IEICE TRANS. FUNDAMENTALS, VOL.E93–A, NO.11 NOVEMBER 2010

PAPER Special Section on Information Theory and Its Applications

Multi-Stage Threshold Decoding for Self-Orthogonal Convolutional Codes

Muhammad AHSAN ULLAH[†a)], **Kazuma OKADA**[†], ***Nonmembers***, ***and*** **Haruo OGIWARA**[†b)], ***Member***

SUMMARY This paper describes a least complex, high speed decoding method named multi-stage threshold decoding (MTD-DR). Each stage of MTD-DR is formed by the traditional threshold decoder with a special shift register, called difference register (DR). After flipping each information bit, DR helps to shorten the Hamming and the Euclidian distance between a received word and the decoded codeword for hard and soft decoding, respectively. However, the MTD-DR with self-orthogonal convolutional codes (SOCCs), type 1 in this paper, makes an unavoidable error group, which depends on the tap connection patterns in the encoder, and limits the error performance. This paper introduces a class of SOCCs type 2 which can breakdown that error group, as a result, MTD-DR gives better error performance. For a shorter code (code length = 4200), hard and soft decoding MTD-DR achieves 4.7 dB and 6.5 dB coding gain over the additive white Gaussian noise (AWGN) channel at the bit error rate (BER) 10^{-5}, respectively. In addition, hard and soft decoding MTD-DR for a longer code (code length = 80000) give 5.3 dB and 7.1 dB coding gain under the same condition, respectively. The hard and the soft decoding MTD-DR experiences error flooring at high E_b/N_0 region. For improving overall error performance of MTD-DR, this paper proposes parity check codes concatenation with soft decoding MTD-DR as well.

key words: *threshold decoding, multi-stage threshold decoding, difference register, error group*

1. Introduction

Threshold decoding (TD) is considered as a least complex decoding technique in the field of coding theory [1]. The TD experiences catastrophic errors with some channel error patterns for convolutional codes [2]. The self-orthogonal convolutional codes (SOCCs) give limited error propagation with TD and prevent the catastrophic error flow [3]. However, the bit error performance of TD is not attractive. Russian scientists have proposed [4]–[7] an improved version of iterative TD, called multi-stage threshold decoding. They introduce an extra shift register called difference register (DR), which conveys the flipping messages of decoded information bits. So, the multi-stage threshold decoding with DR is named as MTD-DR. They have shown that, after flipping each information bit by hard decision, the Hamming distance between a received word and the decoded codeword, which is generated from decoded information bits, becomes shorter, in where DR plays an important role. The working principle of DR in the fundamental theorem of MTD-DR [4]–[7] is hardly understandable. So, this paper gives an intuitive proof of MTD-DR's theorem. They have claimed more than 6 dB coding gain over the additive white Gaussian noise channel (AWGN) at the bit error rate (BER) 10^{-5}. Unfortunately, codes and decoding algorithms are absent in their publications. Therefore, this paper reconstructs MTD-DR with various aspects and investigates the bit error performance over the AWGN channel.

This paper shows that, the MTD-DR, with SOCCs given in [8], generates a strong error group that depends on the tap connection pattern of the encoder. To improve the error performance of MTD-DR, a new class of SOCCs, type 2, is given in this paper. The hard decoding MTD-DR achieves 4.7 dB and 5.3 dB coding gain for a shorter code with code length 4200 and a longer code with code length 80000 at the BER 10^{-5}, respectively.

Iterative soft threshold decoding algorithm for SOCCs is shown in [9]. In the paper, they used min-sum decoding algorithm, which is widely used for decoding LDPC codes. It approximates symbol by symbol maximum a posteriori probability decoding. Moreover, they have considered self-doubly orthogonal convolutional codes, to avoid cycles with length 4 and 6 in the Tanner graph of the parity check matrix. Unlike min-sum decoding, this paper proposes several soft decoding algorithms, to reduce the Euclidian distance between a received word and the decoded codeword after flipping each information bit. This paper considers self-orthogonal codes, i.e. without cycles of length 4, rather than self-doubly orthogonal codes. This paper investigated the error performance of codes in [9] for MTD-DR, unfortunately, no improvement have been observed compared to a self-orthogonal code.

