Yoon *et al. EURASIP Journal on Wireless Communications and Networking* (2020) 2020:126
https://doi.org/10.1186/s13638-020-01740-2

EURASIP Journal on Wireless Communications and Networking

REVIEW Open Access

An advanced low-complexity decoding algorithm for turbo product codes based on the syndrome

Sungsik Yoon[1], Byungkyu Ahn[2] and Jun Heo[1]*

*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

Abstract

This paper introduces two effective techniques to reduce the decoding complexity of turbo product codes (TPC) that use extended Hamming codes as component codes. We first propose an advanced hard-input soft-output (HISO) decoding algorithm, which is applicable if an estimated syndrome stands for double-error. In conventional soft-input soft-output (SISO) decoding algorithms, 2^p (p: the number of least reliable bits) number of hard decision decoding (HDD) operations are performed to correct errors. However, only a single HDD is required in the proposed algorithm. Therefore, it is able to lower the decoding complexity. In addition, we propose an early termination technique for undecodable blocks. The proposed early termination is based on the difference in the ratios of double-error syndrome detection between two consecutive half-iterations. Through this early termination, the average iteration number is effectively lowered, which also leads to reducing the overall decoding complexity. Simulation results show that the computational complexity of TPC decoding is significantly reduced via the proposed techniques, and the error correction performance remains nearly the same in comparison with that of conventional methods.

Keywords: Turbo product codes, Syndrome-based decoding, Soft-input soft-output decoding, Hard-input soft-output decoding, Early termination

1 Introduction

Turbo product codes (TPC) are decoded in general using soft-input soft-output (SISO) decoding, as introduced by Pyndiah in 1994 [1, 2], which nearly achieves the Shannon capacity limit with reasonable decoding complexity. In addition, TPC has many useful characteristics such as simple encoding/decoding method and a high degree of parallelized structure, etc. Moreover, a TPC can be flexibly designed according to the composition of component codes used, and also, it is especially advantageous in terms of decoding complexity when its code rate is high [3]. Because of these reasons, TPC has been adopted in many communication standards such as the IEEE 1901 [4], IEEE 802.20 [5], and IEEE 802.16 [6]. In addition, many studies on TPC decoding for hybrid automatic repeat and request (HARQ) systems have been conducted as well [7–12].

In general, the most widely used decoding algorithm of TPC is the Chase-Pyndiah SISO algorithm, and it is generally divided into two steps. The first is the hard decision decoding (HDD) based on the Chase-II algorithm [13], and the second is the calculation of extrinsic information. During SISO decoding, the number of HDD required is increased in proportion to the number of least reliable bits (LRBs) p; a large number of arithmetic operations are necessary to calculate the extrinsic information as well. Meanwhile, the error correction of TPC can also be accomplished via hard-input hard-output (HIHO) decoding [14–17]. HIHO is characterized by a low-complexity in comparison with SISO decoding because its decoding procedure is only based on hard information. However, since the error correction capability of HIHO decoding is significantly lower than that of SISO decoding, it is used in limited communication scenarios.

Therefore, many studies have been conducted to lower the computational complexity of SISO decoding. At first, Al-Dweik et al. [18] proposed an efficient hybrid decoding algorithm to decrease the computational complexity. This algorithm employed both SISO and HIHO decoding algorithms sequentially. The SISO decoding was used for early iterations, whereas the HIHO decoding was utilized to correct residual errors during later iterations. As a result, it was possible to reduce the complexity with almost similar error correction performance. However, the number of iterations for SISO and HIHO decoding was always fixed. Thus, the SISO decoding with very high-complexity always had to be preceded, even if the successful error correction was possible using only the HIHO decoding. Moreover, the number of iterations for both decoding algorithms had to be determined by carrying out Monte-Carlo simulations according to each TPC.

Meanwhile, [19–22] have introduced various methods to reduce the number of required test patterns during the iterative decoding procedure. For example, in [22], as the iterative decoding progresses, the number of LRBs decreased gradually. This algorithm was based on the Hamming distance results before and after performing error correction using an algebraic (or hard) decoding in the previous iteration. This technique could effectively lower the decoding complexity in proportional to the decrement of the p value. However, it was inefficient when the error correction capability of the component codes was small. Further, in estimating the variable value of p, the required scaling factor was determined through numerous simulations for each TPC.

To reduce the decoding complexity of TPC more effectively, various studies on low-complexity decoding algorithms based on the detected syndrome have been conducted [23–25]. For example, in [23, 24], the calculation of extrinsic information was performed immediately without any error correction if the detected syndrome is the all-zero vector. In other words, it was possible to obtain a valid codeword without using the Chase-II algorithm. However, if a error syndrome was detected, the error correction was performed by using the conventional SISO decoding algorithm. As a result, the computational complexity of such syndrome-based decoding algorithms was determined according to the ratio of input vectors detected as having the no-error syndromes. In addition, Ahn et al. [25] introduced a highly effective low-complexity decoding algorithm for TPC, which was based on the syndrome characteristics of extended Hamming codes. In this algorithm, it was possible to determine the valid codeword conditionally by using only a single HDD operation when the single-error syndrome was identified. Therefore, the error correction with much lower complexity was achievable, as compared with that of conventional syndrome-based decoding algorithms. However, if a double-error syndrome

was detected, the SISO decoding had to be used, and the resulting induced computational complexity was not trivial.

In this paper, we propose an advanced syndrome-based decoding algorithm for TPC that employs the extended Hamming codes as component codes. In the proposed algorithm, distinct decoding methods are adaptively applied based on the syndrome detection result of each input vector. Once a no- and single-error syndrome is detected, conventional syndrome-based decoding algorithms [24, 25] are used. However, if a double-error syndrome is detected, an advanced low-complexity hard-input soft-output (HISO) decoding is applied conditionally. In the conventional SISO decoding algorithms, 2^p number of HDD operations are needed for error correction. However, only a single HDD is required in the proposed HISO decoding method, leading to reducing the computational complexity of TPC decoding significantly. In addition, we introduce an early termination technique for undecodable blocks via predicting the decoding failure. The proposed early termination is also based on the syndromes of the input vectors, particularly the detection number of the double-error syndromes. The average number of iterations is lowered substantially by using this technique, which leads to error correction with low-complexity. As a result, using the proposed two algorithms, it is possible to decrease the computational complexity considerably compared to conventional syndrome-based decoding algorithms; at the same time, the error correction performance is almost the same as before.

The following is the composition of this paper. In Section 2, we first review the conventional Chase-Pyndiah and syndrome-based decoding algorithms. Section 3 provides details of the two proposed novel techniques for low-complexity decoding of TPC. In Section 4, we discuss the results of the computational complexity as well as the error correction performance of the proposed algorithms when compared with that of conventional algorithms [2, 24], and [25]. Finally, Section 5 is the conclusion of this paper.

2 Background

In this section, we briefly introduce three conventional TPC decoding algorithms briefly. First, the Chase-Pyndiah algorithm [2] is discussed, which is a primary SISO decoding method. Subsequently, overviews of two conventional syndrome-based decoding algorithms [24, 25] that reduce the decoding complexity of the Chase-Pyndiah algorithm, are provided. We assume that the two-dimensional TPCs are constructed by two linear block codes C^i (i=1,2). The linear block codes C^i are expressed as $(n, k, d_{\min})$, where each parameter stands for the length of the codeword, number of information bits, and the minimum Hamming distance, respectively. Therefore, if the TPC codeword is constructed by two identical component codes C^1 and C^2, the result is denoted as $(n, k, d_{\min})^2$. In addition, we also suppose that the extended Hamming codes are utilized as component codes. The extended Hamming codes substantially increase the error correction capability compared to constituting TPCs with general Hamming codes when composed of two or more dimensional TPC [26]. For instance, if a two-dimensional TPC is constructed using (7,4) Hamming codes, the minimum distance $d_{\min}$ is 3. Therefore, its error correction capability is $t = \lfloor d^2_{\min} - 1 \rfloor / 2 = \lfloor 9 - 1 \rfloor / 2 = 4$, i.e., it is possible to correct up to four errors. However, provided (8,4) extended Hamming codes are used as component codes, the error correction capability is improved considerably to $t = \lfloor 16 - 1 \rfloor / 2 = 7$ because $d_{\min}$ is 4. Furthermore, using an extended bit, it is possible to distinguish not

only a no-error and single-error syndrome but also a double-error syndrome. Here, the detection of a single-error syndrome suggests that the decoder input vector contains one or more odd numbers of errors. Thus, if there is only one error, it is possible to succeed in error correction using the extended Hamming decoder, since its error correction capability is one, i.e., $t = 1$. However, if there are three or more odd number errors, extended Hamming decoder conducts erroneous decoding. Besides, the double-error syndrome suggests that there are two or more even number errors in the decoder input vector. Therefore, when detecting a double-error syndrome, extended Hamming decoder fail to correct error; instead, it generates additional errors [17].

2.1 Chase-Pyndiah decoding algorithm

The Chase-Pyndiah algorithm [1, 2] is a basic decoding algorithm for TPC. It is also known as SISO decoding and exhibits an excellent error correction capability compared with that of HIHO decoding. The decoding procedure for the Chase-Pyndiah algorithm using the extended Hamming codes as component codes is as follows.

1) The hard-decision vector $R^H=\left(r_1^H, r_2^H, ..., r_n^H\right)$ is generated from the received soft-valued vector $R=(r_1, r_2, ..., r_n)$.

$$r_k^H = \begin{cases} 1, & \text{if } r_k \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where $k \in \{1, 2, ..., n\}$.

2) The reliability of bit component r_k is defined by $|r_k|$ [2], and the positions of p LRBs are determined according to the ascending order of the magnitude of $|r_k|$. Subsequently, the 2^p test patterns T^i are obtained by placing 0 or 1 at the locations of p LRBs and 0 in the remaining bit positions.

3) The test sequences Z_i is obtained using the modulo-2 addition between T^i and R^H.

$$Z_i = R^H \oplus T^i \tag{2}$$

where $i \in \{1, 2, ..., 2^p\}$.

4) The equation

$$S_i = Z_i \cdot H^T \tag{3}$$

is employed to calculate the syndrome S_i, where $i \in \{1, 2, \ldots, 2^p\}$, and H^T represents the transposed parity check matrix of the component code used. Subsequently, the HDD, i.e., the Hamming decoding, which conducts error correction based on the syndrome, is performed to generate the valid codeword $C^i = (c_1^i, c_2^i, \ldots, c_{n-1}^i)$. After that, the equation

$$c_n^i = \sum_{k=1}^{n-1} c_k^i (\text{mod} 2) \tag{4}$$

is used to calculate an extended bit c_n^i, where c_k^i is the kth element of C^i.

5) In order to calculate the squared Euclidean distance (SED) between R and C^i, the equation

$$\|R - C^i\|^2 = \sum_{k=1}^{n} [\, r_k - (2c_k^i - 1)]^2 \tag{5}$$

is used, where $i \in \{1, 2, ..., 2^p\}$.

6) The maximum likelihood (ML) codeword D, which has the minimum Euclidean distance among the 2^p candidate codewords, is determined by using the equation

$$D = arg\ min_{C^i,\ i\in\{1,2,\ldots,2^p\}} \|R - C^i\|^2 \tag{6}$$

where $i \in \{1, 2, ..., 2^p\}$.

7) In the presence of competing codewords related to the kth bit position, the equation

$$w_k = \frac{\|R - C^{j(k)}\|^2 - \|R - D\|^2}{4}(2d_k - 1) - r_k \tag{7}$$

is used to calculate the extrinsic information, where d_k is the kth element of decision codeword D, and $C^{j(k)}$ is the competing codeword with a minimum SED among the candidate codewords carrying a value different from the kth element of D. Otherwise, the equation

$$w_k = \beta \times (2d_k - 1) \tag{8}$$

is used to compute the extrinsic information. The reliability factor β represented in [2] is used when $C^{j(k)}$ does not exist.

2.2 Syndrome-based decoding algorithms

The Chase-Pyndiah algorithm always employs SISO decoding for all input vectors of the decoder, resulting in a severe increase in complexity. Therefore, a variety of studies have been conducted to lower the decoding complexity of TPCs, which are based on the syndrome. At first, in [24], the SISO decoding algorithm was used if the syndrome of the input vector represented non-zero. Conversely, when the detection result of the syndrome was error-free, the hard decision vector R^H was directly deemed to be the decision codeword D, and extrinsic information was calculated based on this result. Through this, the computational complexity of TPC decoding was reduced in proportion to the number of the no-error syndrome detection. This low-complexity decoding algorithm is generally referred to as HISO decoding.

In the recently proposed syndrome-based decoding algorithm [25], the HISO decoding was applied conditionally, not only in the case of the no-error but also the single-error syndrome detection. In this algorithm, HDD was performed as the first step after detection of a single-error syndrome. Subsequently, the SED between a valid codeword obtained from HDD and R^H was calculated, after which it was verified to be a minimum or not. This process distinguished whether a single error or more than three errors occurred in an input vector. If HDD was applied to an input vector, which contained only a single error, it was possible to succeed in error correction with a high probability. In other cases, however, a non-optimal codeword was generated by HDD, leading to substantial performance degradation. Therefore, in such cases, conventional SISO decoding had to be employed. If the valid codeword obtained from HDD was confirmed to have a minimum SED, it was determined to be a decision codeword D, and extrinsic information was computed accordingly. In [25], this syndrome-based decoding algorithm was named as HDD-based HISO decoding, which was possible to further lower the computational complexity compared with the previous algorithms.

