EURASIP Journal on Wireless Communications and Networking

RESEARCH

Open Access

Adaptive deactivation and zero-forcing scheme for low-complexity LDPC decoders

Taehyun Kim[1], Jonghyun Baik[2], Myeongwoo Lee[3] and Jun Heo[1*]

Abstract

A modified message propagation algorithm is proposed for a low-complexity decoder of low-density parity-check (LDPC) codes, which controls the information propagated from variable and check nodes.The proposed threshold-based node deactivation for variable nodes and zero-forcing scheme for check nodes remarkably reduce decoding complexity required for similar error performance. In the proposed scheme, different thresholds, which are determined from the base matrix of the LDPC codes, are applied for each type of variable node. In addition, thresholds for deactivating variable nodes are increased while the decoding process is operated for a reduction in decoding complexity without early error floor, which is a drawback of the conventional threshold-based deactivation scheme. Simulation results show that the proposed scheme enables normalized min-sum decoders to decode successfully with less complexity than the conventional threshold-based deactivating scheme.

Keywords: Low-density parity-check codes, Protograph-based extrinsic transfer chart, Normalized min-sum algorithm, Complexity reduction

1 Introduction

Low-density parity-check (LDPC) codes were first introduced by Gallager [1] in 1962 and rediscovered by Mackay [2] in 1999. LDPC codes have been in competition with turbo codes, which were proposed in 1993 [3], for error control in many applications. In the viewpoint of error performance, LDPC codes exhibit better behavior for high code rates compared to turbo codes, which the last are better for lower code rates. Complexity as well as error performance is one of the factors that determine the error correction code to use. Therefore, reducing the complexity of the decoding algorithm makes LDPC codes more applicable, especially to the energy limited applications, such as Internet of Things (IoT) and deep space communication. Since this paper is the research for LDPC codes, only schemes for LDPC codes are briefly introduced in this section, but there were various researches for practical usage of turbo codes [4–7]. The complexity of the LDPC decoding process is determined by three major factors: the computational complexity, the number of iterations, and the number of activated nodes.

Although the original iterative decoding algorithm for LDPC codes, known as the sum-product algorithm [2], shows good error correction performance, its computational complexity is quite high. The sum-product algorithm can be simplified using a mathematical approximation, which is called the min-sum algorithm [8]. The degraded performance from the approximation can be improved by simply scaling messages [9, 10]. On the other hand, Savin proposed a self-correction (SC) method for improving performance of min-sum algorithm [11, 12]. He checked the sign changes of the messages from variable nodes between two consecutive iterations to identify unreliable messages. Erasing unreliable messages yields better performance compared to other modified min-sum algorithms [9], especially in the error floor region [13]. As an additional effect, the SC method affects power efficiency due to a reduction in switching activity [14, 15]. The cost of the performance improvement is the additional memory required to store the signs of passing messages. This overhead can be relieved by changing the rules of the conventional SC method [16]. This scheme has memory requirements similar to those of the conventional min-sum algorithm.

The decoding complexity is proportional to the number of iterations. In order to reduce the number of

*Correspondence: junheo@korea.ac.kr
[1]The School of Electrical Engineering, Korea University, 145, Anam-ro, Seongbuk-gu, 02841 Seoul, Republic of Korea
Full list of author information is available at the end of the article

iterations, several serial scheduling methods have been proposed. Unlike classical decoding, iteratively generated information is updated following a certain sequence in a serial scheduling. Check node-wise and variable node-wise schedules have been known as layered decoding [17] and shuffled decoding [18], respectively. The required number of iterations can be reduced to half of that of parallel decoding while the performance is maintained. A stopping criterion can also reduce the average number of iterations. Recently, a stopping criterion for reducing not only the average number of iteration but also the number of calculations per iteration was proposed [19].

In addition, decoding complexity has close relationship with the number of activated nodes. An activated node is a node that calculates messages for increasing reliability. In contrast, a node that does not calculate any information, is called an deactivated node. The forced convergence (FC) method [20] reduces the number of activated nodes per iteration. During the decoding procedure, the reliability of some variable node messages become sufficiently high within a few iterations. The update of those reliable variable nodes can be skipped in the rest of the iterations [20]. Similarly, the check nodes can be deactivated when they satisfy some inequality conditions [21]. An important issue for achieving good error correction performance is proper selection of the threshold value. Sarajlc et al. tried to obtain optimized threshold values through an optimization problem [22, 23]. In addition, there exist simpler versions of the FC method [24–26] for operational efficiency.

In this paper, we introduce a low-complexity decoding scheme that reduces the number of activated nodes. Compared to FC scheme, the proposed scheme has lower complexity with less performance loss. The main contribution of this work can be presented as follows:

- Thresholds for the proposed scheme are determined by using information theoretic analysis based on a base matrix of LDPC codes.
- For the variable nodes, a threshold-based deactivation process is proposed with dynamically increasing thresholds.
- For the check nodes, the total amount of check-node operations is reduced by generating zero-forced log-likelihood ratios.

Different thresholds are applied to each variable node according to its connectivity. In addition, prematurely deactivated variable nodes are reactivated by dynamically increasing threshold values. The threshold increasing condition is checked with a simple estimator based on logical circuits. For the check nodes, the SC method is modified to make it useful for reducing check node calculations.

The rest of this paper is organized as follows: In Section 2, we explain precisely the classical decoding algorithm and conventional techniques that motivated our proposed scheme. Section 3 describes the proposed scheme. The simulation results of the proposed scheme are discussed in Section 4 and a conclusion is provided in Section 5.

2 Background

2.1 Decoding algorithms for LDPC codes

There are three major decoding algorithms for LDPC codes: maximum likelihood (ML) decoder, sum-product algorithm (SPA), and min-sum algorithm (MSA). The ML decoder is an alternative maximum a posteriori (MAP) decoder. It essentially selects the codeword that maximizes the likelihood of the received signal. Even though the ML decoder is optimum, it is impractical due to its high computational complexity. The SPA is a decoding method exchanging variable-to-check (V2C) and check-to-variable (C2V) information iteratively. The C2V message, $U_{mn}^{(itr)}$, which propagates from check node m to variable node n at the itr-th iteration, is generated as follows:

$$U_{mn}^{(itr)} = 2\tanh^{-1}\left(\prod_{n'\in N(m)\setminus n} \tanh\left(\frac{1}{2}V_{mn'}^{(itr-1)}\right)\right), \quad (1)$$

where $V_{mn}^{(itr-1)}$ denotes the V2C log-likelihood ratio (LLR) from variable node n to check node m at the $(itr-1)$-th iteration. The term $n' \in N(m) \setminus n$ denotes the neighbors of check node m except the variable node n. The V2C message, $V_{mn}^{(itr)}$, is generated using

$$V_{mn}^{(itr)} = U_{n,ch} + \sum_{m'\in M(n)\setminus m} U_{m'n}^{(itr)}. \quad (2)$$

The term $U_{n,ch}$ denotes the channel LLR of variable node n, and $M(n) \setminus m$ denotes the neighbors of variable node n except the check node m. Along the decoding procedure, the a posteriori probability LLR (APP-LLR) of variable node n at the itr-th iteration, $\beta_n^{(itr)}$, is updated as follows:

$$\beta_n^{(itr)} = U_{n,ch} + \sum_{m\in M(n)} U_{mn}^{(itr)} \quad (3)$$

Although SPA is more efficient to implement than ML, the complexity is still high due to the numerous multiplications of soft values in each iteration. MSA is a low-complexity algorithm that uses min operations instead of using multiplications. The simplified version of the check node updating rule can be represented as follows:

$$U_{mn}^{(itr)} = \prod_{n'\in N(m)\setminus n} \operatorname{sgn}\left(V_{mn'}^{(itr-1)}\right) \min_{n'\in N(m)\setminus n} \left|V_{mn'}^{(itr-1)}\right| \quad (4)$$

The drawback of MSA is degraded performance compared with SPA. For performance improvement, the normalized MSA was used in this study. The performance of MSA can be improved by simply scaling the absolute value of the C2V message. Although optimum scaling factor is an real value [10], the multiplication of real value is sublated in implementation due to the high complexity. Therefore, for the sake of simplicity and practical usefulness, we used a scaling factor of 0.75, which is close to the optimized value and can be implemented by several shift registers.

2.2 Forced convergence

The FC scheme forces variable nodes, which have highly reliable information, to stop updating their information. Specifically, variable nodes whose absolute value of APP-LLR is greater than a threshold t are deactivated during the decoding process as follows:

$$\left|\beta_n^{(itr-1)}\right| \geq t \tag{5}$$

In order to reduce the complexity stem from the variable nodes, the deactivation process should be operated before the variable node operation is started. Therefore, the APP-LLR, which was calculated in the previous iteration, is used as a criterion for deactivation. Deactivated variable nodes do not update their V2C messages and propagate the last updated messages. In order not to experience performance degradation due to hastily deactivated variable nodes, the decoder reactivates the deactivated variable nodes when they are found in the unsatisfied parity-check equation. The behavior of the FC scheme is depicted in Fig. 1. Hereafter, "FC scheme" or "conventional FC scheme" both mean the scheme proposed in [20].

2.3 Self-correction for the min-sum algorithm

The SC method is proposed to recover the performance loss of MSA and achieves a performance gain by erasing unreliable information. It identifies unreliable information by comparing the predicted V2C message with the message at the previous iteration. When the sign is changed, the decoder regards that message as unreliable information. Therefore, the process of the SC decoder can be defined as

$$V_{mn}^{(itr)} = \begin{cases} 0 & V_{mn}^{(itr)} V_{mn}^{(itr-1)} < 0 \\ V_{mn}^{(itr)} & V_{mn}^{(itr)} V_{mn}^{(itr-1)} \geq 0 \end{cases} \tag{6}$$

The erased message can be represented by assigning a zero value, which means the bit states are equiprobable. Whenever the message at the previous iteration is erased, the predicted message is sent without considering its sign. The behavior of the SC scheme is depicted in Fig. 2.

