greater than those incurred from MSA. So, the SPA decoding is very sensitive to the integer quantization effect, but this quantization effect seems to be negligible for MSA decoding, as shown in Figures 11 and 12.

4.3. Decoding Performance of Rate-Compatible LDPC Codes under Integer Quantization. Note again that one can construct rate-compatible LDPC codes via shortening or

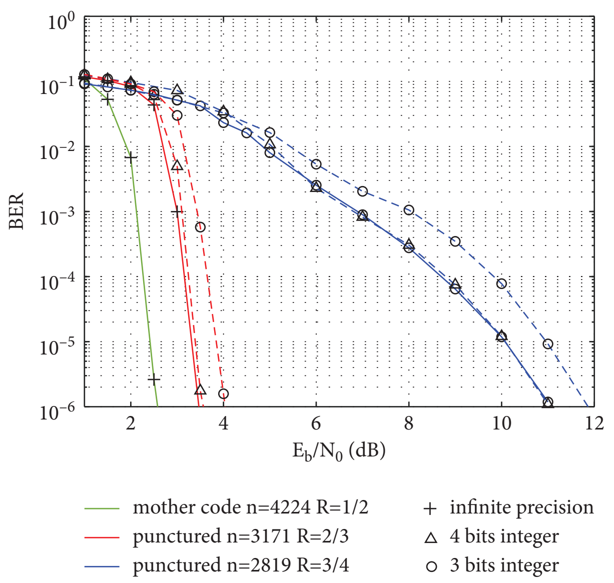

FIGURE 15: BER performance of puncturing LDPC codes. The solid lines represent infinite precision and the dashed lines mean integer quantization.

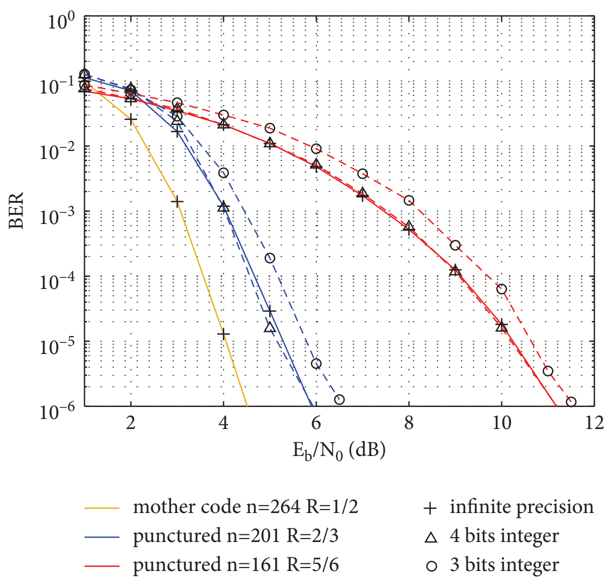

FIGURE 16: BER performance of puncturing LDPC codes. The solid lines represent infinite precision and the dashed lines mean integer quantization.

puncturing techniques From the previous section, our results reveal that the utilization of 4-bit integer quantization is sufficient for MSA decoding to achieve good decoding performance. Therefore, the rate-compatible LDPC codes with 3 and 4 bit integers are explored in this subsection. Figures 13 and 14 show the BER performance of shortening LDPC code with integer quantization. It can be seen that the decoding with 4-bit integer quantization for shortened LDPC codes still provides performance very close to that of infinite precision decoding. Regrettably, the shortened LDPC code with 3-bit integer quantization decoding leads to slight performance degradation.

Next, the performance of puncturing LDPC codes under integer quantization decoding is presented. The integer quantization effect on the puncturing LDPC is shown in Figures 15 and 16. It is observed that the simulation results are similar to those of shortening. Although the MSA decoder with puncturing is influenced by the reduction of the parity bits, it can provide good decoding performance. For 4-bit integer quantization, the BER performance of puncturing approaches the ideal case. Unlike the shortening case, the decoding performance is poorer when the quantization scheme is based on 3-bit integer quantization. So, the effect of integer quantization on the puncturing scheme is more severe when the quantization level is reduced.

5. Conclusions

The LDPC decoding with low-resolution leads to a reduction of computational complexity, power consumption, and increase of speed. The BER performance of rate-compatible LDPC codes under integer quantization decoding are presented in this paper. It is found that the rate-compatible LDPC codes with 4-bit integer quantized MSA decoding algorithms are able to attain a 0.1 dB BER performance loss compared to MSA decoding with infinite precision. As a result, under integer quantization, an efficient implementation of a high-throughput LDPC decoder can be realized. The operation of the MSA decoder with integer quantization can be implemented via look-up tables (LUT), facilitating simple hardware design. Our work can be applied to the research in 5G system. Motivated in part of the performance gap caused by using 3-bit integer quantization. We think that the performance improvement would be an interesting topic for future research.

Data Availability

The simulation data used to support the findings of this study have been deposited in the "DataSet_manuscript" repository (https://kkumailmy.sharepoint.com/:f:/p/patinyamm/En1LIoEYilFMssstMy0KjG4BLr3YFhNcptyS3d6qzPNRkQ?e=9Dxjbl).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The author is grateful to the people in the signal processing and coding laboratory. Moreover, the authors wish to thank the "GS KKU Manuscript Camp" project and Assoc. Prof. Pirat Khunkitti from the Faculty of Engineering, Khon Kaen University, Thailand, for their useful suggestions. Finally, the authors appreciate the scholarship provided by the Faculty of Engineering, Khon Kaen University, grant number is "Mas.Ee-12/2563."