3 Proposed syndrome-based decoding algorithm

In this section, we propose two novel schemes for low-complexity decoding of TPCs, to further decrease the computational complexity in comparison with that of conventional syndrome-based decoding algorithms. The first is about the advanced HISO decoding algorithm. Figure 1 is a block diagram illustrating the main concept of the proposed HISO decoding. We can verify that the appropriate decoding algorithm is applied adaptively according to the syndrome detection result from each decoder input vector. If a no-error syndrome is detected, HISO decoding is carried out, and a reliability factor δ_1 is used to calculate extrinsic information (as in reference [24]). Furthermore, if a single-error syndrome is detected, error correction is performed via HDD-based HISO or SISO decoding. A reliability factor δ_2 is used to calculate extrinsic information when applying HDD-based HISO decoding (as in reference [25]). However, as we noted, all conventional syndrome-based decoding algorithms have always employed SISO decoding for error correction when detecting a double-error syndrome, as expressed in the red box in Fig. 1. Thus, it can cause a considerable increase in decoding complexity. Moreover, if the HDD-based HISO decoding introduced by Ahn et al. [25] is directly applied to an input vector with two or more errors, it severely increases the risk of error correction performance degradation. Hence, we first propose an advanced HISO decoding algorithm that can be used to decode input vectors with double-errors, as shown in the blue box on the right side of Fig. 1. In this algorithm, a minimum number of test patterns are utilized to overcome the limitations of the error correction capability of component codes. Moreover, just a single HDD operation is needed for successful decoding. As a result, the first scheme further diminishes the applicability of SISO decoding as compared with that in the earlier algorithms, and accordingly, it decreases the overall computational complexity more.

Meanwhile, the computational complexity of TPCs is dependent on the required number of iterations as well. Chen et al. [27] developed an early stopping algorithm for TPC decoding. In this technique, if all the Chase decoder outputs in the directions of the row and column are identified as valid codewords, the iterative decoding procedure is completed early. The reason for this is that it can be considered that there are no more errors in the TPC codeword. However, this method is valid when the signal-to-noise ratio (SNR)

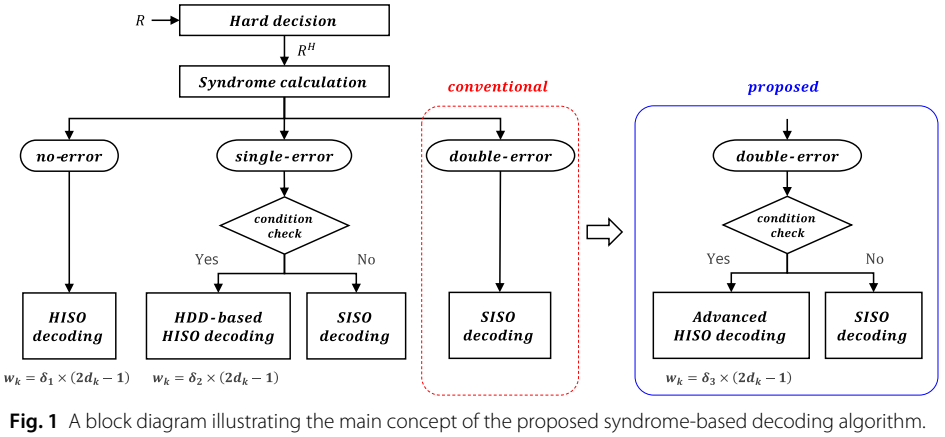

Fig. 1 A block diagram illustrating the main concept of the proposed syndrome-based decoding algorithm. The red box represents the conventional decoding technique (SISO decoding) when the double-error syndrome is detected, and we want to modify it like the blue box (a combination of advanced HISO and SISO decoding)

is high. In other words, it is only applicable to decodable blocks whose error correction is successful before reaching the predefined maximum number of iterations. Conversely, due to the randomness in the communication channel, there are some events where the error correction fails even if the iterative decoding is performed sufficiently. In this scenario, the problems with high power consumption and excessive computational time can be induced. If successful error correction is unlikely, it is more reasonable to increase the system efficiency of the TPC decoder by terminating the iterative decoding early and requesting retransmission to the transmitter. Therefore, in this paper, we introduce the early termination algorithm as the second scheme. This technique continuously evaluates the possibility of error correction failure during the iterative decoding process. When decoding failure is anticipated, the error correction procedure is terminated early, thereby effectively reducing the decoding complexity. To the best of our knowledge, a number of early termination techniques for other error correction codes such as low-density parity check (LDPC) or turbo codes have been introduced [28–35]. However, there have been no studies of early termination techniques for TPCs.

3.1 Scheme 1: Advanced HISO decoding algorithm applicable to the input vector containing two errors

Figure 2 represents the ratios of input vectors detected as having the total error, single-error, and double-error syndromes for four TPCs as the SNR increases. This result was obtained using binary phase shift keying (BPSK) modulation in additive white Gaussian

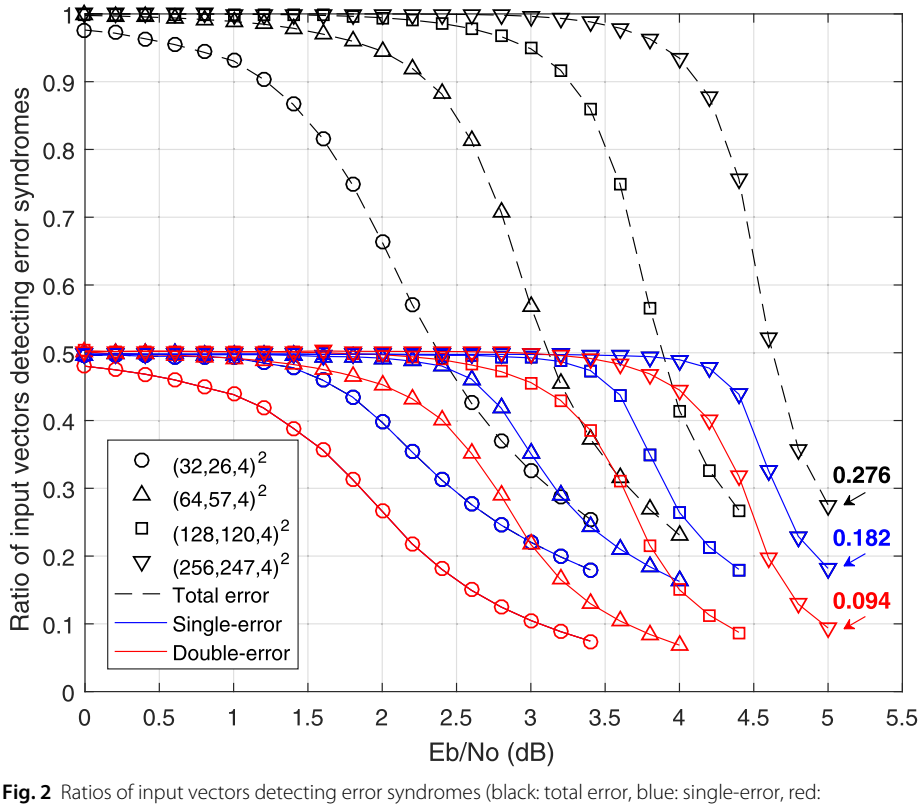

Fig. 2 Ratios of input vectors detecting error syndromes (black: total error, blue: single-error, red: double-error) according to the SNR increment during the TPC decoding based on the Chase-Pyndiah algorithm. We assumed that both the number of maximum iteration and LRBs as 4

noise (AWGN) channel based on the Chase-Pyndiah decoding algorithm. The number of LRBs and the iterations was set to four in common. Each ratio of input vectors detected as having the single- or double-error syndromes at each SNR region is obtained by dividing the number of all input vectors whose syndromes are detected as single- or double-error into the number of entire decoder input vectors during TPC decoding, respectively. Similarly, the ratio of input vectors detecting total error syndromes at each SNR is calculated by dividing the number of all erroneous input vectors into the number of entire input vectors during TPC decoding. The simulation result shows that the ratios of input vectors detecting error syndromes tend to decrease gradually with increasing SNR in all TPCs. Although the ratio of input vectors detected as having double-error syndromes is small compared to that of the input vectors having the single-error syndromes, it is not trivial. We can verify that the proportion of double-error syndrome detection accounts for about a third of the total error syndromes in high SNR regions regardless of TPCs. For example, in the case of $(256, 247, 4)^2$ TPC in Fig. 2, the ratios of input vectors detecting all erroneous or double error syndromes are about 0.276 and 0.094 at SNR 5.0 dB, respectively. Thus, the number of input vectors with double-error syndrome among all erroneous input vectors account for about 34.1%. In conventional syndrome-based decoding algorithms, which use the extended Hamming code as a component code, there was no choice but to employ the Chase-Pyndiah algorithm to the input vector having more than two errors for error correction. Hence, based on the result of Fig. 2, it can be predicted that the decoding complexity increases significantly depending on the number of input vectors whose syndrome detection results are double-error.

Assuming the TPC codeword was transmitted over the AWGN channel, the noise contained in the received codeword has zero-mean; it is independent and identically distributed Gaussian random variable with variance $N_0/2$. Thus, if a double-error syndrome is identified, an event where two errors are included in the decoder input vectors is more frequent than that having four or more errors. In this case, assuming the number of LRBs is p, it is highly likely to succeed in error correction by using only p test patterns that have a single 1 in each pattern. Therefore, using 2^p test patterns consistently, as is done in the conventional decoding algorithms, is quite wasteful in terms of computational complexity. To overcome this disadvantage, many studies have been conducted to reduce the decoding complexity by sequentially decreasing the magnitude of p as the number of iterations increases [19–22]. However, unless the value of p is lowered to zero, its computational complexity is high compared to that of HISO or HDD-based HISO decoding, which conceptually uses only a single test pattern. Therefore, in the HISO decoding algorithm proposed in this study, we first distinguish whether two or more errors exist in the input vector when detecting double-error syndrome. In the former case, it is able to succeed in error correction via only a single test pattern and HDD with a high probability.

Figure 3 presents a block diagram associated with the advanced HISO decoding algorithm we invented. In this algorithm, the advanced HISO or SISO decoding is applied according to the predefined condition through a series of processes. The detailed decoding procedure of this is as follows.

1) The hard decision vector $R^H = \left(r_1^H, r_2^H, ..., r_n^H\right)$ is generated from $R = (r_1, r_2, ..., r_n)$ using Eq. (1).
2) The syndrome S_{de} (double-error syndrome) of hard decision vector R^H is calculated based on Eq. (3).

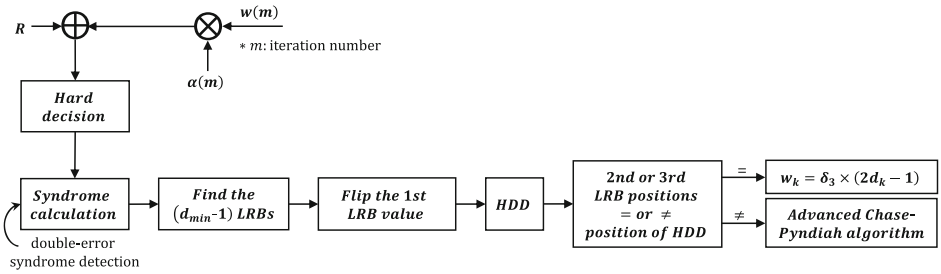

Fig. 3 A block diagram illustrating the detailed procedure of the proposed HISO decoding algorithm when detecting the double-error syndrome

3) The $(d_{\min} - 1)$ LRB positions are determined.
4) A test sequence Z_{de} is obtained by flipping a binary bit value in the first LRB position (i.e., 0→1 or 1→0).
5) The HDD is applied to Z_{de}, and a value of extended bit is calculated using Eq. (4).
6) It is determined whether the bit position corrected by HDD in Z_{de} corresponds to one of the second or third LRB positions.
7) If the positions are matched, the codeword obtained by HDD is determined to be the decision vector $D = \{d_1, d_2, \ldots, d_n\}$; then, the equation

$$w_k = \delta_3 \times (2d_k - 1). \tag{9}$$

is used to calculate the extrinsic information, where δ_3 is a reliability factor used when applying the proposed HISO decoding algorithm.
8) Otherwise, after determining the $p - (d_{\min} - 1)$ LRB positions additionally, the SISO decoding algorithm is applied.

In the proposed HISO decoding algorithm, firstly, the hard decision vector R^H is determined from the received decoding block, and the syndrome is calculated accordingly. The syndrome detected at this time is assumed to be a double-error syndrome. After that, LRB positions are found. However, different from the conventional SISO decoding algorithm, here, only $(d_{\min} - 1)$ LRB positions are required. The reason for this is that these LRB positions are essential in determining the applicability of the proposed HISO decoding.

Subsequently, in the fourth step of Fig. 3, a value of the first LRB position in the hard decision vector R^H is flipped, i.e., it is changed to 1 if the bit value is 0, and it is reversed to 0 in the opposite case. Therefore, applying bit-flipping is the same as using a single test pattern containing bit 1 in the first LRB position only. If there are two errors in the input vector and one of them is at the bit position where the bit-flipping was applied, then the error correction can be successful through HDD in step 5. For this reason, Z_{de} can be considered to be the test sequence. However, we cannot determine whether only two errors exist in the input vector when detecting double-error syndrome. Even if there really are only two errors, we also cannot guarantee that one of the errors is in the first LRB position. If the proposed HISO decoding is unconditionally applied to all input vectors with a double-error syndrome, it is likely to result in severe performance loss. Therefore, after step 5, it is necessary to verify whether the result of HDD is an optimal codeword. In other words, we have to confirm whether the valid codeword obtained through HDD is

the same as the maximum likelihood (ML) codeword that can be acquired by the Chase-Pyndiah algorithm.