3 Adaptive deactivation and zero-forcing scheme

In this section, calculation reduction techniques for variable and check nodes are proposed. We were motivated by the FC and SC schemes that are explained in Section 2. The proposed node deactivation process for variable nodes is conceptually similar to the conventional FC scheme in the viewpoint of reducing the operational complexity of variable nodes with high reliability. The conventional FC scheme uses an equal threshold value for all variable nodes. In contrast, we decided a different threshold value for each variable node. Because the increment of reliability per iteration during the decoding process is different according to the connectivity, the use of various threshold is more efficient for complexity reduction. In addition, not fixed but variable thresholds are used for preventing early error floor, which is a drawback of the FC scheme.

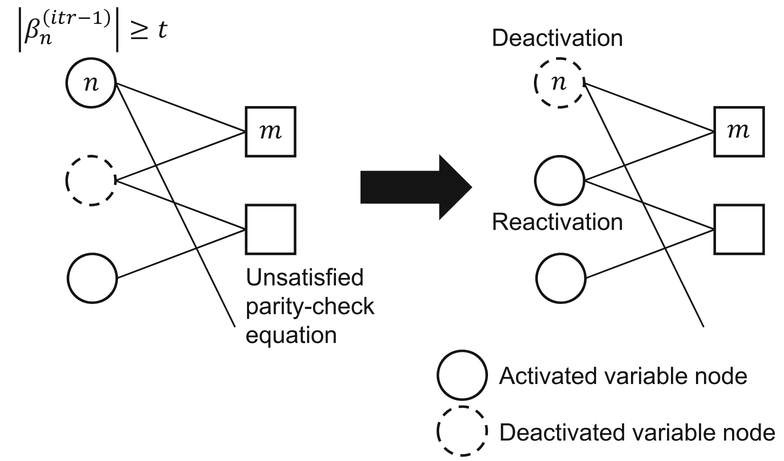
$\left|\beta_n^{(itr-1)}\right| \geq t$
n
m
Unsatisfied parity-check equation
Deactivation
n
m
Reactivation
Activated variable node
Deactivated variable node

Fig. 1 Behavior of the FC algorithm, including deactivation and reactivation; variable nodes whose absolute value of APP-LLR is greater than a threshold t are deactivated. The deactivated variable nodes are reactivated when they are found in the unsatisfied parity-check equation

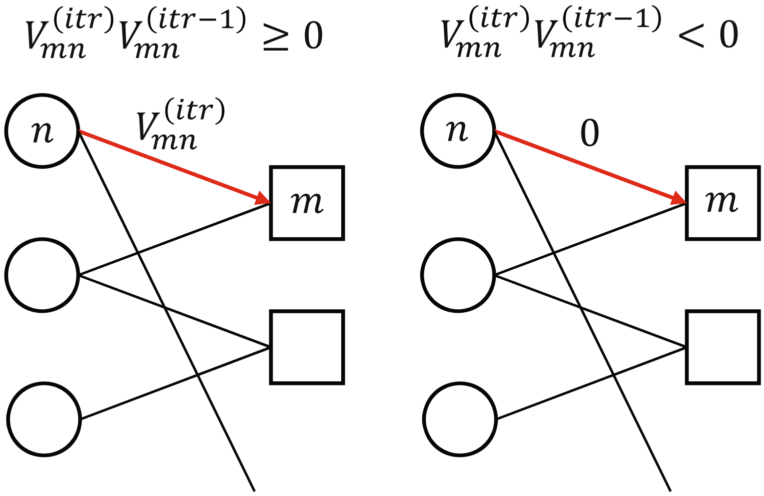
$V_{mn}^{(itr)} V_{mn}^{(itr-1)} \geq 0$
$V_{mn}^{(itr)} V_{mn}^{(itr-1)} < 0$
n
$V_{mn}^{(itr)}$
m
n
0
m

Fig. 2 Behavior of the SC scheme for the min-sum algorithm; variable node to check node (V2C) message that changes its sign between two consecutive iterations is set to zero

From the algorithm of the SC method, check node operation can be simplified according to the number of erased V2C messages. However, the main proposal of the SC method is an error correcting performance improvement. The number of erased V2C messages in the conventional SC method is not enough to obtain sufficient benefits in the viewpoint of complexity. Therefore, we proposed a threshold-based zero-forcing scheme for improving check node complexity reduction.

All thresholds used for the proposed scheme should be properly determined to deactivate nodes as much as possible but not to cause early error floor. We proposed a threshold decision method using an information theoretic analysis tool.

3.1 Adaptive node deactivation for reliable information

Assume that there is an LDPC code that is defined by an $M \times N$ parity check matrix H or an $M_p \times N_p$ base matrix B. Each element of base matrix $b_{(i,j)}$, $(1 \leq i \leq M_p, 1 \leq j \leq N_p)$ represents the number of connected edges between type-i check nodes and type-j variable nodes. As in condition (5), variable nodes satisfying the condition given in (7) are deactivated.

$$\left|\beta_n^{(itr-1)}\right| \geq t_v(j), n \in \text{set of type-}j \text{ variable nodes} \qquad (7)$$

The parameter $t_v(j)$ is a threshold for type-j variable nodes. As mentioned previously, the thresholds for each type of variable node are increased during decoding process. The reason for a usage variable threshold instead of fixed threshold is a trade-off between error performance and complexity reduction. Although substantial complexity reduction can be obtained when the threshold is small, early error floor occurs. The reason is that the hastily deactivated variable nodes arising from the small threshold value degrade error performance. On the other hand, a large threshold value brings the opposite results.

In order to reactivate the hastily deactivated variable nodes, the threshold in the proposed scheme is increased when the rate of variable node deactivation exceeds a certain criterion. The increment of threshold value reactivates the deactivated variable nodes that do not satisfy condition (7). As a result, threshold values from the early phase to the last phase compensate each other by obtaining substantial complexity reduction with relatively less performance loss compared to when the fixed threshold value is used.

Initial thresholds $t_{v,ini}$ are determined using a protograph-based extrinsic information transfer (PEXIT) chart [27], which is an analysis tool for protograph-based and multi-edge type LDPC codes. In contrast to the extrinsic information transfer (EXIT) chart [28], which is based on the degree distribution of LDPC codes, a base matrix is used for the PEXIT chart. The PEXIT chart provides not only the threshold signal to noise (SNR) of LDPC codes but also the growth of reliability for each type of node in the iterative decoding process.

Let $J(\sigma)$ denote the mutual information between a binary random variable X with $Pr(X = \mu) = Pr(X = -\mu) = 1/2$, and a continuous Gaussian random variable Y with mean X and variance $\sigma^2 = 2\mu$. The function $J(\sigma)$ is calculated as follows:

$$J(\sigma) = 1 - \int_{-\infty}^{+\infty} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\left(y-\sigma^2/2\right)^2}{2\sigma^2}} \cdot \log_2(1+e^{-y})dy. \tag{8}$$

When a rate R LDPC code is used, the mutual information for the channel messages to type-j variable nodes is $I_{ch}(j) = J(\sqrt{8R \cdot SNR_{th}})$, where R is the code rate of LDPC codes, and SNR_{th} is the iterative decoding threshold SNR of LDPC codes. Then, transmitted mutual information between check nodes and variable nodes are calculated iteratively as iterative decoding method. The V2C mutual information between a type-i check node and a type-j variable node can be evaluated as

$$I_{V2C}(i,j) = J\left(\sqrt{\sum_{s \neq i}^{M_p} b_{(s,j)} \left[J^{-1}(I_{C2V}(s,j))\right]^2 + \left[J^{-1}(I_{ch}(j))\right]^2}\right), \tag{9}$$

Similarly, the C2V mutual information between a type-i check node and a type-j variable node can be represent as follows:

$$I_{C2V}(i,j) = 1 - J\left(\sqrt{\sum_{s \neq j}^{N_p} b_{(i,s)} \left[J^{-1}(1 - I_{V2C}(i,s))\right]^2}\right). \tag{10}$$

Using I_{ch} and the updated I_{C2V} at each iteration, the cumulative mutual information (CMI) for each type of variable node can be obtained as

$$I_{CMI}(j) = J\left(\sqrt{\sum_{s}^{M_p} b_{(s,j)} \left[J^{-1}(I_{C2V}(s,j))\right]^2 + \left[J^{-1}(I_{ch}(j))\right]^2}\right). \tag{11}$$

Since the mutual information of variable nodes increases at different rates, it is more reasonable to apply different threshold values for each type of variable node rather than an equal threshold value as in conventional FC schemes.

We decided the initial thresholds from the increment of CMI. Let $I_{CMI}^{l}(j)$ denote the CMI of a type-j variable node at the l-th iteration. The end point of minimum increment from the $(l-1)$-th iteration to the l-th iteration is calculated as

$$l^*(j) = \min_{l} \left(I_{CMI}^{l}(j) - I_{CMI}^{l-1}(j)\right). \tag{12}$$

The proposed technique uses APP-LLR at the $l^*(j)$-th iteration as the initial threshold value, $t_{v,ini}(j)$. APP-LLR can be obtained by applying the J^{-1} function to I_{CMI} as follows:

$$t_{v,ini}(j) = \frac{1}{2}\left[J^{-1}\left(I_{CMI}^{l^*(j)}(j)\right)\right]^2. \tag{13}$$

The deactivating process reduces the amount of variable node operations as the ratio of deactivated variable nodes gradually increases. However, the deactivation disturbs the decoding because some hastily deactivated variable nodes provide insufficient V2C LLR values whose magnitudes are not large enough. Therefore, we enabled the decoder to increase the threshold $t_v(j)$ several times in the middle of the decoding procedure as follows:

$$t_v(j) \leftarrow t_v(j) + \Delta t_v(j) \tag{14}$$

The threshold increasing operation is depicted in Fig. 3. However, the unlimited increasing threshold may reduce the ratio of deactivated nodes. Therefore, we set the threshold limit based on the PEXIT chart as the initial threshold. In this case, we found a maximum increment as follows:

$$l^{\dagger}(j) = \max_{l} \left(I_{\mathrm{CMI}}^{l}(j) - I_{\mathrm{CMI}}^{l-1}(j)\right), \tag{15}$$

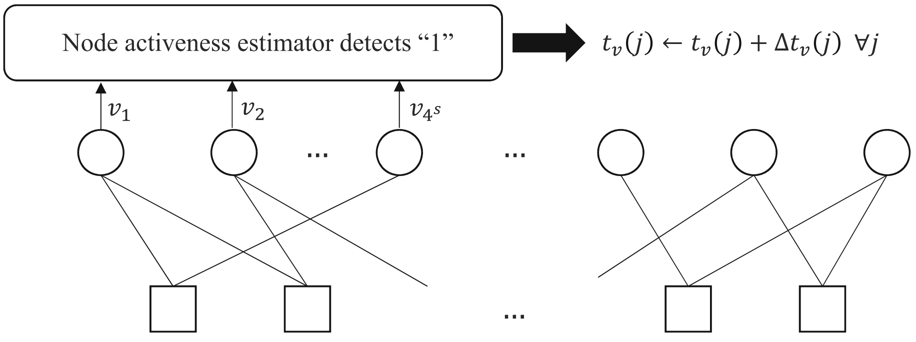
Node activeness estimator detects "1"
$t_v(j) \leftarrow t_v(j) + \Delta t_v(j)\ \forall j$
v_1
v_2
v_{4^s}
...
...
...