The soft decoding MTD-DR reduces the Euclidian distance between a received word and the decoded codeword after flipping each information bit. Weighted bit flipping algorithm for MTD-DR is also investigated in this paper. Combination of the weighted bit flipping algorithm and the soft decoding algorithm makes a new soft decoding algorithm for MTD-DR which is called a combined soft decoding MTD-DR (CMTD). A CMTD achieves 6.5 dB and 7.1 dB coding gain for the shorter code and the longer code, respectively, at the BER 10^{-5} over the AWGN channel. For further improving the overall error performance, this paper proposes a decoding scheme where parity check codes are connected serially with CMTD. Since MTD-DR is a low complex decoding scheme, it may be useful for high speed communication as well as for low power communication

Manuscript received February 10, 2010.
Manuscript revised May 24, 2010.

[†]The authors are with the Department of Electrical Engineering, Nagaoka University of Technology, Nagaoka-shi, 940-2188 Japan.
a) E-mail: ahsan@comm.nagaokaut.ac.jp
b) E-mail: ogiwara@vos.nagaokaut.ac.jp

DOI: 10.1587/transfun.E93.A.1932

equipment.

Rest of the paper is arranged as follows. Section 2 gives the concept of TD and MTD-DR and their hard decoding algorithms. Section 3 discusses the error grouping in the decoded information bit stream and how the SOCC type 2 breaks down that error group. Section 4 dedicates for the bit error performance of TD and MTD-DR. Section 5 gives soft decoding algorithms and illustrates their bit error performance with MTD-DR. Section 6 gives two constructions of combined soft decoding MTD-DR and provides the bit error performance of them. Section 7 discusses about the parity check codes concatenation with CMTDs and their bit error performance are given. Section 8 concludes this paper.

2. Threshold Decoding Concept

2.1 Threshold Decoding

A systematic SOCCs with code rate $R = 1/2$, tap weight J and memory length K are considered. An encoder of SOCC generates a codeword by using N information bits and makes a codeword of length $2N$. Two types of SOCCs are given.

- Type 1: When one generator polynomial generates a codeword from an information bit stream (with N bits) then the code is called SOCC type 1.
- Type 2: When multiple generator polynomials generate n parity bit streams by using n information bit streams (each bit stream has N/n bits, for parity and information bit streams) then the code is called SOCC type 2.

The dotted part in Fig. 1 shows an encoder structure of SOCC type 1 with $J = 4$, $K = 6$ and generator polynomial $G(D) = 1 + D + D^4 + D^6$. The encoder generates a parity bit sequence $V(D)$ which is calculated by

$$V(D) = G(D)X(D) \tag{1}$$

where $X(D)$ is an input information bit sequence to the encoder. The information and parity bit sequences make a systematic codeword $C(D) \triangleq \{X(D), V(D)\}$. The binary codeword is modulated as a binary phase shift keying (BPSK) signal to transmit through the AWGN channel. At the receiving end, channel output is converted into a binary received word by hard decision.

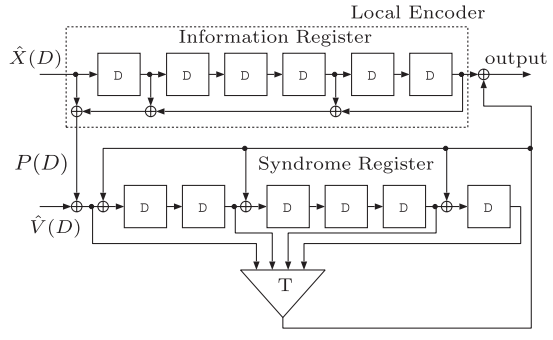


Fig. 1 Threshold decoder for SOCC type 1 with $J = 4$, $K = 6$, $G(D) = 1 + D + D^4 + D^6$. Here $P(D) = G(D)\hat{X}(D)$.

Figure 1 shows a TD for SOCCs type 1 (TD.Tp1). The hard decision channel output makes a received word $\hat{C}(D) \triangleq \{\hat{X}(D), \hat{V}(D)\}$, where $\hat{X}(D)$ is the received information bit stream stored in the information shift register and $\hat{V}(D)$ is the received parity bit stream. The TD generates a syndrome bit stream $S(D)$, which is stored in the syndrome shift register, by

$$S(D) = G(D)\hat{X}(D) \oplus \hat{V}(D) \tag{2}$$

where $\oplus$ represents the modulo two sum operator in this paper. One information bit affects J syndrome bits which make a checking syndrome set and each element of this set is called checking syndrome. For decoding j-th information bit, (the right most bit at the information register in Fig. 1), let $\{\mathbf{S}_j\}$ be the checking syndrome set and $s_{j,k}$ be the k-th checking syndrome in it. For hard decoding TD, the checksum value L_j, that is summation of checking syndromes, is calculated by