In step 6, we identify whether the bit position corrected by HDD matches with one of the second or third LRB positions. If this condition is satisfied, we can ensure that the error correction was performed correctly. Therefore, the proposed HISO decoding can be an alternative to conventional SISO decoding, which we explain as follows.

Let the valid codeword obtained using the proposed HISO decoding be designated as D_{HISO}. In order to prove the validity of the proposed algorithm, we first calculate the difference of the SED between D_{HISO} and R^H from the received codeword R by using the following equation:

$$\|R - D_{\mathrm{HISO}}\|^2 - \|R - R^H\|^2 = 4(|r_1| + |r_{hdd}|) \tag{10}$$

where $|r_1|$ indicates the reliability of the first LRB position, and similarly, $|r_{hdd}|$ represents the reliability of an error bit modified by HDD. The reason for the result of the Eq. (10) is that only the bit values of the first LRB and *hdd*-th bit position are different between D_{HISO} and R^H.

Next, any valid codeword in the Euclidean distance closest to D_{HISO} is a modification of bit values in two LRBs from R^H except for the first LRB and the *hdd*-th bit position. Let this valid codeword be referred to as D_{CMP}. The reason for this is that the minimum distance of the component code used in this study, that is, that of the extended Hamming code is four. In other words, at least four bits have to be different between D_{HISO} and other valid codewords. Thus, the difference of the SED between D_{CMP} and R^H from the received codeword R is expressed using the equation:

$$\|R - D_{\mathrm{CMP}}\|^2 - \|R - R^H\|^2 = \min_{i,j\in\{2,\ldots,n\}\setminus hdd} 4(|r_i| + |r_j|). \tag{11}$$

If the proposed HISO decoding performed error correction correctly, it means that D_{HISO} is identical to the ML codeword obtained by the Chase-Pyndiah decoder. Thus, the difference in SED between D_{HISO} and R^H has to be smaller than any other case. Therefore, the result of Eq. (10) has to be equal with or smaller than the result of Eq. (11), this relationship is expressed as follows:

$$|r_1| + |r_{hdd}| \leq \min_{i,j\in\{2,\ldots,n\}\setminus hdd} (|r_i| + |r_j|). \tag{12}$$

If step 6 in the proposed algorithm is satisfied, the condition in Eq. (12) is always met. We explain this in more detail with the following examples.

Figure 4 shows simple examples illustrating the application of the proposed decoding algorithm when detecting double-error syndrome. We assume that $(8, 4, 4)^2$ TPC was transmitted, and the original message bits were all zero. Two errors exist in each example (*a*), (*b*), (*c*), and (*d*) (i.e., two 1s marked with red). Also, the reliability sequence of received symbols from the AWGN channel supposes as $|R| = \{0.13, 0.28, \ldots, 0.86\}$ for the examples (*a*), (*b*), and (*c*). Therefore, assuming the number of LRBs is 4, the first four-bit positions belong to this range. Besides, in the case of the example (*d*), the reliability sequence supposes as $|R| = \{0.81, 2.87, \ldots, 0.86\}$, and the p LRBs are discontinuously located, as expressed in Fig. 4.

Example (*a*) represents a case where the error correction is performed correctly through the proposed HISO decoding; that is, the value of the SED between D_{HISO} and R^H is a minimum. In this case, two errors exist in the first and second LRB positions.

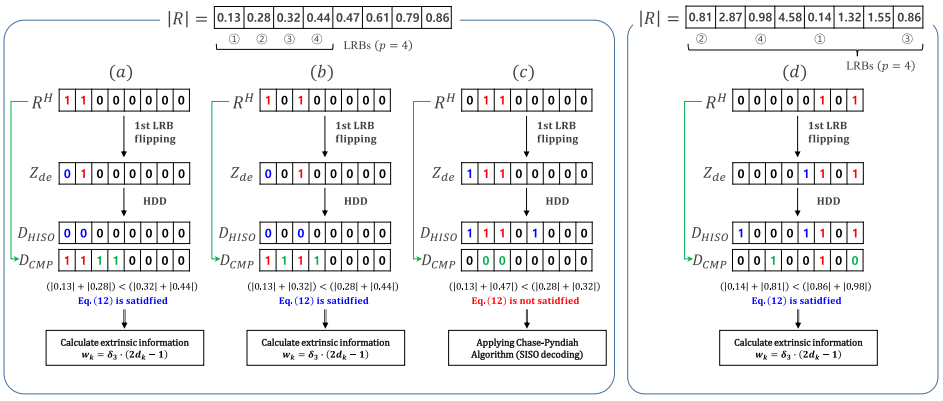

Fig. 4 Examples performing the proposed HISO decoding algorithm when containing two errors in the input vector. Example (*a*) and (*b*) represent the cases of successful error correction via the proposed HISO decoding. By contrast, example (*c*) illustrates the case of wrong error correction. Example (*d*) shows the valid codeword is generated, but it is not the correct codeword

Thus, Eq. (12) is expressed as $(|r_1| + |r_{hdd=2}|) = (|0.13| + |0.28|) \leq (|r_3| + |r_4|) = (|0.32| + |0.44|)$, and this inequality is true. Similarly, in the case of (*b*), there are two errors in the first and third LRB positions, and Eq. (12) is also satisfied since it is represented as $(|r_1| + |r_{hdd=3}|) = (|0.13| + |0.32|) \leq (|r_2| + |r_4|) = (|0.28| + |0.44|)$. Conversely, in the case of (*c*), there are two errors in the second and third LRB positions. Thus, Eq. (12) is described as $(|r_1| + |r_{hdd=5}|) = (|0.13| + |0.47|) \leq (|r_2| + |r_3|) = (|0.28| + |0.32|)$; however, this inequality is not true. In addition, we can confirm that D_{HISO} has incorrectly performed error correction. Instead, D_{CMP} is the correct codeword, which can be obtained by using the Chase-Pyndiah algorithm.

Meanwhile, we can think of the possibility where the proposed HISO decoding makes wrong error correction. In the case of (*d*), the valid codeword D_{HISO} satisfies Eq. (12), but it is not the correct codeword. However, it is clear that the valid codeword obtained by the proposed algorithm is the same valid codeword generated by the Chase-Pyndiah algorithm. This is caused by D_{HISO} being located at an Euclidean distance closer to R^H than any other valid codewords. It is not possible to produce the codeword having a shorter SED from R^H than that of D_{HISO}, even with the Chase decoder. Therefore, the proposed technique can result in the wrong error correction, through it is not a direct cause of performance loss. However, Figs. 11 and 12 show that there is slight performance degradation when applying the proposed algorithm. This degradation is due to the method for calculating extrinsic information, which we will further explain in the bit error rate (BER) performance analysis section.

In fact, the better the status of the communication channel, and the more the iterative decoding progresses, the more likely it is that there are only two errors in the input vector where the double-error syndrome is detected. Furthermore, under the same conditions, the two errors in the input vector are very likely to belong to the low-order LRB positions due to the characteristics of the Chase-Pyndiah algorithm [1, 2]. Therefore, the valid applicability of the proposed HISO decoding also increases.

We have demonstrated that successful error correction with low-complexity is conditionally possible using the proposed HISO decoding algorithm if the input vector involves two errors. Applying bit-flipping to the first LRB is to overcome the limitations of the

error correction capability of the component code decoder, but there is an additional reason. As mentioned before, the proposed HISO decoding is not meant for correcting all kinds of two errors that could possibly be generated in the received decoding block. It is a valid technique when there are two errors in the first and second LRB positions or the first and third LRB positions. In the cases of those error patterns, the first LRB position is included in common. The reason is that the reliability of the first LRB is the smallest, that is, it is most likely that an error occurred. Therefore, it is possible to correct these two types of error patterns by using only a single HDD after applying bit-flipping. Considering the overlapped bit position between the two error patterns, it is reasonable to reverse only the bit value of the first LRB. Since the proposed syndrome-based HISO decoding technique performs error correction based on the bit-flipping and HDD operation, we designate the first proposed scheme as the BFHDD-HISO decoding algorithm.

Finally, if the error correction is achieved via the BFHDD-HISO algorithm, extrinsic information is calculated based on the equation shown in step 7. In this case, Eq. (7) of the Chase-Pyndiah decoding algorithm is not applicable due to the absence of competing codewords related to D_{HISO}. Instead, Eq. (9) is utilized. In this formula, a new reliability factor δ_3 is employed, not the β used in conventional decoding algorithms. This value is determined through the Monte-Carlo simulation to maximize error correction performance. Therefore, if the error correction is conducted through the proposed BFHDD-HISO decoding, we can verify that the computational complexity required for the calculation of extrinsic information is low as well.

3.2 Scheme 2: The early termination algorithm

In general, decoding blocks are classified into decodable and undecodable types based on the randomness in communication channels. In particular, undecodable blocks suggest a very high probability of failure to correct errors. In other words, it is meaningless to perform the error correction as much as the predefined maximum number of iterations. In the previous subsection 3.1, we propose the BFHDD-HISO decoding algorithm for low-complexity error correction. This technique lowers the computational complexity by lessening the number of demanded HDD operations as compared to that of conventional algorithms in which the SISO decoding should be used when detecting double-error syndromes. However, the decoding complexity of TPCs also depends on the number of iterations executed. Therefore, in this subsection, we propose an early termination algorithm that finishes the iterative decoding procedure earlier if the received codeword is undecodable.

Figure 5 shows the ratios of input vectors detected as having each syndrome according to SNR regions when applying the Chase-Pyndiah decoding algorithm to the $(64, 57, 4)^2$ TPC of the IEEE 802.16 [6] standard. During the simulation of each SNR region, both the number of LRBs and the maximum iterations were set to four in common. We also assumed that the binary random bit sequences modulated by BPSK symbols are transmitted over the AWGN channel with zero mean and $N_0/2$ noise variance. To obtain reliable results, we set that at least 1000 codeword errors are detected in each SNR region. Each of the sections (a) to (d) in Fig. 5 represents the ratios of input vectors detecting the no-, single- and double-error syndromes at the SNR of 1.0 dB, 2.75 dB, 3.25 dB, and 3.5 dB. Each SNR value refers to the region corresponding to the BER 10^{-1}, 10^{-2}, 10^{-4}, and 10^{-6}. The horizontal axis of each graph indicates the half-iteration number, with a maximum set

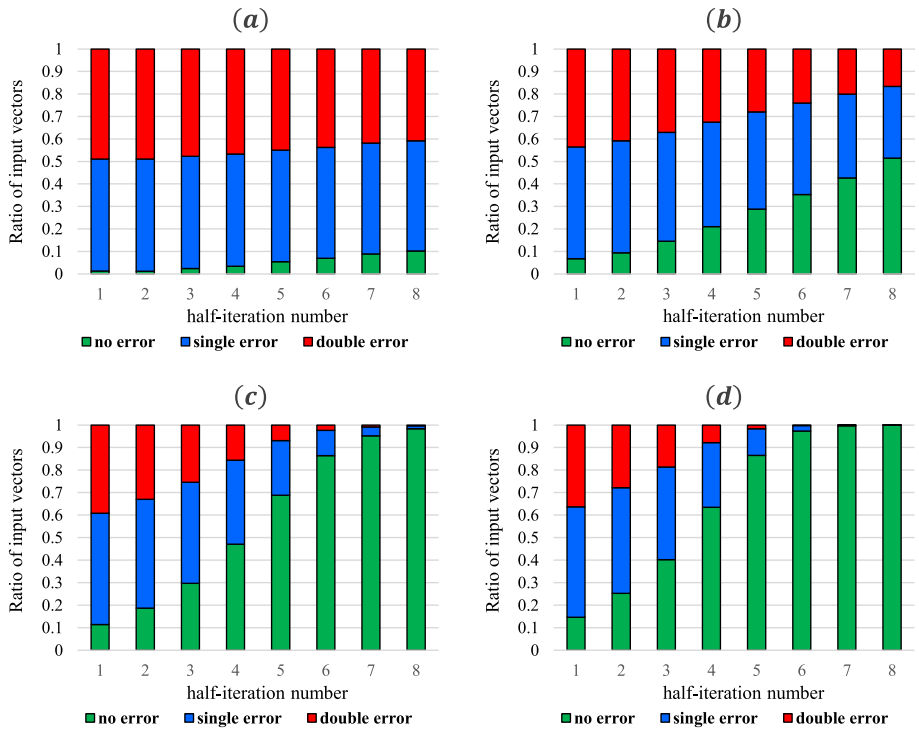

Fig. 5 Ratios of input vectors detecting each syndrome (green: no-error, blue: single-error, red: double-error) in the $(64, 57, 4)^2$ TPC according to the SNR regions when error correction is performed based on the Chase-Pyndiah algorithm. **a** SNR 1.0 dB corresponding to BER 10^{-1}. **b** SNR 2.75 dB corresponding to BER 10^{-2}. **c** SNR 3.25 dB corresponding to BER 10^{-4}. **d** SNR 3.5 dB corresponding to BER 10^{-6}

of eight, suggesting maximum times of four full iterations. Each syndrome detection ratio is obtained by dividing the total number of each syndrome detected at each half-iteration by the number of decoder input vectors n in the row (or column) direction.