Fig. 3 Increasing threshold t_v when the node activeness estimator detects "1"

$$t_{v,max}(j) = \frac{1}{2}\left[J^{-1}\left(I_{\text{CMI}}^{I^{\dagger}(j)}(j)\right)\right]^2. \tag{16}$$

Figure 4 shows an example of CMI and ranges for expected $t_{v,ini}$ and $t_{v,max}$ when a rate-1/2 LDPC code specified in the IEEE 802.11ad standard (WiGig)[29] is used. Because there are 16 types of variable nodes, several behaviors of CMI are depicted in Fig. 4.

When the value of threshold $t_v(j)$ is greater than $t_{v,max}(j)$, it is forced to $t_{v,max}(j)$. Increments of the threshold $\Delta t_v(j)$ are determined by the number of desired increment trials T and the limit of threshold $t_{v,max}(j)$.

$$\Delta t_v(j) = \frac{t_{v,max}(j) - t_{v,ini}(j)}{T} \tag{17}$$

The threshold t_v is increased when the output of a simple device called a node activeness estimator (NAE) is 1. The architecture of NAE is shown in Fig. 5. The estimator is constructed by concatenating AND gates and OR gates where it has 4^s inputs with a positive integer s, $v_1, v_2, ...v_{4^s}$, and one output. The parameter s is the number of stages that consist of AND gates and OR gates. First, choose 4^s out of n variable nodes. For the sake of simplicity, let the chosen variable nodes correspond to the first 4^s columns of H. Then determine the inputs of the estimator as follows:

$$v_i = \begin{cases} 1, & \text{if } i - \text{th variable node is deactivated} \\ 0, & \text{if } i - \text{th variable node is activated} \end{cases} \tag{18}$$

The NAE shown in Fig. 5 detects 1 with high probability when the ratio of 1s among the inputs is greater than 0.7. The probability that the NAE detects 1 can be calculated using the following equation recursively:

$$P_i = 1 - \left(1 - P_{i-1}^2\right)^2 \tag{19}$$

The parameter P_i is the probability that the output is 1 when the binary inputs are processed through a set of AND-OR logical circuits i times. The parameter P_0 is the proportion of the deactivated variable nodes in the 4^s samples. For instance, when the NAE consists of three stages, the probability of detection is $P_3 = 1 - (1-(1-(1-(1-(1-P_0^2)^2)^2)^2)^2)^2$. As a result, when the density of deactivated variable nodes is equal to 0.7, the probability of detection is 0.86 when $s = 3$ and increases to 0.94 when $s = 4$. In other words, the deactivating threshold t_v is increased with high probability

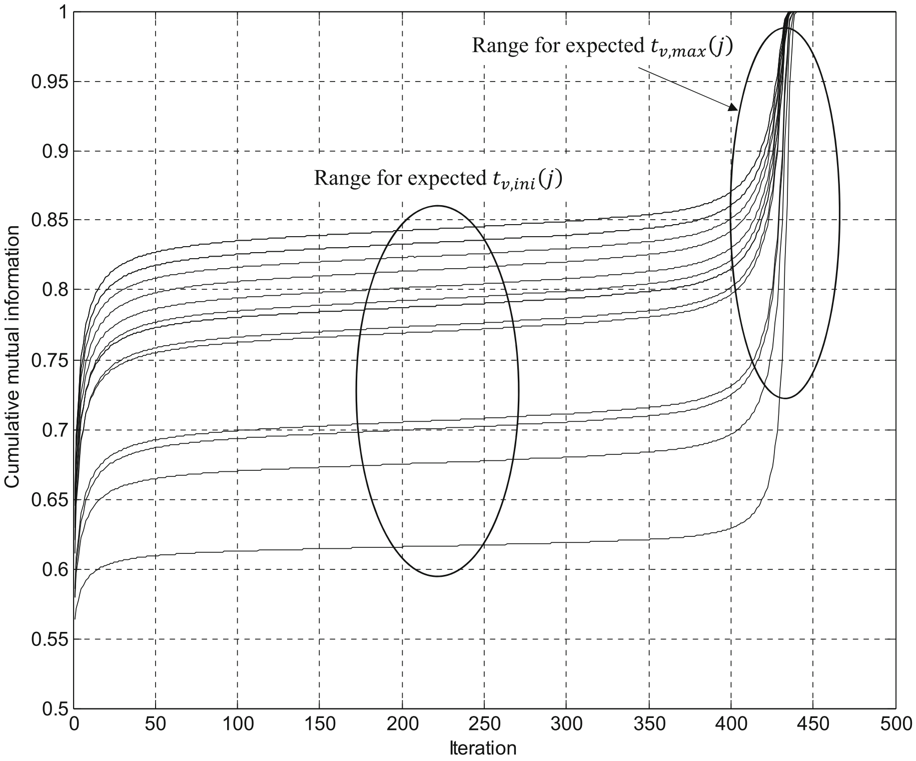

Fig. 4 Cumulative mutual information as a function of the number of iterations for each type of variable node when a rate-1/2 LDPC code specified in IEEE 802.11ad is used

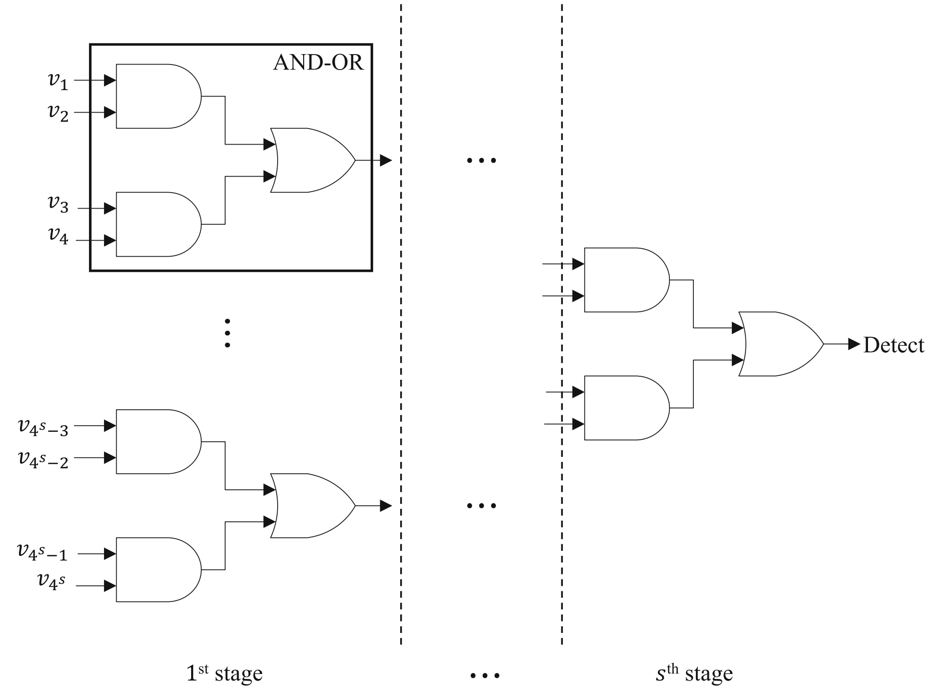

Fig. 5 An architecture of the NAE with 4^s inputs and 1 output

when more than 70% of the variable nodes are deactivated. Figure 6 shows the simulation result of probability that the NAE detects 1 as the density of binary inputs varies.

3.2 Zero-forcing scheme for unreliable information

In the check node operation for normalized MSA, it is required to find the minimum and second minimum values among the V2C messages that come from connected variable nodes. It can be simplified when some V2C messages are equal to 0. For example, let a check node m receive more than two zero-V2C messages. Then the check node does not have to calculate any C2V messages because all of the check node outputs are zero in this case. Consequently, it is known that zero-V2C messages enable check nodes to execute fewer operations. However, V2C messages are generally non-zero values in the conventional decoding algorithm. In order to generate zero-V2C messages, we proposed a threshold-based zero-forcing scheme. In the proposed method, V2C messages, which are regarded as unreliable information, are forced to be zero.

Then, check nodes can selectively calculate C2V LLR values from the received V2C LLRs. For a check node m, if more than two zero V2C LLR values are received, then every C2V LLR from the check node m, $U_{mn}^{(itr)}$ $\forall n$, is set to zero without any calculation. If only one zero V2C LLR, say $V_{mn}^{(itr-1)} = 0$, is received, the check node operation is executed only for $U_{mn}^{(itr)}$. The effects of zero-forced V2C LLRs on check node operations are depicted in Fig. 7. Because less calculation is required on the check-node side as more V2C LLRs are replaced with zero, a higher t_c is helpful for complexity. However, too high t_c results in performance degradation.