$$L_j = \sum_{s_{j,k} \in \{\mathbf{S}_j\}} s_{j,k} \tag{3}$$

The flipping decision is done by comparing threshold value T with the checksum value L_j. The threshold value of a given code is calculated by [1]

$$T = \left\lfloor \frac{J+1}{2} \right\rfloor \tag{4}$$

where $\lfloor x \rfloor$ represents the largest integer not greater than x. When L_j exceeds the threshold value ($L_j > T$), the decoding decision flips the j-th information bit. At the same time, checking syndromes are updated and then all the contents of the registers are shifted one position right. Otherwise the shifting is done without flipping. The tail-biting termination is used in this paper.

2.2 Multi-Stage Threshold Decoding

Multi-stage threshold decoding is an iterative TD. This paper discusses two configurations: (1) iterative TD with difference register (MTD-DR) and (2) iterative TD without difference register (MTD). Figure 2 shows an MTD-DR with $J = 4$ and $K = 6$. MTD is constructed by removing DR from Fig. 2. The DR holds all zero bits at the first stage and updates each bit by the flipping decision. The j-th checksum value L_j of MTD-DR is calculated by

$$L_j = \sum_{s_{j,k} \in \{\mathbf{S}_j\}} s_{j,k} + d_j \tag{5}$$

where $d_j \in \{1, 0\}$ is the j-th bit (right most bit) in the DR. The MTD, on the other hand, calculates the checksum value by Eq. (3). When the checksum value exceeds the threshold value, MTD-DR and MTD flip the target information bit

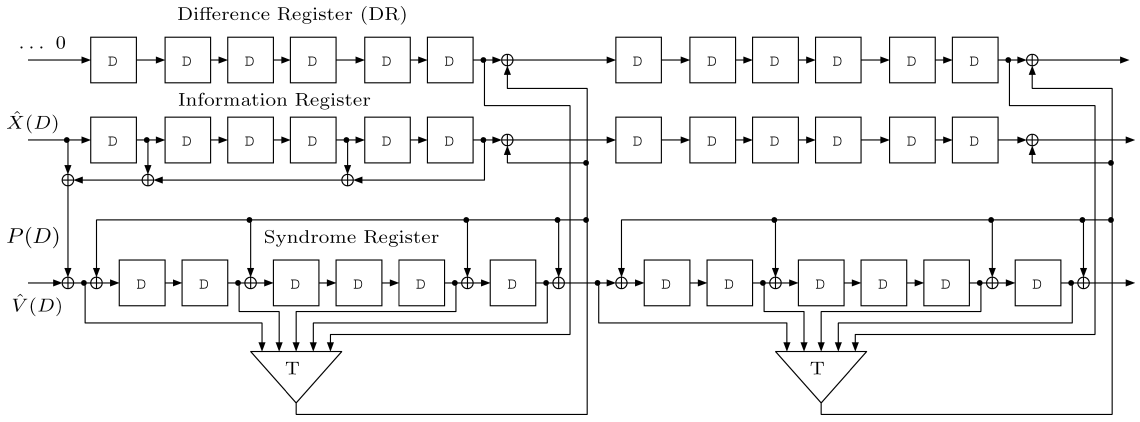


Fig. 2 MTD-DR for SOCC type 1 with $J = 4$, $K = 6$ and $G(D) = 1 + D + D^4 + D^6$. Here $P(D) = G(D)\hat{X}(D)$.

and their checking syndromes. At the same times, MTD-DR updates the related DR bit and then all the contents of the registers are shifted one position right. Otherwise, the shifting is done without flipping.

The difference register gives an unique feature of MTD-DR. We give an intuitive proof of the function of the difference register in the fundamental theorem of MTD-DR. We know that, the Hamming distance between a received word and the decoded codeword is the summation of the Hamming distance of the information part and that of the parity parts for systematic codes. Each bit of DR updates (flips the right most bit of DR in Fig. 2) by the flipping decision. It shows the difference between a received and a decoded information bit. That means, the Hamming weight of DR contents gives the Hamming distance of the information part between a received word and the decoded codeword. At the beginning of decoding, the DR holds all zero bits, this is because, the decoded information bit stream and the hard decision information bits of the received word are the same. The Hamming weight of syndrome bits is equal to the Hamming distance of parity parts between a received word and the decoded codeword, which is generated from the decoded information bits. At j-th bit decoding by the MTD-DR, one bit in DR and J bits in syndrome register are concerned. The flipping decision inverts more than $(J + 1)/2$ bits to zero among them, which confirms to reduce the Hamming distance, between the decoded codeword and a received word.