In all examples, the ratios of input vectors detected as having single- and double-error syndromes in the first half-iteration account for most of the total syndromes. However, as the iterative decoding progresses, the ratio of input vectors detecting the no-error syndromes increases gradually regardless of the SNR, and that of the remaining two syndromes tends to decrease. If the channel environment is good (i.e., cases (*c*) and (*d*)), the ratios of input vectors detecting the single- and double-error syndromes are sharply reduced when the number of half-iterations is increased. In other words, the ratio of input vectors detected as having no-error syndromes increases drastically and converges to 1 with a very high probability before finishing the iterative decoding. Conversely, when the channel state is inferior (i.e., cases (*a*) and (*b*)), the error syndromes still account for a significant proportion at the last half-iteration, even though iterative decoding is carried out sufficiently. It implies that the error correction will fail. Therefore, we can find that changes in the ratios of input vectors detecting each syndrome are varied depending on the channel condition. In other words, it is possible to predict the error correction failure based on the syndrome detected during TPC decoding.

The early termination technique proposed in this paper is based on the syndrome detected in each half-iteration. Notably, we use the number of double-error syndrome detections, which generally contain the more large number of errors and also exhibit

a more significant negative effect on the error correction. The detailed process of the proposed early termination algorithm is as follows.

1) Both tolerance counter S and double-error syndrome counter λ_i are initialized to 0, and the half-iteration counter i is initialized to 1. In addition, the maximum tolerance number $S_{\max}$ and maximum half-iteration number $I_{\max}$ are set.
2) If i=1, λ_1 is increased by one each time a double-error syndrome is detected. Substantially, the decreasing ratio υ is calculated using the following equation.

$$\upsilon = \frac{P_1}{I_{\max}}, \quad \text{where } P_i = \frac{\lambda_i}{n}. \tag{13}$$

3) If $i > 1$, i.e., in the mth half iteration, the counter λ_m and the detection ratio P_m are determined. Based on this, the tolerance counter S is increased by one if the inequality

$$P_{m-1} - P_m < \upsilon \tag{14}$$

is satisfied.
4) If S is equal to $S_{\max}$, the iterative decoding process is terminated. Otherwise, it starts with step 3, and the same procedures are repeated until the counter i approaches $I_{\max}$.

The proposed early termination algorithm comprises two stages: the initialization and the condition check for the application of the proposed early termination. The initialization includes the first and second steps of the proposed algorithm. In step 1, before starting error correction, we initialize various counting parameters, and also set the threshold $S_{\max}$ and the maximum half-iteration number $I_{\max}$. Subsequently, in step 2, after completing the error correction in the first half-iteration, the decreasing ratio υ of the double-error syndrome is calculated by using Eq. (13). Here, P_1 indicates the ratio of input vectors detected as having the double-error syndromes in the first half-iteration. It is the value of the total number of double-error syndromes λ_i detected in the first half-iteration divided by the total number of input vectors n. υ denotes the value of P_1 divided by the maximum half-iteration number $I_{\max}$. It implies an expected reduction rate when assuming that the number of double-error syndromes decreases linearly during the TPC decoding. In other words, if the ratio of input vectors detecting double-error syndromes diminishes as much as υ for each half-iteration, it can be predicted that P_i eventually converges to zero at the end of the iterative decoding. It suggests a high probability of successful error correction.

The remaining steps 3 and 4 include the condition evaluation procedures for applying the proposed early termination. We first calculate the detection ratio P_m in any mth half-iteration. Subsequently, the difference in the ratios of input vectors detecting double-error syndromes between two consecutive half-iterations is obtained, and this result is compared with υ. If the result meets Eq. (14), it increases the tolerance counter S by one. What satisfying Eq. (14) indicates that the actual reduction ratio of double-error syndrome detection is smaller than the expected reduction ratio. In other words, since the decoding convergence of error correction is slow, it may suggest a risk of failure to correct errors. However, in the opposite case, the value of S is maintained because it can be intuitively expected that the error correction will be successful with a high probability before the iterative decoding is completed. In fact, the error correction alternates in the

directions of the row and column during the decoding of two-dimensional TPCs, such that the ratios of input vectors detecting erroneous syndromes can be decreased nonlinearly. In other words, the number of double-error syndrome detection declines slowly in early half-iterations but possibly diminished rapidly in subsequent half-iterations, leading to successful error correction. Therefore, we intend to prevent the wrong application of early termination using the tolerant counter S, i.e., the early termination is applicable only when the value of S steadily increases and reaches to $S_{\max}$.

For example, in example (d) of Fig. 5, i.e., SNR region of 3.5 dB, the initial ratio of input vectors detecting double-error syndromes is about 0.36, and the value of υ is estimated to be 0.045. The ratios of input vectors detected as having double-error syndromes in subsequent half-iterations are about 0.28, 0.19, 0.08, and so on. Therefore, all the differences in the ratios of input vectors detecting double-error syndromes between two consecutive half-iterations are greater than υ. We can find that it is possible to succeed in error correction before i reaches the maximum number of iterations. Conversely, in example (b) of Fig. 5, i.e., the SNR region of 2.75 dB, the initial ratio of input vectors detecting double error syndromes and the expected decreasing ratio are about 0.44 and 0.055, respectively; also, the ratios detecting double-error syndromes in subsequent half-iterations are about 0.41, 0.37, 0.33, 0.28, and so on. Although the decreasing ratio tends to decline to a certain extent, its degree is not as enough as expected. That is, all differences in the ratios of input vectors detecting double-error syndromes between two consecutive half-iterations are smaller than υ. Therefore, the proposed early termination algorithm enables us to estimate the decoding convergence of TPC; at the same time, it is possible to predict the possibility of error correction failure.

The value of $S_{\max}$ in step 4 can be varied depending on the code parameters such as the length of the component code used, so it has to be appropriately selected for each TPC. $S_{\max}$ can be obtained via simulations and is determined within a range that reduces the number of iterations as much as possible without degrading the error correction performance. As a result, the proposed early termination technique reduces the computational complexity of TPC decoding by not doing the unnecessary iterative decoding process.

4 Results and discussion

In this section, we analyze the proposed algorithm in terms of the decoding complexity and error correction performance. First, we examine the reduction in the average half-iteration number (AHIN) I_{ave} obtained using the proposed early termination algorithm. We can confirm that there is a trade-off between the error correction capability and the complexity reduction depending on the selected threshold value $S_{\max}$. Second, we verify how much SISO decoding and HDD usage is lowered compared with that used in conventional algorithms when the proposed two schemes are applied. In this process, the relative complexity serves as the indicator for the comparison, which is useful in identifying the reduction in computational complexity from a macroscopic perspective. In addition, for more accurate analysis, we also investigate the number of arithmetic operators utilized in the decoding of the proposed algorithm in comparison with that of conventional approaches. Finally, we provide simulation results related to error correction performance and confirm that it is almost the same as before.

During all simulations and analyses, we used TPCs as specified in the IEEE 802.16 standard [6], utilizing the extended Hamming codes as component codes, including

$(32, 26, 4)^2$, $(64, 57, 4)^2$, $(128, 120, 4)^2$, and $(256, 247, 4)^2$. We assumed that the number of LRBs and maximum iterations are both four. The weighting and reliability factors for the m-th half-iteration, which are expressed as $\alpha(m) = \{0.2, 0.3, 0.5, 0.7, 0.9, 1.0, \cdots\}$ and $\beta(m) = \{0.2, 0.4, 0.6, 0.8, 1.0, \cdots\}$, respectively, are used as they are in the conventional algorithms. Additional reliability factors δ_1, δ_2, and δ_3 (see Fig. 1) are determined to be 2.0, 1.0, and 0.5, via heuristic simulations. During simulations, the search step for finding the reliability factor was set to 0.05. The reason for this is that the range of the reliability parameter is set between 0.0 and 1.0 in general [1, 2]. The performances of the proposed and conventional TPC decoding algorithms are compared under BPSK modulation over the AWGN channel with a mean and power spectral density are zero and $N_0/2$, respectively, and codewords with random bits are utilized. In addition, the simulations were performed with code written in the C language environment on a personal computer with an Intel® Core™ i7-4790 3.6 GHz processor and 16G RAM. Furthermore, all figures related to the performance results were drawn using MATLAB.

In this study, performance analysis is conducted compared with the Chase-Pyndiah [2] and two conventional syndrome-based decoding algorithms [24, 25]. Here, the TPC decoding algorithms proposed in [24] and [25] are referred to as the syndrome-based decoding algorithm I (SBDA-I) and the syndrome-based decoding algorithm II (SBDA-II) respectively in the rest of this paper.

4.1 Reduction of average half-iteration number through early termination

Figure 6 shows the results of AHIN before and after applying the proposed early termination technique to $(64, 57, 4)^2$ TPC based on the conventional Chase-Pyndiah algorithm. In the case of using the early termination, the threshold value $S_{\max}$ was variously set from 2 to 4. We can identify that the magnitude of AHIN is effectively decreased regardless of the value of $S_{\max}$ in the low and middle SNR regions. Because any received decoding block is likely to be determined as the undecodable in these SNR regions, so the early termination is employed more actively. Meanwhile, as the SNR rises, the value of AHIN gradually increases, and eventually, its magnitude gets consistent with that of the conventional SISO decoding algorithm. The reason for this is that the better the channel environment, the more likely the received codeword is assumed to be a decodable block, which also increases the chances for successful error correction. In addition, the smaller the value of $S_{\max}$, the more the AHIN reduction is maximized, and the overall decoding complexity can also be expected to decrease accordingly. However, if the $S_{\max}$ is set too small, there is a risk of causing severe performance loss.

Figure 7 shows BER performance results depending on the value of $S_{\max}$ under the same simulation conditions as Fig. 6. This result indicates that if the magnitude of $S_{\max}$ is not large enough, the error correction capability can be greatly degraded. For example, if $S_{\max}$ is set to 2, the AHIN is diminished to about 3.25. In other words, the decoding complexity can be lowered up to nearly 40% in comparison with that of the Chase-Pyndiah algorithm. However, in this case, performance degradation of over 0.5 dB occurs at BER 10^{-6} because it is very likely that the proposed early termination technique is incorrectly applied to the decodable blocks. As mentioned above, even if the decoding convergence speed was somewhat slower in the early iterations, it can be accelerated as the iterative decoding progresses, leading to success in decoding eventually. However, if the early termination is applied too early, there will remain many error bits that are correctable with

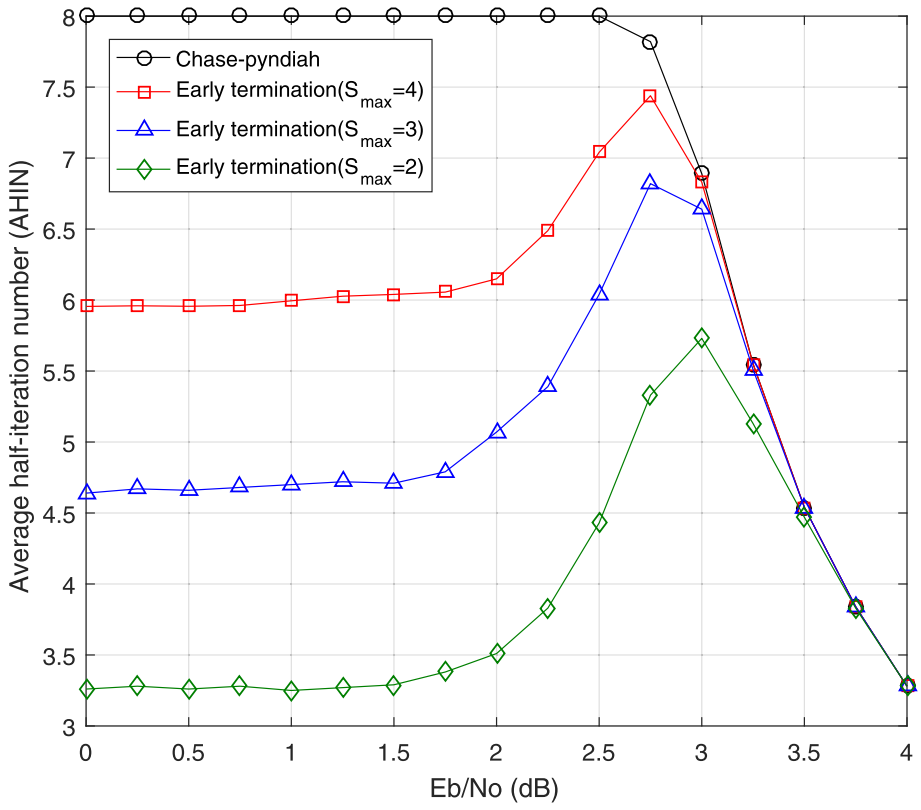

Fig. 6 Average half-iteration number before and after application of the proposed early termination algorithm for the $(64, 57, 4)^2$ TPC. $I_{\max}$, 8 (i.e., the number of maximum full-iterations, 4), LRB number, 4

high probability. Therefore, it is desirable to determine $S_{\max}$ for each TPC to be as large as possible without degrading the error correction performance through the simulations. For instance, if $S_{\max}$ is set to four for $(64, 57, 4)^2$ TPC, the error correction performance of the proposed algorithm is entirely consistent with that of the SISO decoding algorithm. At the same time, we can obtain at most 25% of the reduction effect in AHIN in low and moderate SNR regions.