Thresholds for identifying unreliable information are obtained by using the PEXIT chart as thresholds for variable nodes. In this case, the mutual information I_{V2C}, which is transmitted from variable nodes to check nodes, is used. From the PEXIT chart, there are M_pN_p kinds of I_{V2C} that exist. Although we could achieve the thresholds for each type of V2C message, a large amount of memory is required for thresholds. Therefore, an equal threshold t_c is used for every V2C message from an equal variable node. The threshold t_c for each variable node is approximately calculated by averaging the values that are obtained by applying J^{-1} function to the corresponding I_{V2C}. Let $l^{\ddagger}(i,j)$ denote the number of iterations when the increment of I_{V2C} is a minimum. Then, the calculation of threshold for V2C messages from type-j variable nodes $t_c(j)$ can be represented as follows:

$$l^{\ddagger}(i,j) = \min_{l} \left(I_{V2C}^{l}(i,j) - I_{V2C}^{l-1}(i,j) \right), \tag{20}$$

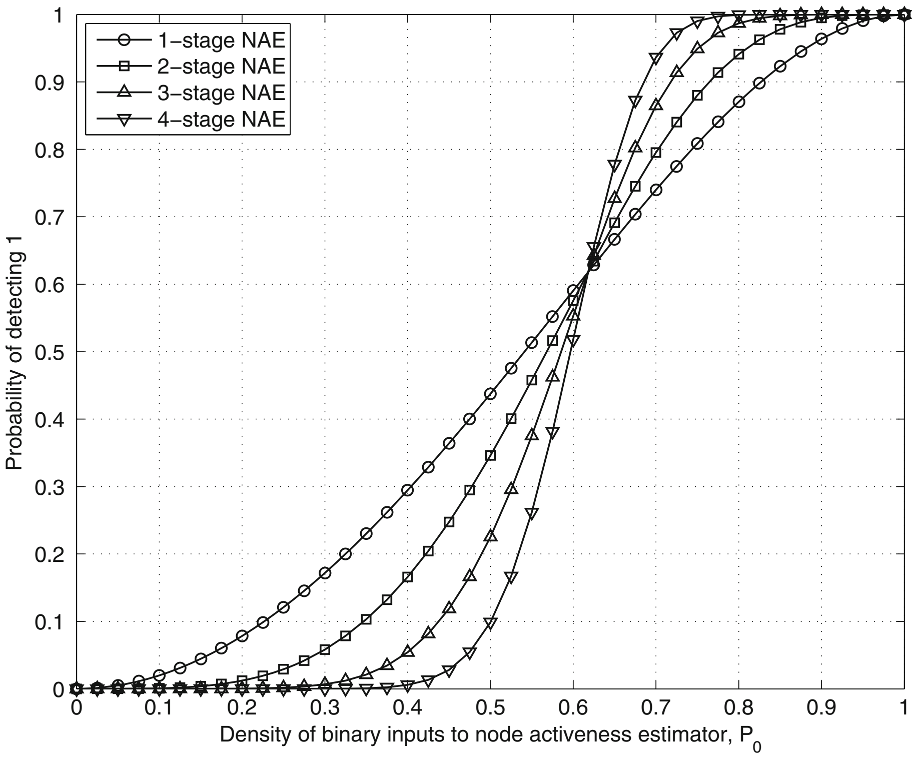

Fig. 6 Probability that the NAE detects 1 versus density of input vectors

$$t_c(j) = \frac{1}{d_{v,j}} \sum_{s=1}^{M_p} b_{(s,j)} \frac{1}{2} \left[J^{-1}\left(I_{V2C}^{l^{\ddagger}(s,j)}(s,j) \right) \right]^2, \qquad (21)$$

where $d_{v,j}$ is the degree of type-j variable nodes. Figure 8 shows an example of V2C LLR and a range for expected t_c when the rate-1/2 LDPC code specified in IEEE 802.11ad is used. Similar to the case of CMI, because there are 16 types of variable nodes, several behaviors of V2C LLR are depicted in Fig. 8.

Since a V2C LLR value is said to be unreliable when its magnitude is small, our method forces V2C LLR values whose magnitudes are less than threshold to be zero as follows:

$$V_{mn}^{(itr)} = \begin{cases} 0, & \text{if } \left| V_{mn}^{(itr)} \right| \le t_c(j), n \in \text{type-}j \text{ variable nodes} \\ V_{mn}^{(itr)}, & \text{otherwise} \end{cases} \qquad (22)$$

For LDPC codes of length N and rate $R = \frac{N-M}{N}$, the proposed simultaneous node deactivation and zero-forcing scheme for highly reliable and unreliable information is described in Algorithm 1.

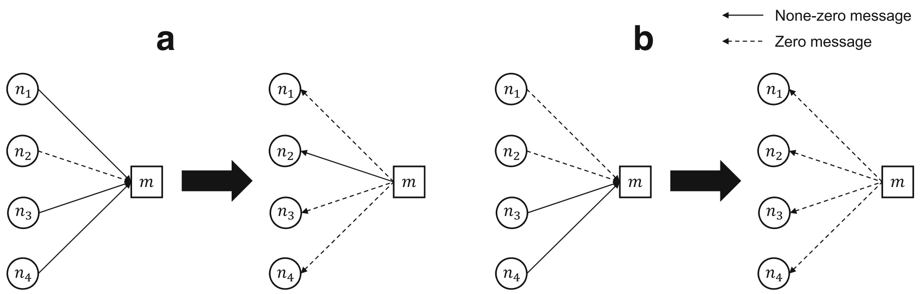

Fig. 7 Effects of zero V2C LLRs on check node operations. Dashed lines represent zero-LLR V2C or C2V messages. **a** The case where one zero V2C LLR is received. **b** The case where two zero V2C LLRs are received

Algorithm 1 Normalized Min-sum Decoding with the Proposed Algorithm

```
 1: Initialization:
    $itr = 1$, set thresholds $t_v = t_{v,ini}$, $t_c$, and the maximum iteration $I_{max}$.
    $U_{mn}^{(0)} = 0$ and $\beta_n^{(0)} = V_{mn}^{(0)} = U_{n,ch}$ for all $(m, n)$.
 2: for every check node ($m = 1, \cdots, M$) do
 3:   if ($V_{mn}^{(itr-1)} == 0$ and $V_{mn'}^{(itr-1)} \neq 0$ $(n' \neq n)$) then
 4:     Calculate $U_{mn}^{(itr)}$ using (4) with scaling factor and $U_{mn'}^{(itr)} = 0$ $(n' \neq n)$
 5:   else if (more than two V2C LLRs are zero) then
 6:     $U_{mn}^{(itr)} = 0\ \forall n$
 7:   end if
 8: end for
 9: for every variable node ($n = 1, \cdots, N$) do
10:   if $\left(\left|\beta_n^{(itr-1)}\right| \geq t_v(j)\right)$ then
11:     Deactivate $n$-th variable node.
12:     $\beta_n^{(itr)} = \beta_n^{(itr-1)}$ and $V_{mn}^{(itr)} = V_{mn}^{(itr-1)}$
13:   else
14:     Calculate $V_{mn}^{(itr)}$ and $\beta_n^{(itr)}$ using (2) and (3)
15:     if $\left(\left|V_{mn}^{(itr)}\right| < t_c(j)\right)$ then
16:       $V_{mn}^{(itr)} = 0.$
17:     end if
18:   end if
19: end for
20: if stopping condition is not satisfied then
21:   if node activeness estimator detects 1 then
22:     Reactivate all variable nodes
23:     $t_v(j) \leftarrow t_v(j) + \Delta t_v(j)\ \forall j$
24:     if $t_v(j) > t_{v,max}(j)$ then
25:       $t_v(j) = t_{v,max}(j)$
26:     end if
27:   end if
28:   Go back to line 2
29: end if
```

4 Simulation results

In order to examine the performance of the proposed algorithm, simulations were performed over an additive white Gaussian noise (AWGN) channel with binary phase shift keying (BPSK) modulation. LDPC codes specified in the IEEE 802.11ad standard of length $N = 672$ and rate $R = 1/2$ and 3/4 were used to evaluate performance. In addition, the simulation results for LDPC codes specified in the IEEE 802.22 standard [30] of length $N = 384$ and IEEE 802.11n standard [31] of length $N = 1944$ are represented in the table at the end of this section. We performed all simulations using 7-bit fixed-point precision, with 4 bits for the integer part and 3 bits for the fractional part. The maximum number of iterations for all simulations was 20.

First, the frame error rates (FERs) of the proposed adaptive deactivation and zero-forcing (ADZF) scheme are compared to those of the normalized MSA without any additional manipulation, and the conventional FC schemes, as shown in Fig. 9. Performance of the proposed scheme was evaluated when the number of stages in NAE was equal to 4, and the number of increment trials was 10. The simulation results of the FC scheme applied to WiGig LDPC code with $R = 1/2$ show the influence of the threshold value on the error correcting performance of the FC scheme in a high SNR regime. Even though significant complexity reduction can be achieved by the FC scheme using a low threshold value, the performance loss in the target SNR region makes the scheme meaningless. Figure 10 shows the influence of threshold value on the required SNR for achieving FER $= 10^{-3}$. As represented in Fig. 10, in order to overcome performance loss, a sufficiently large threshold value should be used. In this section, a threshold value that show similar error performance at FER $= 10^{-3}$, such as $t = 9$ or 10, was used for the comparison of complexity.

Next, we evaluated the average number of iterations for each scheme. Figure 11 shows the average number of iterations where the maximum iteration $I_{max} = 20$. The ADZF and FC schemes require a similar average number of iterations to the normalized MSA. The reason for slight difference in iterations at low SNR region $\left(\text{FER} < 10^{-2}\right)$ is that the initial thresholds $t_{v,ini}$ in the proposed ADZF scheme are smaller than the threshold value for the FC scheme. Figure 12 shows the influence of threshold value on the average number of iterations for achieving FER $= 10^{-2}$. When a smaller threshold value is used, the average number of iteration for maintaining error performance increases. In the proposed ADZF scheme, when the NAE detects 1, the threshold values increase from $t_{v,ini}$, which can be lower than 6, to $t_{v,max}$, which can be higher than 10. Since the NAE is operated at the end of each iteration, the opportunity for threshold increment increases at the low SNR region, where more iteration is required compared to the high SNR region. In other words, the thresholds for the proposed ADZF scheme can be higher than that of the FC schemes at the low SNR region. Nevertheless, since the difference in iterations decreases to zero at practical operation region $\left(\text{FER} \geq 10^{-2}\right)$, it does not affect system's performance significantly.