2.2.1 MTDs for SOCCs Type 1

This paper considers two constructions of MTDs for SOCCs type 1.

1. Multi-stage threshold decoding with difference register for SOCCs type 1 or MTD-DR.Tp1
2. Multi-stage threshold decoding without difference register for SOCCs type 1 or MTD.Tp1

The MTD.Tp1 is constructed by removing difference register from Fig. 2. MTD-DR.Tp1 and MTD.Tp1 calculate their checksum values by Eq. (5) and Eq. (3), respectively, and the decoding decision is done accordingly. The SOCCs type 1 make a strong error group which limits the error performance of MTD-DR (see Sect. 3).

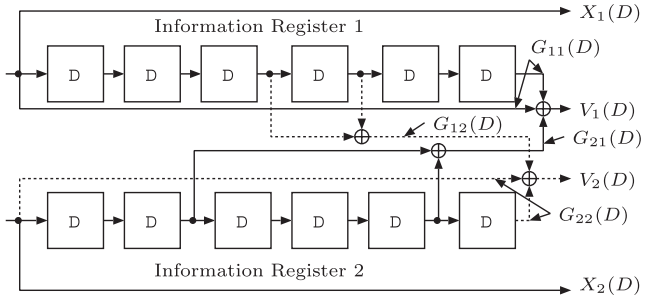


Fig. 3 An encoder of a SOCC type 2 with $J_{11} = J_{12} = J_{21} = J_{22} = 2$, $K = 6$, $G_{11}(D) = 1 + D^6$, $G_{12}(D) = D^3 + D^4$, $G_{21}(D) = D^2 + D^5$, and $G_{22}(D) = 1 + D^6$.

2.2.2 MTDs for SOCCs Type 2

In general, a SOCC type 2 uses n information bit streams and generates equal number of parity bit streams. This paper gives SOCCs type 2 for $n = 2$. Figure 3 shows an encoder of SOCC type 2 with $J_{xy} = 3$, $x = y = \{1, 2\}$ and $K = 6$. The encoder generates two parity sequences by using four generating polynomials and gives the code rate $R = 2/4$. Two information bit sequences ($X_1(D)$ and $X_2(D)$) produce two parity sequences by four generating polynomials $G_{11}(D)$, $G_{12}(D)$, $G_{21}(D)$ and $G_{22}(D)$. Parity sequences of SOCC type 2 are defined by

$$V_1(D) = G_{11}(D)X_1(D) \oplus G_{21}(D)X_2(D) \tag{6}$$

$$V_2(D) = G_{12}(D)X_1(D) \oplus G_{22}(D)X_2(D) \tag{7}$$

The information and parity bit sequences make a codeword $C(D) \triangleq \{X_1(D), V_1(D), X_2(D), V_2(D)\}$. MTD-DR for SOCCs type 2 (MTD-DR.Tp2) as well as MTD for SOCCs type 2 (MTD.Tp2) generate two syndrome sequences, which are defined by

$$S_1(D) = G_{11}(D)\hat{X}_1(D) \oplus G_{21}(D)\hat{X}_2(D) \oplus \hat{V}_1(D) \tag{8}$$

$$S_2(D) = G_{12}(D)\hat{X}_1(D) \oplus G_{22}(D)\hat{X}_2(D) \oplus \hat{V}_2(D) \qquad (9)$$

where $\hat{X}(D)$ represents the corresponding received information bit sequence. For hard decoding, MTD-DR.Tp2 calculates a checksum value $L_j^{(x)}$ for decoding j-th information bit of the x-th information bit stream by

$$L_j^{(x)} = \sum_{s_{j,k}^{(xy)} \in \{\mathbf{S}_j^{(y)}\}} s_{j,k}^{(xy)} + \sum_{s_{j,k}^{(xy)} \in \{\mathbf{S}_j^{(y)}\}} s_{j,k}^{(xy)} + d_j^{(x)} \qquad (10)$$

where $\{\mathbf{S}_j^{(y)}\}$ is a checking syndrome set selected from the syndrome sequence $S_y(D)$ according to the generator polynomial $G_{xy}(D)$. The checking syndrome $s_{j,k}^{(xy)}$ is the k-th element in $\{\mathbf{S}_j^{(y)}\}$. The MTD.Tp2 calculates the checksum value by Eq. (10) in where $d_j^{(x)} = 0$ for every decoding stage. When the checksum value (for each bit in each information bit stream) exceeds the corresponding threshold value, the decoding decision is done accordingly. The threshold value for x-th information bit stream T_x is given by