4.2 Relative complexity reduction analysis based on SISO decoding and HDD operation

In this subsection, we discuss the reduction of the computational complexity that can be obtained by applying the proposed syndrome-based decoding algorithm compared to the conventional TPC decoding algorithms. Tables 1 and 2 show the usage number of the HDD and SISO decoding required in the conventional and the proposed syndrome-based decoding algorithms. Here, the subscript j of S_j and H_j signifies the type of decoding algorithm (i.e., 1: SBDA-I [24], 2: SBDA-II [25], and 3: proposed). To better understand the formulas related to the relative complexity of HDD and SISO decoding, the parameters used are listed in Table 3. The equations concerned with the relative complexity of SISO decoding are defined as (the applicability of SISO decoding to an input vector)$\times$(the number of total input vector n)$\times$(AHIN), which are expressed in Table 1. In addition, the formulas provided in Table 2 indicate the relative HDD complexity of each decoding algorithm, in which the number of test patterns used in each HISO and SISO decoding

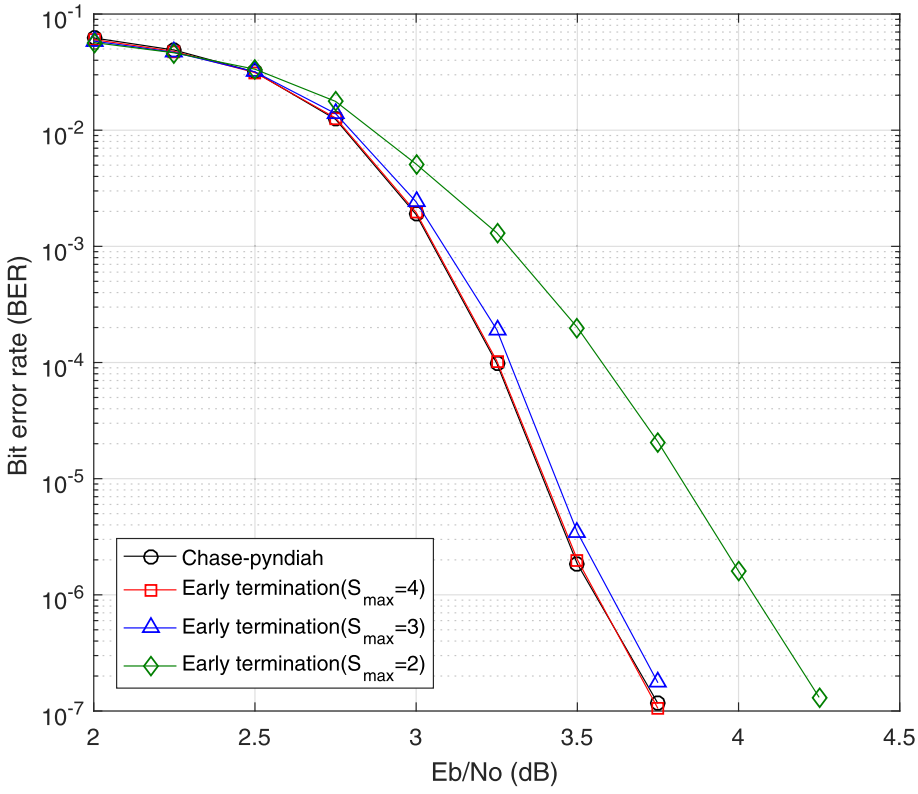

Fig. 7 Bit error rate performance results before and after applying the proposed early termination algorithm to $(64, 57, 4)^2$ TPC. l_{max}, 8 (i.e., the number of maximum full-iterations, 4), LRB number, 4

algorithm is reflected as the weighting factor. For example, in the case of the SBDA-I, the value of the weight is 2^p when employing SISO decoding. On the other hand, in the case of the SBDA-II, the number of required HDD operations has reduced by half, considering the events of generating the same candidate codewords when SISO decoding is applied. Therefore, the weight is $2^{(p-1)}$, and we also used this value identically for the proposed decoding algorithm. Meanwhile, the weighting value for the HISO decoding in the SBDA-I is equal to zero since it does not perform separate error correction when detecting a no-error syndrome. However, in the case of the SBDA-II and the proposed algorithm, only a single HDD is commonly needed for each HISO decoding, such that a value of 1 is assigned as a weighting.

First of all, we compare the relative complexity of SISO decoding required in each algorithm based on Table 1. The relative SISO complexity of each syndrome-based decoding algorithm can be expressed by the following equation indicating the relative ratio when

Table 1 The usage number of SISO decoding in each syndrome-based decoding algorithm

	The number of required SISO decoding
SBDA-I	$S_1 = l_{ave} \times n \times (1 - \lambda_{zero})$
SBDA-II	$S_2 = l_{ave} \times n \times (\lambda_{DESS} + \lambda_{SESS})$
Proposed	$S_3 = l_{ave} \times n \times (\lambda_{DESS} + \lambda_{SESS})$

Table 2 The usage number of HDD in each syndrome-based decoding algorithm

	The number of required HDD operations
SBDA-I	$H_1 = I_{ave} \times n \times 2^p \times (1 - \lambda_{zero})$
SBDA-II	$H_2 = I_{ave} \times n \times \left\{2^{p-1}(\lambda_{DESS} + \lambda_{SESS}) + \lambda_{SEHS}\right\}$
proposed	$H_3 = I_{ave} \times n \times \left\{2^{p-1}(\lambda_{DESS} + \lambda_{SESS}) + (\lambda_{SEHS} + \lambda_{DEHS})\right\}$

assuming the usages of SISO decoding in the Chase-Pyndiah algorithm as one.

$$R^j_{SISO} = \frac{S_j}{I_{ave} \times n} \tag{15}$$

where the superscript j in R^j_{SISO} also stands for the type of decoding algorithm, and the denominator of Eq. (15) refers to the usage number of SISO decoding required during error correction when using the Chase-Pyndiah algorithm.

Figure 8 shows the relative complexity of SISO decoding required when applying the three kinds of syndrome-based decoding algorithms to $(64, 57, 4)^2$ TPC, as determined by applying Eq. (15). The part (*a*) in Fig. 8 is the case that the TPC decoding is performed without early stopping scheme [27], and (*b*) is the opposite case. In addition, the results of relative complexity concerning the proposed algorithm are divided into two cases. The first is the case of applying the BFHDD-HISO decoding only, and the second is the use of both proposed schemes. The magnitude of the threshold S_{max} for the early termination technique used for the $(64, 57, 4)^2$ TPC was set to four.

Let us consider the relative complexity of each decoding algorithm in high SNR regions. In the case of (*a*) in Fig. 8, the error correction is conducted without early stopping. The usage number of SISO decoding in the proposed algorithm is reduced to nearly 1/10 compared with that of the Chase-Pyndiah algorithm at the SNR of 4.0 dB. Besides, comparing with the conventional syndrome-based decoding algorithms, the complexities are decreased by about 55.44% and 26.84% of each than that of SBDA-I and SBDA-II, respectively. Conversely, in the case of (*b*), the number of SISO decoding required in the proposed algorithm lowered to about 1/4 in comparison to that obtained by the Chase-Pyndiah algorithm, and the complexities diminished by almost 53.38% and 26.89% when compared to that of SBDA-I and SBDA-II, respectively. Moreover, in all cases of Fig. 8, we can verify that the results of relative complexity under the condition of applying BFHDD-HISO decoding alone or both schemes are the same. The reason for this is that the complexity reduction effect obtainable by applying the early termination decreases as the SNR increases.

Meanwhile, especially in (*b*), the complexity gap of each decoding algorithm tends to increase as the value of SNR rises. The reason for this is as follows. The higher the SNR regions, the smaller the relative noise power; therefore, it is very likely that the error bit

Table 3 The definition of symbols used in Tables 1 and 2

Symbol	Meaning
I_{ave}	The average half-iteration number
λ_{zero}	The ratio of no-error syndrome detection
λ_{SESS}	The ratio at which SISO decoding is applied when the single-error syndrome is detected
λ_{DESS}	The ratio at which SISO decoding is applied when the double-error syndrome is detected
λ_{SEHS}	The ratio at which HDD-based HISO decoding is applied when the single-error syndrome is detected
λ_{DEHS}	The ratio at which BFHDD-HISO decoding is applied when the double-error syndrome is detected

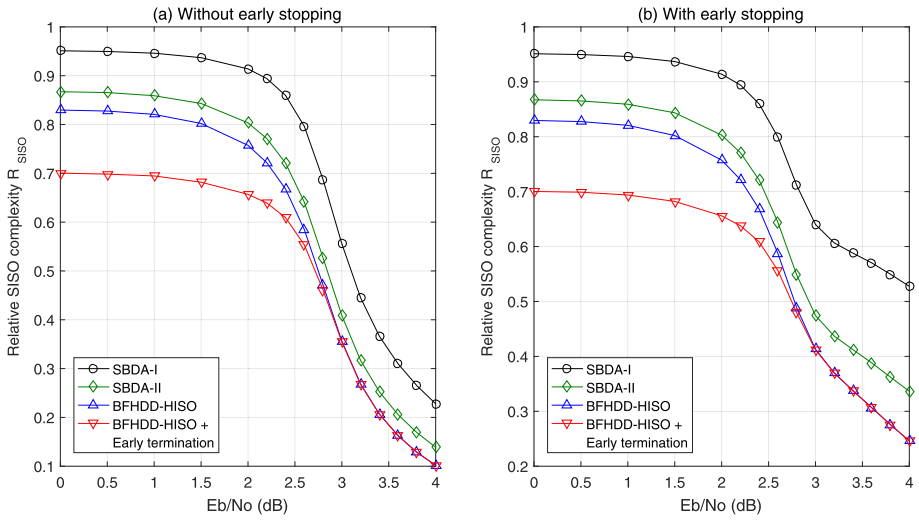

Fig. 8 Relative complexity results for $(64, 57, 4)^2$ TPC compared to the Chase-Pyndiah decoding algorithm [2] in terms of the SISO decoding requirements. **a** Without early stopping. **b** With early stopping. (l_{max}, 8; LRB number, 4)

positions belong to the range of p LRBs. For this reason, the HDD-based HISO and the proposed BFHDD-HISO decoding algorithms are applied more frequently, specifically proportional to the increment of SNR. As a result, the complexity reduction induced by the BFHDD-HISO decoding is greater than that of the SBDA-II; similarly, the complexity reduction arisen by the SBDA-II is more than that of the SBDA-I as well.

In addition, regardless of the decoding algorithms, under SNR greater than 2.5 dB, we can identify that the magnitude of the relative complexity when applying the early stopping is higher than that of the case of not using, which can be explained as follows. If the error correction succeeds in any mth half-iteration, the subsequent iterative decoding process is unnecessary. However, if the early stopping is not employed, such as in (a), the error correction is conducted continually until the maximum number of iterations is reached. In this case, it is possible to use only high-complexity SISO decoding when error correction is based on the conventional Chase-Pyndiah algorithm. Therefore, it is likely to cause a substantial increment in the decoding complexity. However, if the error correction is conducted using syndrome-based decoding algorithms, the low-complexity HISO decoding algorithm is applicable with a very high probability in the iterations after successful error correction. Despite successful error correction, some residual error may remain in subsequent iterations owing to the incompleteness of the calculated extrinsic information; still, it is highly probable that these can also be corrected by using the HDD-based HISO or the proposed BFHDD-HISO decoding. Therefore, when TPC decoding is conducted without applying early stopping like (a), the relative complexity is diminished more significantly as the iterative decoding progresses. Conversely, as shown in (b), the degree of complexity reduction is small because no further error correction is needed after successful decoding.

The following is about the complexity analysis when the channel condition is poor. In low SNR regions, the early termination is applied more effectively because of the considerably high probability of receiving undecodable blocks. Therefore, it is possible to reduce

the AHIN efficiently. The proposed algorithm results in a nearly 30% reduction in complexity compared with the Chase-Pyndiah algorithm. Moreover, when compared with the result of the SBDA-II, we can verify that there is almost more than twice the complexity reduction gain in the proposed algorithm. Meanwhile, the proposed BFHDD-HISO decoding is not much effective in low SNR regions. The reason for this is that the worse the channel state is, the more likely it is that the decoder input vector contains more than two even-numbered errors. Besides, even if the input vector contains only two errors, it is more frequent that Eq. (12) is not satisfied.

Next, we examine the complexity related to the usage number of HDD during the error correction of each syndrome-based decoding algorithm based on Table 2. Here, the more precise analysis is possible because the increment in complexity caused by the application of HISO, HDD-based HISO, and BFHDD-HISO decoding is considered. The relative complexity of HDD is expressed using the following equation.

$$R^{j}_{\mathrm{HDD}} = \frac{H_j}{I_{\mathrm{ave}} \times n \times 2^p} \tag{16}$$

where the denominator refers to the usage number of HDD during decoding based on the Chase-Pyndiah algorithm, requiring 2^p test patterns for any input vector.

Based on the Eq. (16), Fig. 9 illustrates the relative HDD complexity of the SBDA-II and the proposed algorithm. The superscript j of R^{j}_{HDD} denotes the type of decoding algorithm. The TPCs used in this complexity analysis are $(32, 26, 4)^2$, $(64, 57, 4)^2$, $(128, 120, 4)^2$, and $(256, 247, 4)^2$. We set the value of $S_{\max}$ of each code associated with the early termination as 5, 4, 3, and 2, respectively, using Monte-Carlo simulations.