Figure 13 shows the variable node complexity of each scheme when the complexity of the normalized MSA is evaluated as 1. The variable node complexity is defined as follows:

$$\sum_{itr=1}^{I_{max}} \frac{\sum_{i=1}^{N} \left(1 - v_i^{itr}\right) \cdot d_{v,i}}{\sum_{i=1}^{N} d_{v,i}} \tag{23}$$

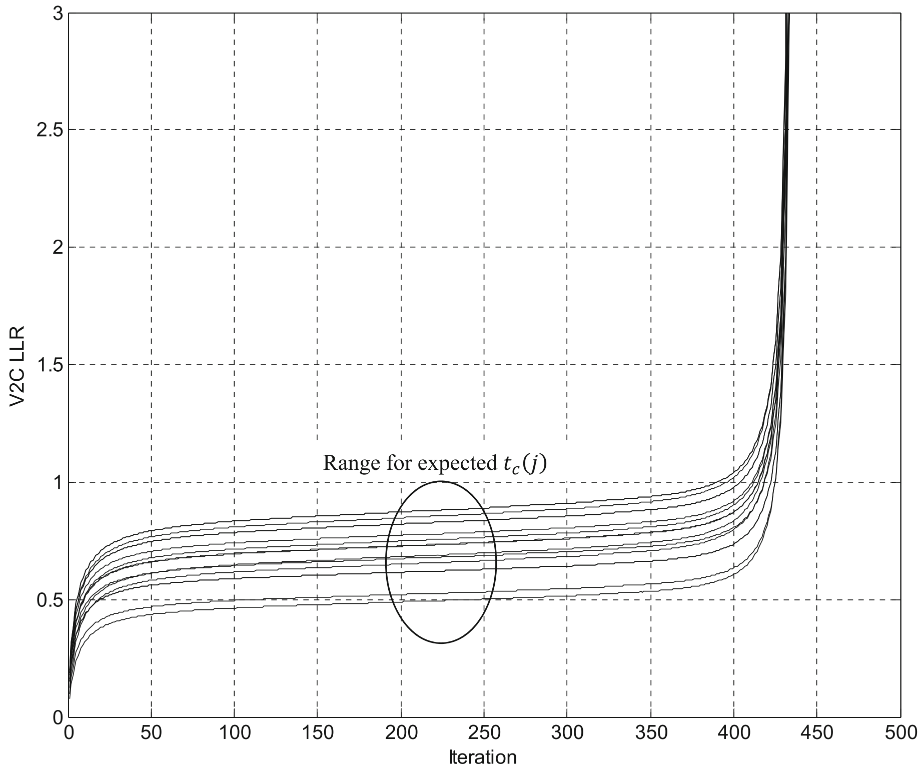

Fig. 8 Average V2C LLR curve for each type of variable node when a rate-1/2 LDPC code specified in IEEE 802.11ad is used

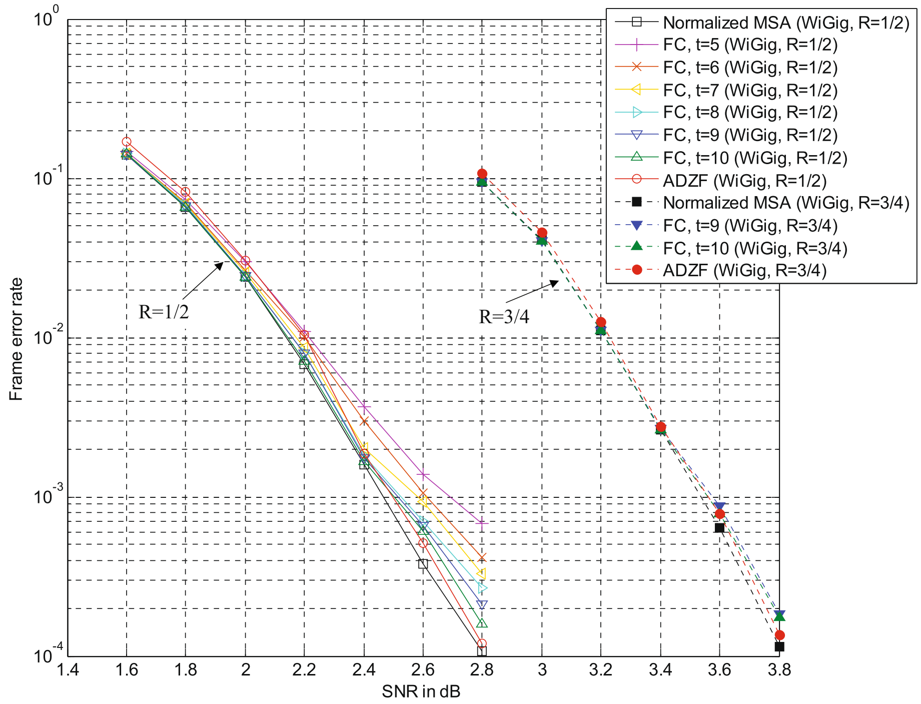

Fig. 9 Frame error rates of the normalized min-sum algorithm, conventional FC scheme, and proposed algorithm (ADZF)

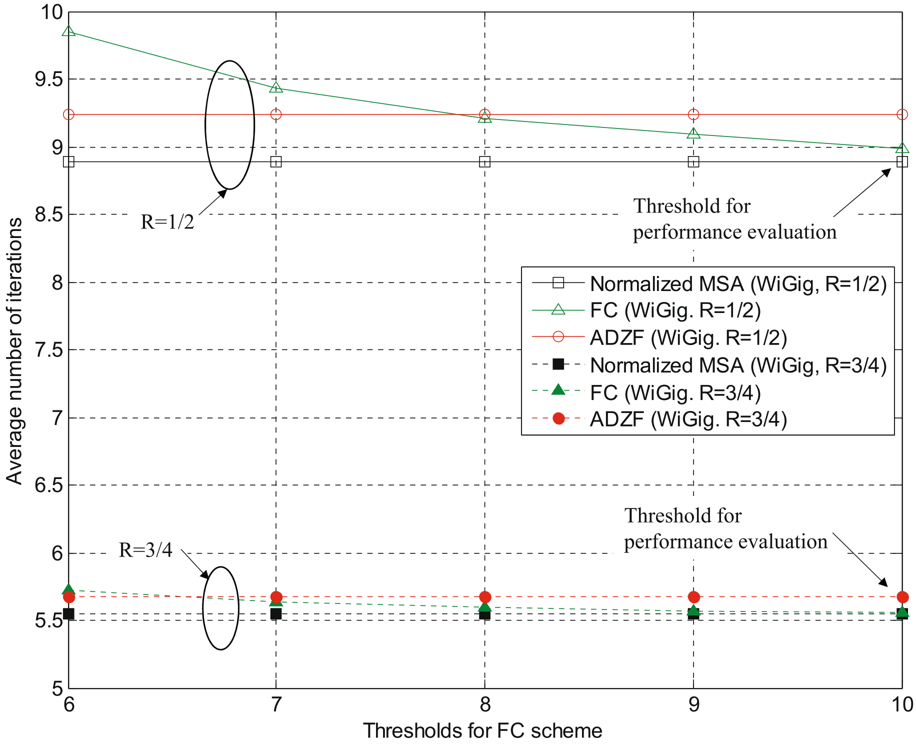

Fig. 10 Required SNR for achieving an FER $= 10^{-3}$

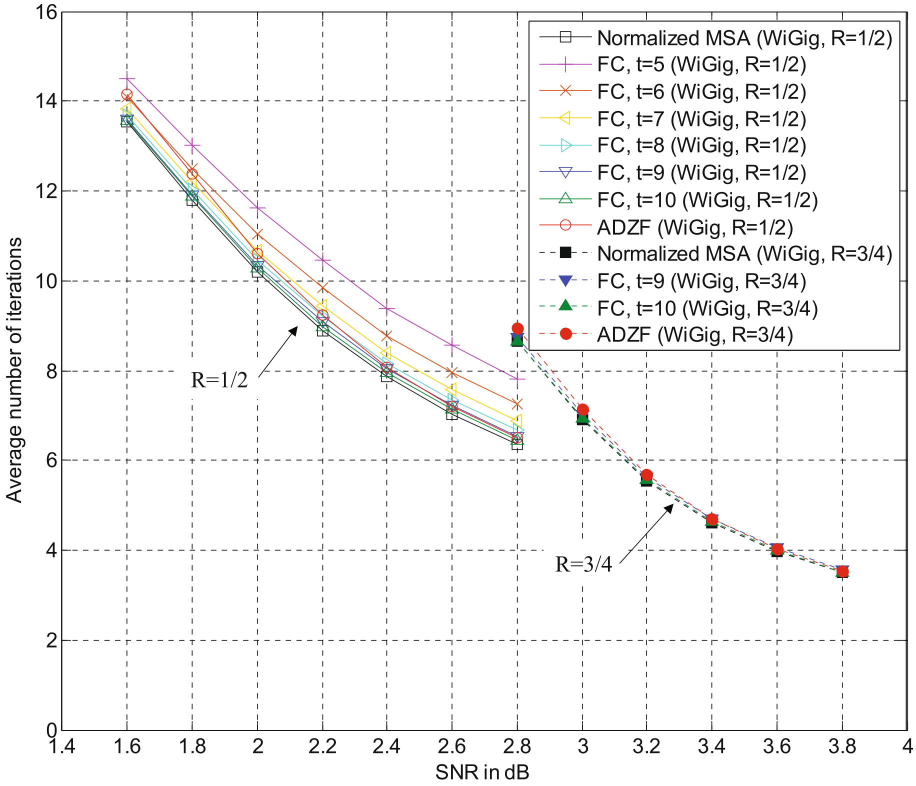

Fig. 11 Average number of iterations of the normalized min-sum algorithm, conventional FC scheme, and proposed algorithm (ADZF) where $I_{max} = 20$