$$T_x = \left\lfloor \frac{\sum_y J_{xy} + 1}{2} \right\rfloor \qquad (11)$$

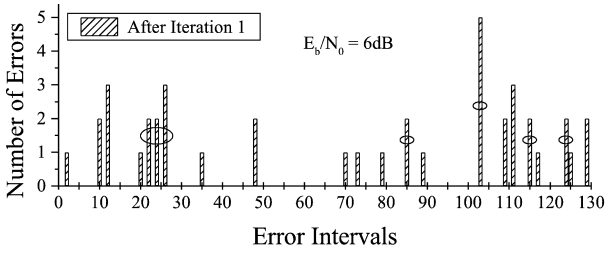
where J_{xy} is the number of terms in the generator polynomial $G_{xy}(D)$.

3. Error Grouping by MTD-DR

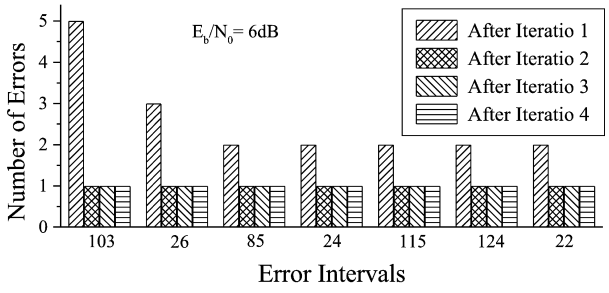
The MTD-DR, for SOCCs type 1, makes a strong error group in the decoded information bit stream. After many iterations, when errors in the decoded information bit stream make an error pattern equal to the tap connection pattern of an encoder, the MTD-DR cannot correct them.

Figure 4(a) shows the histogram of error intervals given by the errors in the decoded information bit stream after first iteration for SOCC type 1. The "error interval u" means that there exists $u-1$ error free bits between two error bits. Similarly, a tap connection interval τ_i is the distance between $(i+1)$-th and i-th tap connection position in the information register. The encircled error intervals, in this figure, are matched with the tap connection intervals of the encoder. Figure 4(b) shows that, after many iterations, the errors which are matched with the tap connection pattern cannot be corrected. That means, the MTD-DR.Tp1 makes a strong error group which depends on the tap connection patterns of the encoder of SOCCs type 1.

For improving error performance, a new type of code, called SOCC type 2, given in Sect. 2.2.2, is proposed. MTD-DR.Tp2 decodes each information bit by using two syndrome sequences. In this case, MTD-DR.Tp2 corrects some of the errors, which are matched with the tap connection pattern of the encoder, from each information bit stream due to two generator polynomials make different tap connection patterns for each information bit stream in the encoder. As a result, MTD-DR.Tp2 breaks down the error pattern which is matched with the tap connection pattern of the encoder of SOCCs type 2. Figures 5(a) and (b) show the histograms of



(a) Histogram of error intervals in the decoded information bit stream.



(b) Histogram of error intervals equal to the tap connection intervals. Tap connection intervals in the information register = {103,26,85,24,115,124,22}.

Fig. 4 Remaining error pattern for SOCCs type 1. The generator polynomial $G(D) = 1 + D^{103} + D^{129} + D^{214} + D^{238} + D^{353} + D^{477} + D^{499}$ and tap connection positions in the information register are {0, 103, 129, 214, 238, 353, 477, 499}.

error intervals which are matched with the tap connection intervals of encoder regarding to the decoded information bit stream 1 and 2, respectively. These figures illustrate that, after second iteration, almost all the errors which are matched with the tap connection pattern of the encoder are corrected. After iteration 3, all the errors in the group were corrected in this experiment. As a result, MTD-DR.Tp2 improves the error performance.