When BFHDD-HISO decoding is employed alone (i.e., broken lines), the complexity is reduced regardless of the SNR region. We can verify that there is about an 85% reduction in complexity on average compared to the Chase-Pyndiah algorithm in high SNR regions. Noticeably, when compared with the SBDA-II, it is possible to lower the relative complexity of HDD effectively from a minimum of 14.58% to a maximum of 25.63%, depending on the kinds of codes. In addition, as the value of the SNR increases, the complexity gap expands more. The reason for this is that, as we aforementioned above, the

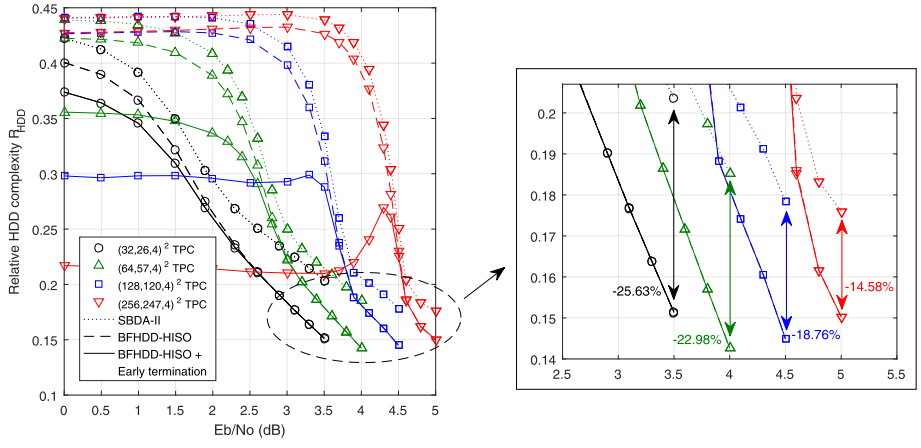

Fig. 9 Relative complexity results of four types of TPCs compared with the Chase-Pyndiah decoding algorithm [2] in terms of HDD requirements. ($I_{\max}$, 8; LRB number, 4)

better the channel condition, and the more progress in the iterative decoding, the applicability of BFHDD-HISO decoding increases. Further, the complexity gap increases due to the faster decoding convergence of the proposed algorithm compared with the SBDA-II. We examined the decoding convergence of the two algorithms via simulations, to confirm that the AHIN of the proposed algorithm was comparatively smaller than that of the SBDA-II. For example, assuming the decoding of $(64, 57, 4)^2$ TPC is conducted at SNR 3.5 dB, each AHIN value of the SBDA-II and the proposed algorithm are 4.7901 and 4.6376. If the value of SNR is 4.0 dB, these values are 3.5818 and 3.2732, respectively. Thus, we can verify that there arises a larger complexity gap as the SNR increases.

In sequence, the solid lines in Fig. 9 represent the results of relative HDD complexity when both BFHDD-HISO decoding and early termination are jointly applied. In the high SNR regions, the relative complexity results under both schemes are similar to that of using only BFHDD-HISO decoding. The reason for this is that most received decoding blocks are more likely to succeed in error correction, i.e., the early termination is almost ineffective. However, in low SNR regions, complexity reduction depends on the value of $S_{\max}$ set differently for each code. We can confirm that the complexity is lowered to about 62.63%, 64.43%, 70.21%, and 78.31% as compared with that of the Chase-Pyndiah algorithm.

When applying the proposed algorithm, the magnitude of $S_{\max}$ tends to be chosen smaller inversely proportional to the codeword length of the TPC, which can be explained as follows. Figure 10 represents the average number of errors depending on the SNR in each TPC received from the AWGN channel, i.e., the number of initial errors before performing TPC decoding. Assuming the average number of errors in the $(64, 57, 4)^2$ TPC at SNR 0.0 dB is about 427, the SNR regions where a similar number of errors occur in $(128, 120, 4)^2$ and $(256, 247, 4)^2$ TPCs are about 3.3 dB and 5.1 dB, respectively. In this case, the error ratios existing in each codeword are about 10.42%, 2.61%, and 0.65%, respectively. This result indicates that the shorter the length of the TPC, the more likely that the errors are more densely positioned. In fact, the TPC decoding is influenced by not only the number of errors but also the error pattern [36]. It is because the TPCs are generally composed of the form of a matrix with over the two-dimensional. Thus, the more the erroneous bits and the higher the density of errors, the more the negative factors such as the closed-chain [17] caused by the error pattern increase. For this reason, the decoding convergence speed may be slow temporarily in early iterations if the codeword length of the TPC is short. However, as the iterative decoding progresses, the detection number of the double-error syndrome can be rapidly reduced in subsequent iterations. In other words, if the number of errors is decreased to a certain level, the adverse effect caused by the error pattern is significantly lowered, leading to the success in error correction. Therefore, when the early termination is applied to TPCs, which is composed using a short length of the component code, a larger $S_{\max}$ value should be used to prevent performance degradation.

4.3 Complexity reduction analysis of the required number of arithmetic operations

Finally, we quantitatively estimate the usage number of various arithmetic operators for a more accurate analysis of the computational complexity of the proposed algorithm. The reason for this is that it is possible to analyze more fair complexity comparison by considering extra operators induced when applying the proposed BFHDD-HISO

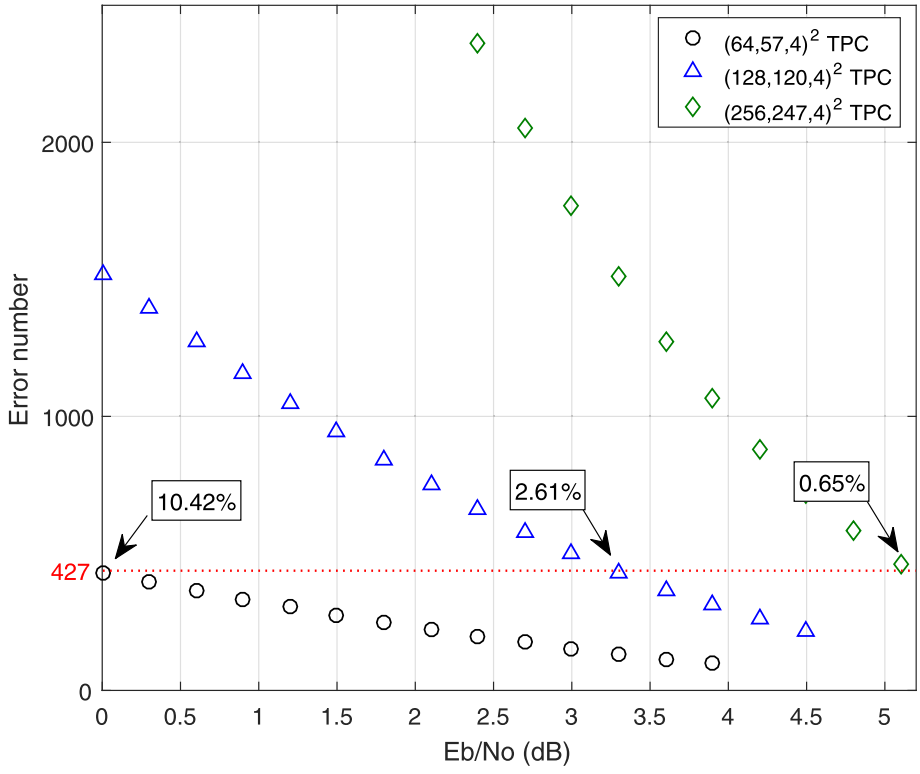

Fig. 10 The average number of errors according to the SNR region of the three types of TPCs received via the AWGN channel

decoding and early termination algorithms. The representative arithmetic operators used for complexity comparison are four: addition, multiplication, comparison, and modulo-2 addition.

Let us firstly look at the changes in the usage number of arithmetic operators when applying the proposed BFHDD-HISO decoding. If the SISO decoding is used when detecting a double-error syndrome, we should generate 2^p test patterns and test sequences in proportion to LRB number p, and also perform the same number of HDD operations. Subsequently, extrinsic information is calculated by using one of two Eqs. (7) and (8), depending on the presence of competing codewords. The computational complexity of Eq. (7) is even higher than the other, and which can be applied for calculating the extrinsic information of the maximum $(p + 2^p + 1)$ bits. Eventually, a considerable number of arithmetic operators is needed in the SISO decoding procedure. However, when BFHDD-HISO decoding is used, it is possible to generate the test pattern Z_{de} and the valid codeword D_{HISO} using bit-flipping and HDD operation only once each. Further, when calculating extrinsic information, Eq. (9), whose complexity is much lower than Eq. (7), is always used. Therefore, we can verify that the computational complexity of the proposed algorithm is much lower than the conventional SISO decoding. In addition, we have to consider the newly supplemented operators by applying BFHDD-HISO decoding as well. The extra operators are only a total of three comparators, including one for identifying double-error syndrome and the remaining two for the verification of the bit positions corrected by HDD. Thus, the computational complexity that is increased from

using the proposed algorithm is very slight compared to that of lowered by the decrement of applying SISO decoding.

Table 4 shows the total number of operators used to correct errors based on the proposed syndrome-based decoding algorithm. As shown in Fig. 1, the decoding algorithm of each input vector is first classified into three categories: no-error, single-error, and double-error, depending on the syndrome detected. Also, in the case of error syndrome detection, it is divided into two cases. The subscript c of $N_{c,t}^{\mathrm{DEC}}$ is used to distinguish the types of operators used (c: 1,. . . ,4), and t is for the discrimination of the decoding algorithms (t: 1,. . . ,5).

The proposed early termination technique did not exist in conventional TPC decoding algorithms. Thus, the number of arithmetic operators added for applying this method should be considered as well. The early termination, as we aforementioned before, is separated by two steps: the 1st half-iteration and subsequent half-iterations. First of all, the total number of double-error syndromes detected should be counted during the 1st half-iteration. During this process, it is required that the comparison and addition operators be applied once each for an input vector. Therefore, each operator is used n times during a single half-iteration. In addition, the multiplication is used once each to calculate values of υ and P_1. Subsequently, in any mth half-iteration, the operators of comparison and addition are also used n times each, as in the first half-iteration. Further, to calculate the value of P_m and identify whether Eq. (14) is satisfied, we employ a single multiplication, addition, and comparison. Also, the addition and comparison are used once each to increase the tolerance counter S and confirm if its magnitude reaches to $S_{\max}$. Therefore, we can verify that the number of extra operators caused by applying early termination is also quite small, similar to the case of applying BFHDD-HISO decoding. As a result, the usage number of each arithmetic operator employed in the proposed early termination algorithm is displayed in Table 5 mathematically. The subscript i in $N_{c,i}^{\mathrm{ET}}$ denotes the number of half-iterations.

The equation about the total usage of each arithmetic operator consumed in the case of applying both schemes is defined as follows.

$$N_c^{\mathrm{total}} = n \times I_{\mathrm{ave}} \times \left\{ \sum_{t=1}^{5} (\phi_t \times N_{c,t}^{\mathrm{DEC}}) \right\} + N_{c,1}^{\mathrm{ET}} + \left\{ (I_{\mathrm{ave}} - 1) \times N_{c,m}^{\mathrm{ET}} \right\} \tag{17}$$

Table 4 The usage number of four arithmetic operators per decoding of a single row/column vector required in the error correction of the proposed syndrome-based decoding algorithm (reference to Fig. 1)

$N_{c,t}^{\mathrm{DEC}}$		Addition ($c = 1$)	Multiplication ($c = 2$)	Comparison ($c = 3$)	Modulo-2 ($c = 4$)
No error	HISO ($t = 1$)	n	$2n$	$2n - k$	$n(n - k)$
Single error	HDD-based HISO ($t = 2$)	n	$3n$	$n(d + 2) - k + 1$	$n(n - k + 1)$
	SISO ($t = 3$)	$n(2^{p-1} + 2)$	$n(2^{p-1} + 3)$	$2^{p-1}(2n + 3)$ $+n(d + 1) - k - 4$	$(2^{p-1} - 1)(n^2 - nk + k + p)$ $+n(n - k + 1)$
Double error	BFHDD-HISO ($t = 4$)	n	$3n$	$n(d + 2) - k - d + 5$	$n(n - k + 1)$
	SISO ($t = 5$)	$n(2^{p-1} + 2)$	$n(2^{p-1} + 3)$	$2^{p-1}(2n + 3)$ $+n(d + 1) - d - k$	$(2^{p-1} - 1)(n^2 - nk + k + p)$ $+n(n - k + 1)$

Table 5 The usage number of four arithmetic operators per half-iteration required in the proposed early termination algorithm

$N_{c,i}^{\mathrm{ET}}$	Addition ($c=1$)	Multiplication ($c=2$)	Comparison ($c=3$)	Modulo-2 ($c=4$)
1st half-iteration ($i=1$)	n	2	n	-
mth half-iteration ($i\neq 1$)	$n+2$	1	$n+2$	-

The first term on the right side in Eq. (17) stands for the total number of any operator c used in error correction, and the remaining two terms represent the sum of operators required in early termination. In the first term, ϕ_t indicates the applicability of each decoding method applied based on the syndrome detected, such that $\sum_{t=1}^{5}\phi_t = 1$, which is used for the following reason. During the error correction of the proposed algorithm, one of the five decoding algorithms is selected for an input vector. Therefore, the applicability of each decoding algorithm varies depending on the channel status and the iteration number; so, it should be estimated via simulations. In addition, the endpoint of the iterative decoding can be varied by applying the early termination and stopping techniques. Thus, the AHIN, i.e., I_{ave}, also has to be reflected. Conversely, in the second and third terms of Eq. (7), a probability value like ϕ_t does not need to be considered because the early termination is applied in half-iteration units rather than in each input vector unit. The number of operators consumed in the 1st half-iteration is always fixed because the early termination is applicable after performing at least two half-iterative decoding procedures. However, since the number of operators used in the following half-iterations is variable, it reflects the weight value $(I_{\mathrm{ave}}-1)$.