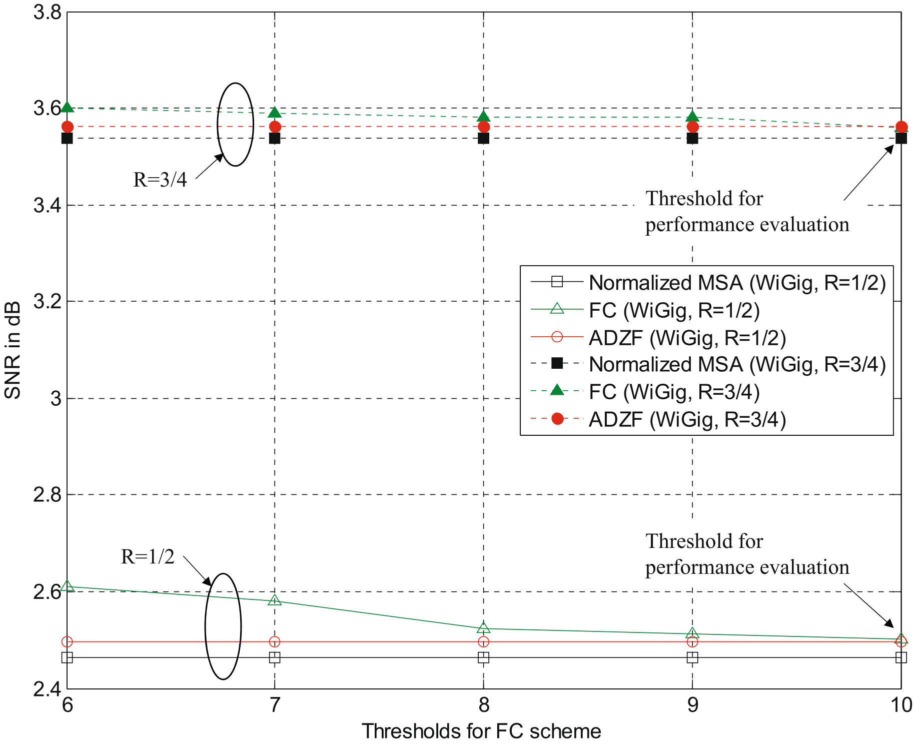

Fig. 12 Average number of iterations of the normalized min-sum algorithm, conventional FC scheme with various threshold values, and proposed algorithm (ADZF) where $I_{max} = 20$ and FER $= 10^{-2}$

The parameter v_i^{itr} means that the i-th variable node is activated(=0) or deactivated(=1) at the itr-th iteration. Because the effect of a deactivated high degree variable node on computation complexity is greater than that of a deactivated low degree variable node, we considered the degree of variable nodes as a weight. When the iteration is stopped before reaching the maximum number of iterations, all v_i^{itr} values for the rest of the iterations are equal to 1. In this way, a reasonable complexity considering early stopping can be obtained. For the proposed scheme and FC schemes, the complexity decreases as SNR increases, because each variable node is likely to have a higher APP-LLR at a higher SNR. As represented in Fig. 13, the variable node complexity of the proposed scheme applied to WiGig LDPC code of rate 1/2 is only 0.26, which is almost half of that of the FC scheme (0.53). When the threshold value for the FC scheme is smaller than 6, the complexity of the FC scheme is lower than that of the proposed scheme. However, as represented in Fig. 9, the FC scheme with a low threshold value is meaningless due to the error performance loss. For the proposed scheme, using an NAE with $s = 3$ shows less complexity than that with $s = 4$. The reason is that the probability of detecting 1 becomes higher when NAE has more stages, as shown in Fig. 6. Although more stages in NAE cause less complexity reduction compared to fewer stages, it has an advantage in the viewpoint of error floor. Similarly, the number of increment trials also influences error performance and decoding complexity. The large number of trials brings low decoding complexity and early error floor. On the contrary, lower complexity gain with less performance loss can be obtained when the smaller number of increment trials is used. Consequently, the number of stages in NAE and increment trials should be selected cautiously in accordance with the requirements of the communication system due to a trade-off between complexity and error performance.

Check node complexity is shown in Fig. 14, which is the effect of zero-forced V2C LLRs as explained in Section 3, and deactivated variable nodes. Check node complexity is obtained by evaluating the amount of check node calculations:

$$\sum_{itr=1}^{I_{max}} \frac{\sum_{i=1}^{M} \left(1 - c_i^{itr}\right) \cdot d_{c,i}}{\sum_{i=1}^{M} d_{c,i}} \tag{24}$$

The parameter $d_{c,i}$ is the degree of the i-th check node, and c_i^{itr} means that the operation of the i-th check node can be omitted (=1) or not (=0) at the itr-th iteration. Similar to the calculation of variable node complexity, the degree of check node is considered as a weight. In addition, when the iteration is stopped before reaching the

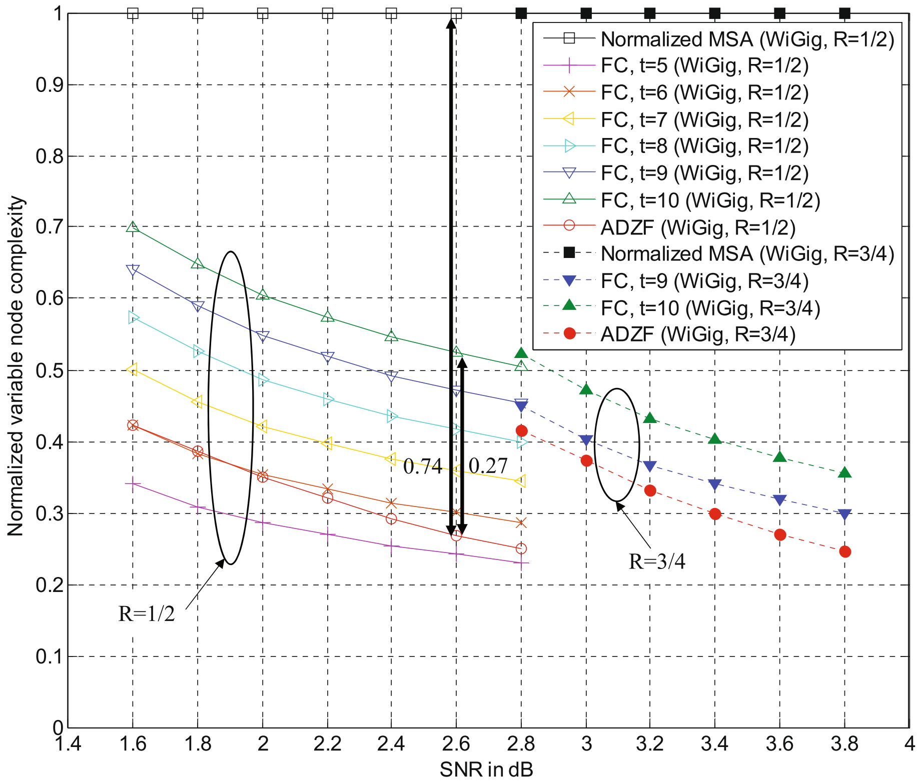

Fig. 13 Normalized variable node complexity of the normalized min-sum algorithm, conventional FC scheme, and proposed algorithm (ADZF)

maximum number of iterations, all c_i^{itr} values for the rest of the iterations are equal to 1. As mentioned previously, when a check node receives more than two zero-V2C messages, the corresponding check node operation also can be omitted, because all C2V messages from that check node are zero. In addition, because the operation of check nodes is meaningless when all related variable nodes are deactivated, we can omit the operation of corresponding check nodes. On the other hand, the FC scheme has only the effect of deactivated variable nodes. Moreover, as represented in Fig. 13, the number of deactivated variable nodes in the FC scheme is less than that in the proposed scheme. As a result, the check node complexity of the proposed scheme applied to WiGig LDPC code of rate 1/2 is 0.55, which is lower than that of the FC scheme (0.76). Complexity gain increases as SNR increases because the the portion of deactivated variable nodes is increased. Meanwhile, the influence of the SC scheme on the check node complexity reduction is decreased due to the increment of channel information reliability. Figure 15 shows the percentage of V2C messages for which the absolute value of LLR is lower than 0.5 and 1.0 as a function of SNR and iteration. As shown in Fig. 15, the percentage of unreliable V2C messages decreases significantly while SNR is increased.

Table 1 shows the effect of the proposed ADZF scheme when other LDPC codes are used. The average number of iterations, variable node complexity, and check node complexity of each scheme for obtaining FER $= 10^{-2}$ and 10^{-3} are represented. The represented variable node complexity and check node complexity are normalized by the complexity of the normalized MSA. Similar to the results for WiGig LDPC codes, the proposed scheme shows significantly lower node operation complexity compared to FC scheme. If the threshold value is smaller than the value represented in the Table 1, early error floor occurs. Figure 16 shows the error performance of each scheme when the smaller threshold value is used. To make it easier to see, we represent results only at the SNR regions that the FER of normalized MSA is approximately 10^{-2} and 10^{-3}. As represented in Fig. 16, when the smaller threshold is used, there is a noticeable performance gap at the SNR $= 2.8$ and 3.0 dB due to the early error floor.