References

[1] I. Ali, H. Lee, A. Hussain, and S.-H. Kim, “Protograph-based folded spatially coupled LDPC codes for burst erasure channels,” *IEEE Wireless Communications Letters*, vol. 8, no. 2, pp. 516–519, 2019.

[2] G. Ge and L. Yin, “Design and analysis of adaptive message coding on LDPC decoder with faulty storage,” *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 7658093, 13 pages, 2018.

[3] S. Suresh Kumar and M. Rajaram, “On analyzing LDPC codes over multiantenna MC-CDMA system,” *Mathematical Problems in Engineering*, vol. 2014, Article ID 652527, 6 pages, 2014.

[4] S. Muller, M. Schreger, M. Kabutz, M. Alles, F. Kienle, and N. Wehn, “A novel LDPC decoder for DVB-S2 IP,” in *Proceedings of the Design, Automation & Test in Europe Conf. & Exhibition*, pp. 1308–1313, IEEE, Nice, France, April 2009.

[5] L. A. C. Torres, J. I. A. Regalado, and A. L. O. Ortega, “LDPC-DWDM processing for Optical communications using polymer fiber,” in *Proceedings of the IEEE Colombian Conf. on Commun. and Comput*, pp. 1–5, IEEE, Cartagena, Colombia, April 2016.

[6] A. Aronov, L. Kazakevich, J. Mack, F. Schreider, and S. Newton, “5G NR LDPC decoding performance comparison between GPU & FPGA platforms,” in *Proceedings of the IEEE Long Island Systems, Applications and Technology Conference*, pp. 1–6, IEEE, New York, NY, USA, April 2019.

[7] X.-Y. Hu, E. Eleftheriou, D.-M. Arnold, and A. Dholakia, “Efficient implementations of the sum-product algorithm for decoding LDPC codes,” in *IEEE Global Telecommun. Conf.*, vol. 2, pp. 1036–1036E, IEEE, 2001.

[8] 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.

[9] I.-W. Yun, H.-r. Lee, and J. T. Kim, “An alternative approach obtaining a normalization factor in normalized Min-Sum algorithm for low-density parity-check code,” *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 1398191, 7 pages, 2018.

[10] S. W. Shaker, “DVB-S2 LDPC finite-precision decoder,” in *Proceedings of the 13th Int. Conf. on Advanced Commun. Technology*, pp. 1383–1386, IEEE, Seoul, Korea, November 2011.

[11] L. Chu, H. He, L. Pei, and R. C. Qiu, “NOLD: a neural-network optimized low-resolution decoder for LDPC codes,” *Journal of Communications and Networks*, vol. 23, no. 3, pp. 159–170, 2021.

[12] D. Oh and K. K. Parhi, “Performance of quantized min-sum decoding algorithms for irregular LDPC codes,” in *Proceedings of the IEEE Int. Symp. on Circuits and Systems*, pp. 2758–2761, IEEE, Baltimore, MA, USA, May 2007.

[13] X. Qu and L. Yin, “Non-uniform quantization scheme for the decoding of low-density parity-check codes with the sum-product algorithm,” in *Proceedings of the 6th Int. Conf. on Electronics Information and Emergency Communication*, pp. 121–125, IEEE, Beijing, China, June 2016.

[14] L. Ping and W. Leung, “Decoding low density parity check codes with finite quantization bits,” *IEEE Communications Letters*, vol. 4, no. 2, pp. 62–64, 2000.

[15] X. Zhang, Y. Tian, J. Cui, H. Yang, and Z. Lai, “Uniform all-integer quantization for irregular LDPC decoder,” in *Proceedings of the 5th Int. Conf. on Wireless Commun., Networking and Mobile Computing*, pp. 1–4, IEEE, Dubrovnik, Croatia, October 2009.

[16] X. Zhang and P. H. Siegel, “Quantized iterative message passing decoders with low error floor for LDPC codes,” *IEEE Transactions on Communications*, vol. 62, no. 1, pp. 1–14, 2014.

[17] K. Le Trung, F. Ghaffari, and D. Declercq, “An adaptation of min-sum decoder for 5G low-density parity-check codes,” in *Proceedings of the IEEE Int. Symp. on Circuits and Systems*, pp. 1–5, IEEE, Sapporo, Japan, May 2019.

[18] H. Cui, F. Ghaffari, K. Le, D. Declercq, J. Lin, and Z. Wang, “Design of high-performance and area-efficient decoder for 5G LDPC codes,” *IEEE Transactions on Circuits and Systems I*, vol. 68, no. 2, pp. 879–891, 2021.

[19] X. Liu, X. Wu, and C. Zhao, “Shortening for irregular QC-LDPC codes,” *IEEE Communications Letters*, vol. 13, no. 8, pp. 612–614, 2009.

[20] Y. Xu, B. Liu, L. Gong, B. Rong, and L. Gui, “Improved shortening algorithm for irregular QC-LDPC codes using known bits,” *IEEE Transactions on Consumer Electronics*, vol. 57, no. 3, pp. 1057–1063, 2011.

[21] A. Wongsriwor, V. Imtawil, and P. Suttisopapan, “Design of rate-compatible LDPC codes based on uniform shortening distribution,” *Engineering and Applied Science Research*, vol. 45, no. 2, pp. 140–146, 2018.

[22] J. Ha, J. Kim, D. Klinc, and S. W. McLaughlin, “Rate-compatible punctured low-density parity-check codes with short block lengths,” *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 728–738, 2006.

[23] H. Li, B. Bai, X. Mu, J. Zhang, and H. Xu, “Algebra-assisted construction of quasi-cyclic LDPC codes for 5G new radio,” *IEEE Access*, vol. 6, pp. 50229 229–250 244, 2018.