The relative complexity results of the four operators required in the proposed algorithms are shown in Tables 6, 7, 8, and 9. The values indicated in these four tables can be obtained by dividing the total number of arithmetic operators used in the proposed decoding algorithm, i.e., Eq. (7), by the number of operators required in the Chase-Pyndiah algorithm and the SBDA-II.

Let us first discuss the complexity results of the proposed algorithm based on the Chase-Pyndiah decoder. These results suggest that the proposed algorithm substantially reduces the computational complexity than does the Chase-Pyndiah algorithm in all four operators. In other words, it shows that the proposed algorithm is highly effective in lowering the usage number of arithmetic operators. The complexity of them is diminished

Table 6 The relative complexity result of $(32, 26, 4)^2$ TPC for four arithmetic operators on the proposed decoding algorithm when each complexity of Chase-Pyndiah algorithm and SBDA-II are set to 1 (Add: addition, Mult: multiplication, Comp: comparison, Mod-2: modulo-2)

SNR(dB)	Comparison with Chase-Pyndiah algorithm				Comparison with SBDA-II			
	Add	Mult	Comp	Mod-2	Add	Mult	Comp	Mod-2
0.0	0.394495	0.426616	0.404718	0.355014	0.790736	0.845111	0.842361	0.748615
1.0	0.369514	0.405899	0.382056	0.333080	0.792639	0.853272	0.849599	0.753996
2.0	0.298011	0.349990	0.315328	0.270411	0.822307	0.911014	0.900990	0.796912
2.5	0.259579	0.318718	0.278528	0.235971	0.799718	0.904739	0.888998	0.782533
3.0	0.223062	0.284799	0.242887	0.204193	0.745826	0.859702	0.841167	0.740483
3.5	0.189242	0.252226	0.208943	0.174055	0.694333	0.815821	0.794351	0.698372
3.7	0.176981	0.240423	0.196323	0.163113	0.679410	0.805009	0.781364	0.687129

Table 7 The relative complexity result of $(64, 57, 4)^2$ TPC for four arithmetic operators on the proposed decoding algorithm when each complexity of Chase-Pyndiah algorithm and SBDA-II are set to 1 (Add: addition, Mult: multiplication, Comp: comparison, Mod-2: modulo-2)

SNR(dB)	Comparison with Chase-Pyndiah algorithm				Comparison with SBDA-II			
	Add	Mult	Comp	Mod-2	Add	Mult	Comp	Mod-2
0.0	0.360384	0.384786	0.366661	0.324737	0.697383	0.739349	0.739360	0.659354
1.0	0.358526	0.383470	0.365102	0.323134	0.699304	0.742017	0.741984	0.661561
2.0	0.347826	0.376845	0.356487	0.314005	0.717658	0.766158	0.765459	0.681699
2.5	0.330915	0.369542	0.343051	0.299654	0.780605	0.844463	0.841401	0.747770
3.0	0.263462	0.316738	0.276035	0.240949	0.831389	0.924829	0.911808	0.812799
3.5	0.219545	0.276995	0.233182	0.202099	0.778747	0.884732	0.868392	0.772682
4.0	0.180289	0.239847	0.195073	0.167226	0.714671	0.830879	0.812782	0.721825

to less than half in all four TPCs, regardless of the SNR regions. Besides, the relative complexities of all operators are less than 25% in high SNR regions corresponding to the BER 10^{-6} of each TPC. Notably, the complexity reductions of the addition and modulo-2 operators are the most significant among them.

Furthermore, we investigate the complexity results of the proposed algorithm based on the SBDA-II in more detail. We can verify that the computational complexity in the low SNR region is lower than that of other SNR regions. The reason for this is that, as with the previous relative complexity analysis, it was greatly influenced by the early termination technique. In other words, the lower the SNR, the greater the gap in the AHIN between the SBDA-II and the proposed algorithm. In particular, the longer the TPC codeword length, the higher the gain in computational complexity reduction arises. For example, when the error correction is conducted using $(256, 247, 4)^2$ TPC in SNR 3.5dB, the decoding complexity of the modulo-2 operator is reduced to about 31.82%. In addition, as the value of SNR increases, the degree of complexity reduction of all operators tends to decline regardless of TPCs. The reason for this is that the applicability of early termination is gradually diminished accordingly. Finally, in the high SNR region, i.e., the waterfall region of the BER curve, we can confirm that the complexity reduction increases again in proportion to the SNR rise. It is also related to the increased applicability of the BFHDD-HISO decoding algorithm, similar to the complexity analysis of HDD and SISO decoding. In other words, in high SNR regions, the magnitude of ϕ_4 increases gradually, and conversely, that of ϕ_5 declines increasingly. Thus, it is possible to decode with more low-complexity. In particular, the computational complexity of addition and

Table 8 The relative complexity result of $(128, 120, 4)^2$ TPC for four arithmetic operators on the proposed decoding algorithm when each complexity of Chase-Pyndiah algorithm and SBDA-II are set to 1 (Add: addition, Mult: multiplication, Comp: comparison, Mod-2: modulo-2)

SNR(dB)	Comparison with Chase-Pyndiah algorithm				Comparison with SBDA-II			
	Add	Mult	Comp	Mod-2	Add	Mult	Comp	Mod-2
2.0	0.281401	0.298393	0.284807	0.253641	0.542004	0.571195	0.572731	0.512133
2.5	0.277994	0.295498	0.281708	0.250654	0.541414	0.571257	0.572728	0.512017
3.0	0.288120	0.309429	0.293435	0.260157	0.586134	0.621624	0.622713	0.556107
3.5	0.319036	0.358388	0.329765	0.289802	0.786831	0.851565	0.848466	0.756271
3.9	0.232026	0.284852	0.238346	0.213565	0.842469	0.937952	0.918825	0.827780
4.2	0.209189	0.263290	0.217154	0.193299	0.797571	0.896417	0.881109	0.790621
4.5	0.184042	0.240194	0.193833	0.171058	0.753972	0.860652	0.846065	0.757026

Table 9 The relative complexity result of $(256, 247, 4)^2$ TPC for four arithmetic operators on the proposed decoding algorithm when each complexity of Chase-Pyndiah algorithm and SBDA-II are set to 1 (Add: addition, Mult: multiplication, Comp: comparison, Mod-2: modulo-2)

SNR(dB)	Comparison with Chase-Pyndiah algorithm				Comparison with SBDA-II			
	Add	Mult	Comp	Mod-2	Add	Mult	Comp	Mod-2
3.5	0.174314	0.183719	0.175623	0.157071	0.337003	0.352963	0.354922	0.318219
4.0	0.207845	0.222399	0.210931	0.187699	0.428504	0.452181	0.454235	0.406517
4.2	0.263805	0.288074	0.269797	0.238957	0.589739	0.628781	0.629919	0.562833
4.4	0.294754	0.335571	0.303778	0.268663	0.789309	0.856129	0.852279	0.762068
4.6	0.229201	0.279516	0.232309	0.211294	0.857501	0.946509	0.929408	0.840487
4.8	0.205451	0.256428	0.207309	0.190142	0.835578	0.929628	0.907596	0.823941
5.0	0.191185	0.243109	0.195552	0.177495	0.797229	0.891691	0.879294	0.791919

modulo-2 operations are decreased most substantially. For instance, in the case of decoding by $(32, 26, 4)^2$ TPC in SNR 3.7dB corresponding to the BER 10^{-6}, the computational complexities of each operator are reduced by approximately 32.06% and 31.29%.

4.4 BER performance comparisons

Figure 11 shows the BER performances of TPC, which use the $(32, 26, 4)$ extended Hamming codes as component codes. These results are obtained by applying error correction through the Chase-Pyndiah, existing two syndrome-based, and the proposed decoding algorithms. We can verify that all BER curves, except that of a broken purple line, are almost identical. However, the error correction results of the SBDA-II and the proposed algorithms show that there are slight performance gaps compared with that of the Chase-Pyndiah algorithm. This result is attributed to a reduction in the applicability of Eq. (7) when calculating the extrinsic information. In the proposed decoding algorithm, we calculate the extrinsic information using reliability factors such as δ_1, δ_2, and δ_3 if SISO decoding is not applicable. Thus, because the accuracy of extrinsic information obtained using these factors is lower than that obtained based on Eq. (7) in the Chase-Pyndiah algorithm, it is causing performance degradation.

In addition, we can find that the performance gap is slightly higher in the proposed algorithm than in the SBDA-II. The reason for this is that the applicability of Eq. (7) in the proposed algorithm is further lowered than before. However, the performance loss that occurred when applying the proposed decoding algorithm is trivial, i.e., under 0.1 dB at BER 10^{-6} as compared with that of the Chase-Pyndiah algorithm.

Meanwhile, the broken purple line is the BER performance result when the HDD-based HISO decoding proposed in [25] is applied to not only single-error but also double-error syndrome detection. As mentioned earlier, this result reveals that error correction capability is significantly degraded, because it is very likely to be performed the wrong error correction via HDD-based HISO decoding. Consequently, in this case, there is a performance loss of over 0.5 dB at BER 10^{-6}.

Finally, Fig. 12 compares the error correction performances, which are obtained by applying the conventional and proposed decoding algorithms to the four TPCs. Similar to the previous result of Fig. 11, we can verify that the BER performances of the proposed algorithm are almost similar to that of the Chase-Pyndiah algorithm, regardless of the kind of TPC. In particular, the longer the TPC codeword length, the smaller the performance gap is between the Chase-Pyndiah and proposed algorithms. The reason for this is as follows.

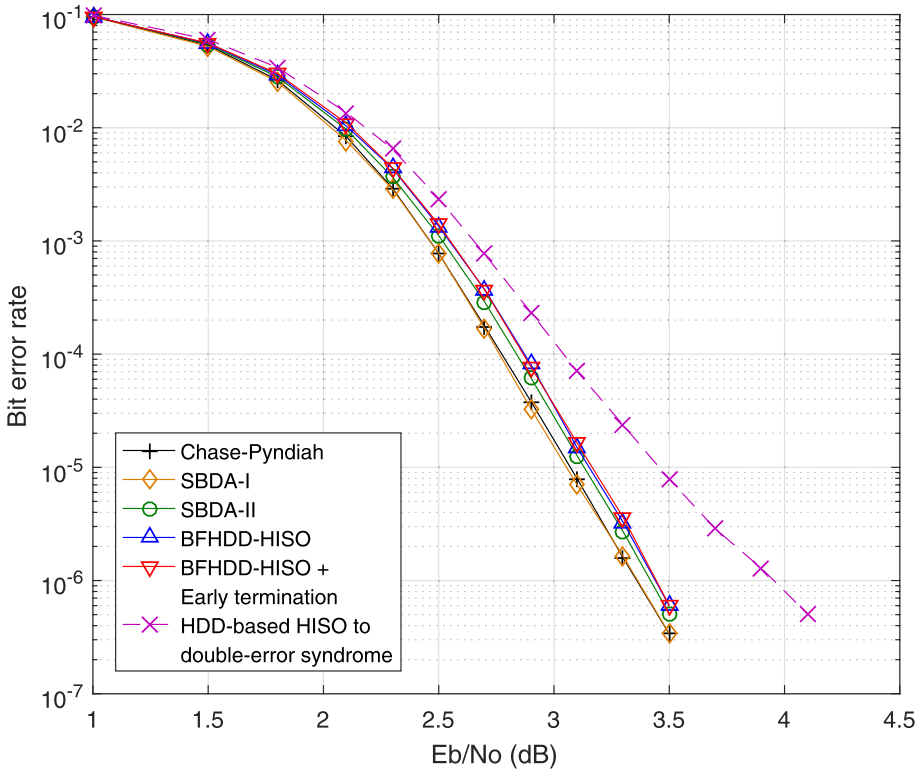

Fig. 11 Error correction performance results of different decoding algorithms of $(32, 26, 4)^2$ TPC (I_{max}, 8; LRB number, 4.)

In general, the reliability factors are used to calculate extrinsic information when there are no competing codewords. For example, assuming the LRB number is p, the calculation of extrinsic information using Eq. (7) is possible for up to $(p + 2^p + 1)$ bit positions during the error correction of the input vector based on the Chase-Pyndiah algorithm. In addition, let us assume that all input vectors in any half-iteration have the maximum number of bit positions having competing codewords. Then, the soft outputs of $n(n-(p+2^p+1))$ bit positions are computed using the reliability factor β. Therefore, in the case of decoding of $(256, 247, 4)^2$ TPC, about 91.80% of the bits in the TPC codeword use β for calculation of extrinsic information. Similarly, in the case of decoding of $(32, 26, 4)^2$ TPC, about 34.38% of the bits use β for the same reason. However, the applicability of SISO decoding is minimized in the proposed algorithm, causing the much more frequent use of reliability factors. Thus, the rate of using the reliability factors can be more significantly increased as the length of the component code is short. As we aforementioned, the accuracy of calculating extrinsic information using β is lesser than that obtained using Eq. (7), so it has a negative impact on error correction. For this reason, the shorter the length of the component code used, the larger the performance gap between the Chase-Pyndiah and the proposed algorithm gets. Conversely, the longer the length of n, it is not so detrimental that the influence of the increment of using reliability factors. Thus, the error correction performances of both algorithms are almost identical, as shown in Fig. 12. As a result, the proposed schemes effectively reduce the computational complexity while maintaining nearly the same BER performance as that of the conventional TPC decoding algorithm.