For the sake of fair comparison, we should consider the additional complexity which is caused by variable node deactivation, zero-forcing message, and NAE. Then, the total complexity of ADZF scheme C_{ADZF} is calculated as

$$C_{\text{ADZF}} = C_{\text{variable}} + C_{\text{check}} + C_{\text{D}} + C_{\text{ZF}} + C_{\text{NAE}}, \quad (25)$$

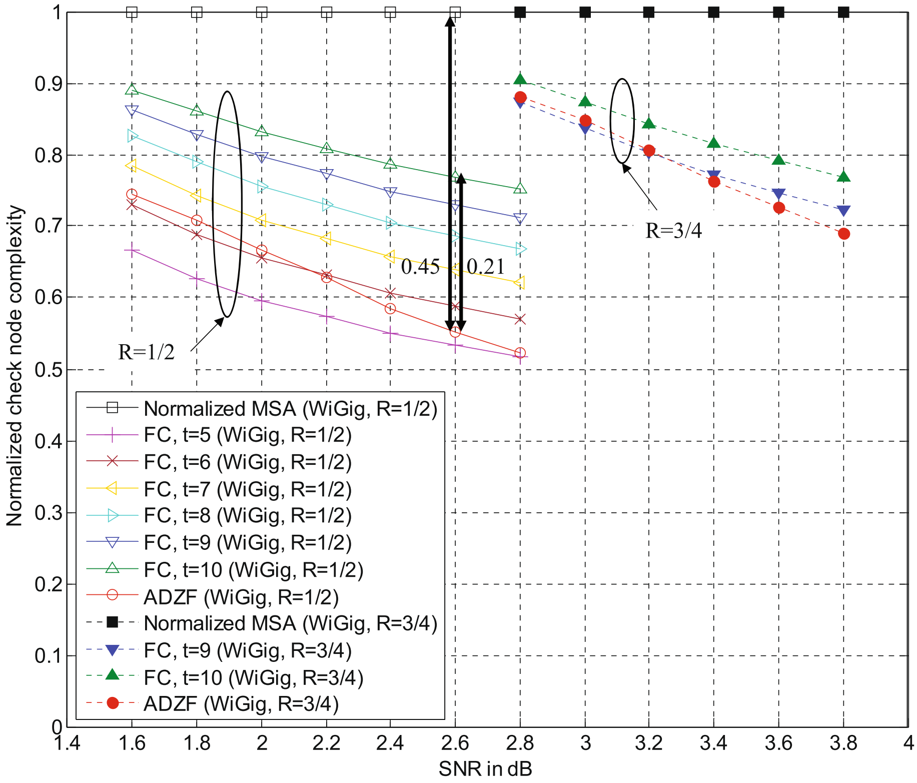

Fig. 14 Normalized check node complexity of the normalized min-sum algorithm, conventional FC scheme, and proposed algorithm (ADZF)

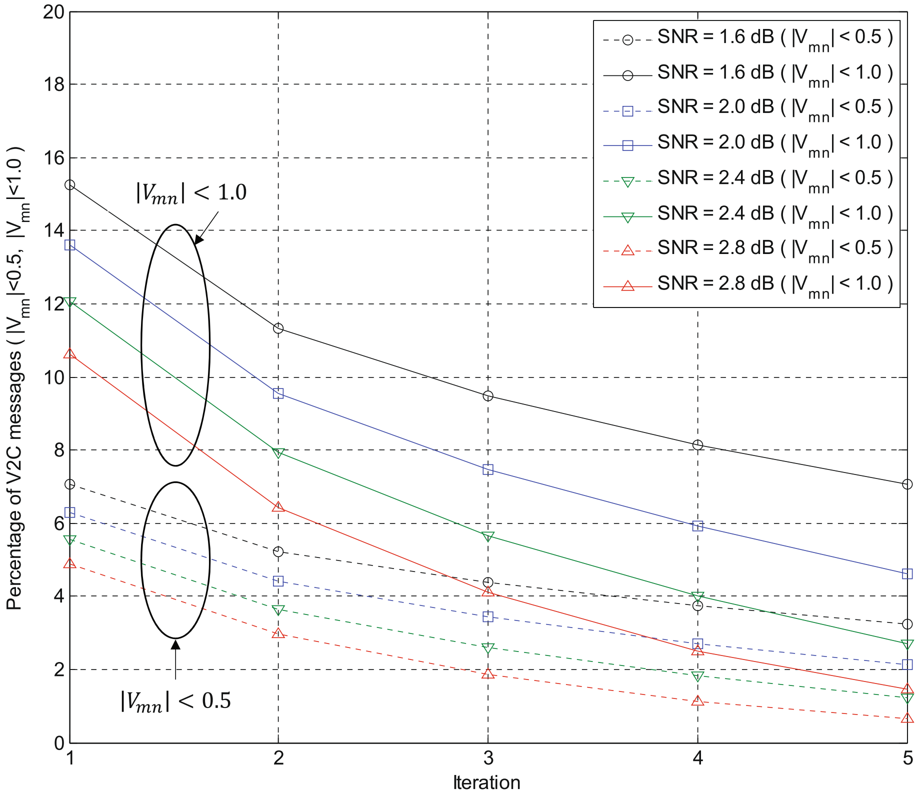

Fig. 15 Percentage of V2C messages for which magnitude is lower than 0.5 and 1.0

Table 1 Simulation results of the FC scheme and proposed ADZF scheme when 802.22 LDPC codes and 802.11n LDPC codes are used

Code	FER	Scheme	Average iteration	VN complexity	CN complexity
802.22 LDPC	10^{-2}	FC ($t = 10$)	6.637	0.631	0.894
($N = 384, R = 1/2$)		ADZF (proposed)	7.050	0.373	0.707
	10^{-3}	FC ($t = 10$)	5.186	0.586	0.861
		ADZF (proposed)	5.408	0.320	0.634
802.11n LDPC	10^{-2}	FC ($t = 12$)	9.091	0.531	0.854
($N = 1944, R = 1/2$)		ADZF (proposed)	9.390	0.225	0.507
	10^{-3}	FC ($t = 12$)	7.207	0.508	0.826
		ADZF (proposed)	7.377	0.202	0.458

where C_{variable} and C_{check} are the complexity of variable node operation and check node operation, respectively. The parameters C_{D}, C_{ZF} and C_{NAE} are the complexity of deactivation, zero-forcing, and NAE, respectively. Similarly, the total complexity of FC scheme C_{FC} is calculated as follows:

$$C_{\text{FC}} = C_{\text{variable}} + C_{\text{check}} + C_{\text{D}} \tag{26}$$

In order to combine the complexity of different operations, we first checked the number of additions, comparisons, and AND-OR logical circuits that are basic components of each operation. Variable node operation consists of addition. Comparison is used for check node operation, deactivation, and zero-forcing process. At last, node activeness estimator consists of AND-OR logical circuits as shown in Fig. 5. Then, we decided the weight of each component, since the computational complexity of each component are different. Using the number of components and their weight, the Eqs. (25) and (26) can be rewritten as

$$\begin{aligned} C_{\text{scheme}} = W_{\text{add}} \cdot N_{\text{add}} + W_{\text{comp}} \cdot N_{\text{comp}} \\ + W_{\text{AND-OR}} \cdot N_{\text{AND-OR}}, \end{aligned} \tag{27}$$

where N_{add}, N_{comp} and $N_{\text{AND-OR}}$ represent the number of additions, comparisons, and AND-OR logical circuits, respectively. The parameters W_{add}, W_{comp}, and $W_{\text{AND-OR}}$ mean the weight of addition, comparison, and AND-OR logical circuit, respectively. To quantify the computational complexity of each component, we used execution time. From the simulation, we obtained that the execution time for the one AND-OR logical circuit is approximately 1.5 times longer than that of addition and comparison. Therefore, the weight of the AND-OR logical circuit is 1.5 when that of the addition and comparison is 1. Figure 17 shows the normalized total complexity of each scheme when the total complexity of normalized-MSA is 1. Simulated error rate region is FER $= 10^{-3}$, and threshold of FC scheme is $t = 12$ for 802.11n code and $t = 10$ for the rest. As a result, even though ADZF scheme has extra operations

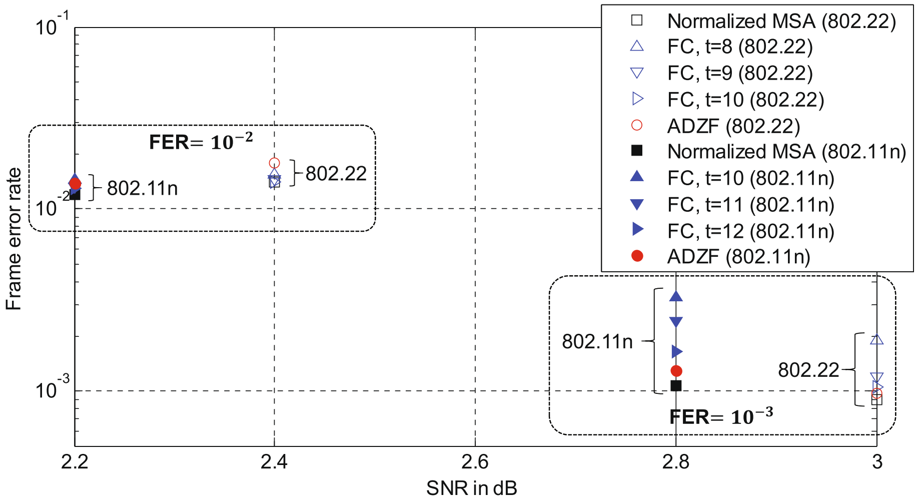

Fig. 16 Frame error rate of each scheme when 802.22 LDPC codes and 802.11n LDPC codes are used. The simulated error region is that the frame error rate of normalized min-sum is approximately 10^{-2} and 10^{-3}

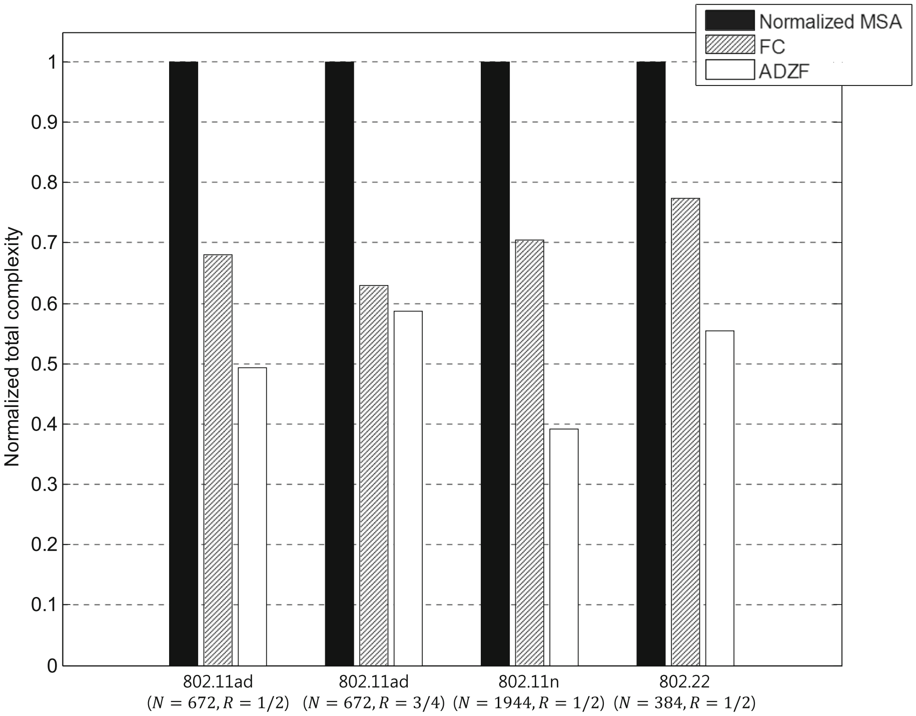

Fig. 17 Normalized total complexity of normalized min-sum algorithm, conventional FC scheme, and the proposed algorithm (ADZF) when FER= 10^{-3}

compared with FC scheme, it still have benefit in the viewpoint of complexity.