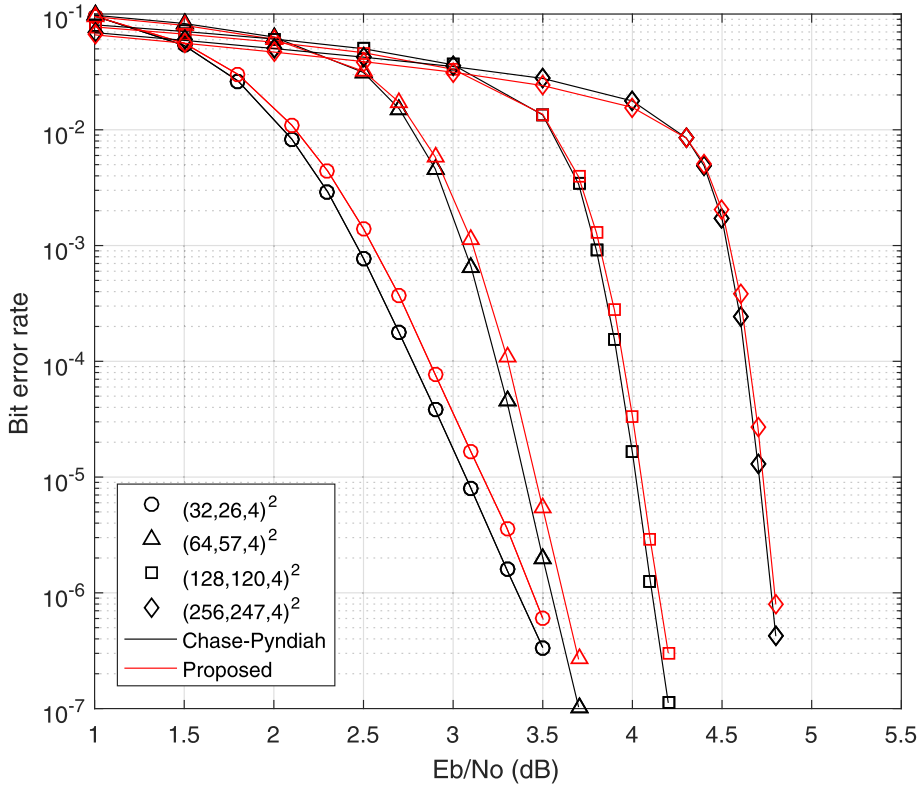

Fig. 12 Error correction performance results of the Chase-Pyndiah and the proposed decoding algorithm in three kind of TPCs (I_{max}, 8; LRB number, 4.)

5 Conclusions

This paper introduces two effective techniques, which are based on the syndrome, to reduce the decoding complexity of the TPC. First, if a double-error syndrome is detected, it conventionally had to apply the high-complex SISO decoding for error correction. To resolve problems related to high complexity, we propose the low-complexity BFHDD-HISO decoding algorithm. Whereas 2^p HDD operations are performed proportional to the LRB number p in the conventional methods, the proposed algorithm requires only a single HDD operation. Therefore, this technique effectively lowers the decoding complexity. The proposed BFHDD-HISO decoding algorithm is more effective as the channel status is better, and the more iterative decoding procedure progresses. Second, we propose an early termination algorithm that is applicable to undecodable blocks. Through this, we can continuously check the possibility of error correction failure. If the received codeword is determined to be undecodable, the iterative decoding is terminated early, so that AHIN can be reduced significantly. Similar to the first proposal, this technique is also based on the syndrome detection results of decoder input vectors. In particular, it uses the detection number of the double-error syndrome and its decreasing ratio. We have confirmed that the proposed early termination algorithm is more effective when the codeword length of the TPC is lengthy. As a result, this paper has substantially lowered the computational complexity of the TPC decoding based on the two proposed techniques. The complexity analysis has been conducted by various indicators, such as the number

of applying SISO decoding, the usage number of HDD, and the number of various arithmetic operators required during decoding. Finally, simulation results show that the error correction performance of the proposed decoding algorithm is almost the same as that achieved with conventional methods.

Abbreviations

TPC: Turbo product code; HIHO: Hard-input hard-output; HISO: Hard-input soft-output; SISO: Soft-input soft-output; HDD: Hard decision decoding; LRB: Least reliable bit; SED: Squared Euclidean distance; SNR: Signal-to-noise ratio; LDPC: Low-density parity check; BPSK: Binary phase shift keying; AWGN: Additive white Gaussian noise; BER: Bit error rate; AHIN: Average half-iteration number; BFHDD: Bit-flipping and hard decision decoding; SBDA-I: Syndrome-based decoding algorithm I; SBDA-II: Syndrome-based decoding algorithm II; HARQ: Hybrid automatic repeat and request

Acknowledgements

Not applicable.

Authors' contributions

SY designed the main concept of the proposed algorithms, analyzed their error correction performances and complexity, and wrote this paper. BA conducted an idea review and performance analysis of the proposed technique, and wrote this paper. JH gave valuable suggestion on the idea of this paper. All authors read and approved the final manuscript.

Funding

This work was supported as part of Military Crypto Research Center(UD170109ED) funded by Defense Acquisition Program Administration(DAPA) and Agency for Defense Development(ADD).

Availability of data and materials

All data generated or analyzed during this study are included in this paper.

Competing interests

The authors declare that they have no competing interests.

Author details

[1]The School of Electrical Engineering, Korea University, 145, Anam-ro, Seongbuk-gu, 02841, Seoul, Republic of Korea. [2]The School of Electrical Engineering, Korea University, 145, Anam-ro, Seongbuk-gu, 02841, Seoul, Republic of Korea.

Received: 9 May 2019 Accepted: 21 May 2020
Published online: 18 June 2020

References

1. R. Pyndiah, A. Glavieux, A. Picart, S. Jacq, in *1994 IEEE GLOBECOM*, Near optimum decoding of product codes (IEEE, Communications: The Global Bridge, 1994), pp. 339–343
2. R. M. Pyndiah, Near-optimum decoding of product codes: block turbo codes. IEEE Trans. Commun. **46**(8), 1003–1010 (1998)
3. J. Li, K. R. Narayanan, E. Kurtas, C. N. Georghiades, On the performance of high-rate TPC/SPC codes and LDPC codes over partial response channels. IEEE Trans. Commun. **50**(5), 723–734 (2002)
4. IEEE standard for broadband over power line networks: medium access control and physical layer specifications. IEEE Std 1901-2010, 1-1586 (2010). https://doi.org/10.1109/IEEESTD.2010.5678772
5. IEEE standard for local and metropolitan area networks - part 20: air interface for mobile broadband wireless access systems supporting vehicular mobility - physical and media access control layer specifications. IEEE Std 802.20-2008, 1-1039 (2008). https://doi.org/10.1109/IEEESTD.2008.4618042
6. IEEE standard for local and metropolitan area networks part 16: air interface for broadband wireless access systems. IEEE Std 802.16-2009 (Revision of IEEE Std 802.16-2004), 1-2080 (2009). https://doi.org/10.1109/IEEESTD.2009.5062485
7. N. Gunaseelan, L. Liu, J.-F. Chamberland, G. H. Huff, Performance analysis of wireless hybrid-ARQ systems with delay-sensitive traffic. IEEE Trans. Commun. **58**(4), 1262–1272 (2010)
8. T.-Y. Lin, S.-K. Lee, H.-H. Tang, M.-C. Lin, An adaptive hybrid ARQ scheme with constant packet lengths. IEEE Trans. Commun. **60**(10), 2829–2840 (2012)
9. H. Mukhtar, A. Al-Dweik, M. Al-Mualla, in *2015 IEEE Global Communications Conference (GLOBECOM)*, Low complexity hybrid ARQ using extended turbo product codes self-detection (IEEE, 2015), pp. 1–6
10. H. Mukhtar, A. Al-Dweik, M. Al-Mualla, in *2015 International Conference on Information and Communication Technology Research (ICTRC)*, Hybrid ARQ with partial retransmission using turbo product codes (IEEE, 2015), pp. 28–31
11. H. Mukhtar, A. Al-Dweik, M. Al-Mualla, CRC-free hybrid ARQ system using turbo product codes. IEEE Trans. Commun. **62**(12), 4220–4229 (2014)
12. H. Mukhtar, A. Al-Dweik, M. Al-Mualla, A. Shami, Adaptive hybrid ARQ system using turbo product codes with hard/soft decoding. IEEE Commun. Lett. **17**(11), 2132–2135 (2013)
13. D. Chase, Class of algorithms for decoding block codes with channel measurement information. IEEE Trans. Inf. Theory. **18**(1), 170–182 (1972)
14. A. Omar, in *OFC 2001. Optical Fiber Communication Conference and Exhibit. Technical Digest Postconference Edition (IEEE Cat. 01CH37171), vol. 2*, FEC techniques in submarine transmission systems (IEEE, 2001), pp. 1–1
15. G. Bosco, G. Montorsi, S. Benedetto, A new algorithm for "hard" iterative decoding of concatenated codes. IEEE Trans. Commun. **51**(8), 1229–1232 (2003)

16. A. J. Al-Dweik, B. S. Sharif, Non-sequential decoding algorithm for hard iterative turbo product codes-[transactions letters]. IEEE Trans. Commun. **57**(6), 1545–1549 (2009)
17. A. J. Al-Dweik, B. S. Sharif, Closed-chains error correction technique for turbo product codes. IEEE Trans. Commun. **59**(3), 632–638 (2011)
18. A. Al-Dweik, S. Le Goff, B. Sharif, A hybrid decoder for block turbo codes. IEEE Trans. Commun. **57**(5), 1229–1232 (2009)
19. J. Cho, W. Sung, in *2011 IEEE Workshop on Signal Processing Systems (SiPS)*, Reduced complexity Chase-Pyndiah decoding algorithm for turbo product codes (IEEE, 2011), pp. 210–215
20. X. Dang, M. Tan, X. Yu, Adaptive decoding algorithm based on multiplicity of candidate sequences for block turbo codes. China Commun. **11**(13), 9–15 (2014)
21. L. Li, F. Zhang, P. Zheng, Z. Yang, in *2014 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, Improvements for decoding algorithm of turbo product code (IEEE, 2014), pp. 374–378
22. J. Son, K. Cheun, K. Yang, Low-complexity decoding of block turbo codes based on the chase algorithm. IEEE Commun. Lett. **21**(4), 706–709 (2017)
23. E.-H. Lu, P.-Y. Lu, in *2010 International Symposium on Computer, Communication, Control and Automation (3CA), vol. 1*, A syndrome-based hybrid decoder for turbo product codes (IEEE, 2010), pp. 280–282
24. P.-Y. Lu, E.-H. Lu, T.-C. Chen, An efficient hybrid decoder for block turbo codes. IEEE Commun. Lett. **18**(12), 2077–2080 (2014)
25. B. Ahn, S. Yoon, J. Heo, Low complexity syndrome-based decoding algorithm applied to block turbo codes. IEEE Access. **6**, 26693–26706 (2018)
26. B. Ahn, S. Ha, S. Yoon, J. Heo, in *Proceedings of the 2016 International Conference on Communication and Information Systems*, An efficient decoding scheme for improved throughput in three dimensional turbo product codes based on single parity code (ACM, 2016), pp. 118–122
27. G. T. Chen, L. Cao, L. Yu, C. W. Chen, An efficient stopping criterion for turbo product codes. IEEE Commun. Lett. **11**(6), 525–527 (2007)
28. F.-M. Li, A.-Y. Wu, On the new stopping criteria of iterative turbo decoding by using decoding threshold. IEEE Trans. Sig. Process. **55**(11), 5506–5516 (2007)
29. L. Huang, Q. Zhang, L. L. Cheng, Information theoretic criterion for stopping turbo iteration. IEEE Trans. Sig. Process. **59**(2), 848–853 (2011)
30. J. Geldmacher, K. Hueske, J. Götze, M. Kosakowski, in *2011 8th International Symposium on Wireless Communication Systems*, Hard decision based low SNR early termination for LTE turbo decoding (IEEE, 2011), pp. 26–30
31. M. AlMahamdy, J. Dill, in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, Early termination of turbo decoding by identification of undecodable blocks (IEEE, 2017), pp. 1–6
32. G. Han, X. Liu, A unified early stopping criterion for binary and nonbinary LDPC codes based on check-sum variation patterns. IEEE Commun Lett. **14**(11), 1053–1055 (2010)
33. T. Mohsenin, H. Shirani-mehr, B. Baas, in *2011 IEEE International Symposium of Circuits and Systems (ISCAS)*, Low power LDPC decoder with efficient stopping scheme for undecodable blocks (IEEE, 2011), pp. 1780–1783
34. C.-C. Cheng, J.-D. Yang, H.-C. Lee, C.-H. Yang, Y.-L. Ueng, A fully parallel LDPC decoder architecture using probabilistic min-sum algorithm for high-throughput applications. IEEE Trans. Circ. Syst. I Reg. Pap. **61**(9), 2738–2746 (2014)
35. T. Xia, H.-C. Wu, S. C.-H. Huang, in *2013 IEEE Global Communications Conference (GLOBECOM)*, A new stopping criterion for fast low-density parity-check decoders (IEEE, 2013), pp. 3661–3666
36. H. Mukhtar, A. Al-Dweik, A. Shami, Turbo product codes: applications, challenges, and future directions. IEEE Commun. Surv. Tutor. **18**(4), 3052–3069 (2016)

Publisher's Note