Finally, as a merit of the proposed scheme, ADZF scheme is compatible with the various existing schemes using LDPC codes, such as [32–34], in the viewpoint of the practical usage of their schemes. Decoding complexity of their schemes is reduced through simply adopting proposed scheme to LDPC decoder.

5 Conclusions

A node operation reducing technique, called the ADZF scheme, is proposed for a low-complexity decoder of LDPC codes. Variable nodes with high reliability are deactivated to omit their calculations. A simple NAE constructed with logical circuits is introduced to increase the threshold value when deactivated variable nodes are saturated. Hastily deactivated variable nodes are automatically reactivated by the increased threshold value. Messages with low reliability propagated from variable nodes are replaced with zeroes to reduce check node operations. Since the threshold value is decided only once in the offline mode, the proposed threshold value decision method does not incur additional computational complexity for the decoder. Simulation results show that the proposed scheme reduced decoding complexity more than the FC scheme. Moreover, the proposed scheme resolves the early error floor problem, which is a drawback of the FC scheme.

Acknowledgements
This research was supported by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2017-2015-0-00385) supervised by the IITP (Institute for Information and communications Technology Promotion).

Authors' contributions
TK designed the main of the algorithm, analyzed the data, and wrote this paper. JB and ML designed a part of the algorithm and completed simulations. JH gave valuable suggestions on the idea of this paper. All authors read and approved the final manuscript.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Author details
[1]The School of Electrical Engineering, Korea University, 145, Anam-ro, Seongbuk-gu, 02841 Seoul, Republic of Korea. [2]LG electronics, 19, Yangjae-daero 11-gil, Seocho-gu, 06772 Seoul, Republic of Korea. [3]Samsung electronics, 1-1, Samsungjeonja-ro, 18448 Hwaseong-si, Republic of Korea.

Received: 20 March 2017 Accepted: 22 August 2017
Published online: 11 September 2017

References

1. R Gallager, Low-density parity-check codes. IRE Trans. Inf. Theory. **8**(1), 21–28 (1962)
2. DJC Mackay, Good error-correcting codes based on very sparse matrices. IEEE Trans. Inf. Theory. **45**(2), 399–431 (1999)
3. C Berrou, A Glavieux, P Thitimajshima, in *IEEE International Conference on Communications (ICC 1993)*. Near Shannon limit error-correcting coding and decoding: Turbo-codes. 1 (IEEE, Geneva, 1993)

4. SK Chronopoulos, G Tatsis, V Raptis, P Kostarakis, in *Pan-Hellenic Conference on Electronics and Telecommunications (PACET 2012)*. A Parallel turbo encoder-decoder scheme (The Electronics and the Telecommunications laboratories of the Departments of Physics and Electrical and Computer Engineering, of the Aristotle University of Thessaloniki (AUTh), Thessaloniki, 2012)
5. SK Chronopoulos, T Giorgos, P Kostarakis, Turbo codes–a new PCCC design. Commun. Netw. **3**(4), 229–234 (2011)
6. SK Chronopoulos, T Giorgos, P Kostarakis, Turbo coded OFDM with large number of subcarriers. J. Signal Inf. Process. **3**(2), 161–168 (2012)
7. SK Chronopoulos, V Christofilakis, G Tatsis, P Kostarakis, Performance of turbo coded OFDM under the presence of various noise types. Wireless Pers. Commun. **87**(4), 1319–1336 (2016)
8. MPC Fossorier, M Mihaljevic, H Imai, Reduced complexity iterative decoding of low density parity check codes based on belief propagation. IEEE Trans. Commun. **47**(5), 673–680 (1999)
9. J Chen, MP Fossorier, Near optimum universal belief propagation based decoding of low density parity check codes. IEEE Trans. Commun. **50**(3), 406–414 (2002)
10. J Heo, Analysis of scaling soft information on low density parity check code. IEE Electron. Lett. **37**(25), 1530–1531 (2001)
11. V Savin, in *IEEE International Symposium on Information Theory (ISIT 2007)*. Iterative LDPC decoding using neighborhood reliabilities (IEEE, Nice, 2007)
12. V Savin, in *IEEE International Symposium on Information Theory (ISIT 2008)*. Self-corrected min-sum decoding of LDPC codes (IEEE, Toronto, 2008)
13. J Andrade, G Falcao, V Silva, JP Barreto, N Goncalves, V Savin, in *IEEE International Conference on Communications (ICC 2013)*. Near-LSPA performance at MSA complexity (IEEE, Budapest, 2013)
14. E Amador, V Rezard, R Pacalet, in *17th IFIP International Conference on Very Large Scale Integration (VLSI-SoC 2009)*. Energy efficiency of SISO algorithms for turbo-decoding message-passing LDPC decoders (IEEE, Florianopolis, 2009)
15. E Amador, R Knopp, V Rezard, R Pacalet, in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI 2010)*. Dynamic power management on LDPC decoders (IEEE, Lixouri, 2010)
16. O Boncalo, A Amaricai, V Savin, in *21st IEEE International Conference on Electronics, Circuits and Systems (ICECS 2014)*. Memory efficient implementation of self-corrected min-sum LDPC decoder (IEEE, Marseille, 2014)
17. E Yeo, P Pakzad, B Nikolic, V Anantharam, in *IEEE Global Telecommunications Conference (GLOBECOM 2001)*. High throughput low-density parity-check decoder architectures (IEEE, San Antonio, 2001)
18. J Zhang, M Fossorier, Shuffled iterative decoding. IEEE Trans. Commun. **53**(2), 209–213 (2005)
19. CY Lin, MK Ku, in *IEEE International Symposium on Circuits and Systems (ISCAS 2009)*. Node operation reduced decoding for LDPC codes (IEEE, Taipei, 2009)
20. E Zimmermann, P Pattisapu, PK Bora, G Fettweis, in *7th International Symposium on Wireless Personal Multimedia Communications (WPMC 2004)*. Reduced complexity LDPC decoding using forced convergence (WPMC steering board, Abano Terme, 2004)
21. E Zimmermann, W Rave, G Fettweis, in *11th European Wireless Conference 2005-Next Generation Wireless and Mobile Communications and Services*. Forced convergence decoding of LDPC Codes - EXIT chart analysis and combination with node complexity reduction techniques (VDE, Nicosia, 2005)
22. M Sarajlc, L Liu, O Edfors, in *IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC 2014)*. Reducing the complexity of LDPC decoding algorithms: an optimization-oriented approach (IEEE, Washington, 2014)
23. M Sarajlc, L Liu, O Edfors, in *IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC 2015)*. Modified forced convergence decoding of LDPC codes with optimized decoder parameters (IEEE, Hong Kong, 2015)
24. J Fan, H Yang, in *IEEE International Conference on Communications Technology and Applications (ICCTA 2009)*. A new forced convergence decoding scheme for LDPC codes (IEEE, Beijing, 2009)
25. BJ Choi, MH Sunwoo, in *International SoC Design Conference (ISOCC 2013)*. Efficient forced convergence algorithm for low power LDPC decoders (IEEE, Busan, 2013)
26. BJ Choi, MH Sunwoo, in *IEEE Asia Pacific Conference on Circuits and Systems (APCCAS 2014)*. Simplified forced convergence decoding algorithm for low power LDPC decoders (IEEE, Ishigaki Island, 2014)
27. G Liva, M Chiani, in *IEEE Global Telecommunications Conference (GLOBECOM 2007)*. Protograph LDPC codes design based on EXIT analysis (IEEE, Washington, 2007)
28. S ten Brink, Convergence behavior of iteratively decoded parallel concatenated codes. IEEE Trans. Commun. **49**(10), 1727–1737 (2001)
29. (IEEE, New York, 2012). http://standards.ieee.org/getieee802/download/802.11ad-2012.pdf
30. (IEEE, New York, 2011). http://standards.ieee.org/getieee802/download/802.22-2011.pdf
31. (IEEE, New York, 2009). http://standards.ieee.org/getieee802/download/802.11n-2009.pdf
32. M Baldi, N Maturo, G Ricciutelli, F Chiaraluce, Security gap analysis of some LDPC coded transmission schemes over the flat and fast fading Gaussian wire-tap channels. EURASIP J. Wirel. Commun. Netw. **232**(1), 1–12 (2015)
33. M Baldi, N Maturo, E Paolini, F Chiaraluce, One the use of ordered statics decoders for low-density parity-check codes in space telecommand links. EURASIP J. Wirel. Commun. Netw. **272**(1), 1–15 (2016)
34. K Kwon, T Kim, J Heo, Pre-coded LDPC coding for physical layer security. EURASIP J. Wirel. Commun. Netw. **283**(1), 1–18 (2016)