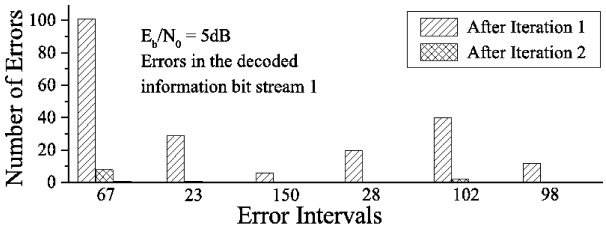
4. Performance of Hard Decoding MTDs

This section gives the comparative error performance of TD, MTD and MTD-DR for SOCCs type 1 and 2. In Table 1, $G(D)$ represents a generator polynomial of a SOCC type 1 with $J = 10$ and $K = 1000$. The generator polynomial $G_{xy}(D)$ represents a SOCC type 2 with $J_{xy} = 5$ and $K \approx 500$ (for shorter code) and $K \approx 10000$ (for longer code). These codes are used in this paper to get the error performance for each decoding scheme. Since the memory length of SOCC type 2 is about 500 and it uses two information bit streams, the SOCC type 1 will be a fair comparison with the memory length 1000.

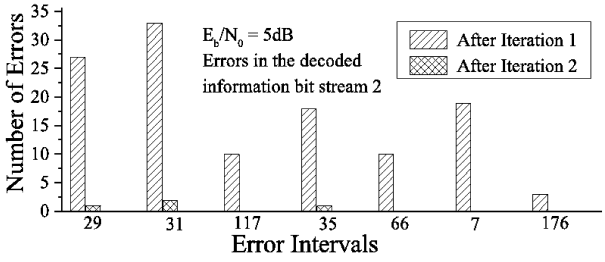
There thousands of SOCCs type 1 and 2 are found by computer search. For example, The code searching parameters are $K = 1000$ and $J = 10$ for SOCCs type 1 and that are $K = 500$ ($K = 10000$ for longer codes) and $\mathbf{J} = \{J_{11}, J_{12}, J_{21}, J_{22}\}$ for SOCCs type 2, where $J_{xy} = 5$.

Table 1 Generating polynomials for SOCCs: Type 1 and Type 2.

$G(D)$	$1 + D^{117} + D^{151} + D^{205} + D^{218} + D^{225} + D^{298} + D^{388} + D^{789} + D^{999}$	Type 1 K=1000
$G_{11}(D)$	$1 + D^{51} + D^{198} + D^{251} + D^{465}$	Type 2 $K \approx 500$
$G_{12}(D)$	$D^{23} + D^{187} + D^{247} + D^{370} + D^{371}$	
$G_{21}(D)$	$D^{40} + D^{76} + D^{176} + D^{200} + D^{259}$	
$G_{22}(D)$	$D^{161} + D^{230} + D^{281} + D^{328} + D^{483}$	
$G_{11}(D)$	$1 + D^{408} + D^{850} + D^{8286} + D^{9850}$	Type 2 $K \approx 10000$
$G_{12}(D)$	$D^{2341} + D^{3008} + D^{4167} + D^{4584} + D^{5339}$	
$G_{21}(D)$	$D^{780} + D^{2563} + D^{4716} + D^{9116} + D^{9718}$	
$G_{22}(D)$	$D^{4994} + D^{6152} + D^{6187} + D^{6390} + D^{6659}$	



(a) Histogram of error intervals equal to the tap connection intervals of encoder in the information register 1. Tap connection intervals of encoder in the information register 1 = {67,23,150,28,102,98}.



(b) Histogram of error intervals equal to the tap connection intervals of encoder in the information register 2. Tap connection intervals of encoder in the information register 2 = {29,31,117,35,66,7,176}.

Fig. 5 Remaining error pattern for SOCCs type 2. Generator polynomials are $G_{11}(D) = 1 + D^{90} + D^{268} + D^{370}$, $G_{12}(D) = 1 + D^{67} + D^{240} + D^{468}$, $G_{21}(D) = 1 + D^{60} + D^{212} + D^{285}$ and $G_{22}(D) = D^{29} + D^{177} + D^{278} + D^{461}$. Tap connection positions in the information register 1 and 2 are {0, 67, 90, 240, 268, 370, 468} and {0, 29, 60, 177, 212, 278, 285, 461}, respectively.

For finding SOCCs type 1, terms D^0 and D^{K-1} of the generator polynomial $G(D)$ are set as default terms. Remaining $(J-2)$ non-decreasing terms are generated randomly which are situated in D^0 to D^{K-1}. The generated code is checked whether it is a SOCC or not and the SOCC is selected.

For finding SOCCs type 2, four generator polynomials $(G_{11}(D), G_{12}(D), G_{21}(D), G_{22}(D))$ are generated randomly by using the parameters given above. In this case, all terms in each polynomial are generated randomly. Among them, at least one polynomial holds D^0 term and no terms in each polynomial exceeds the degree $K-1$. The generated polynomials make a code and the code is checked whether it is a SOCC or not and the SOCC is selected.

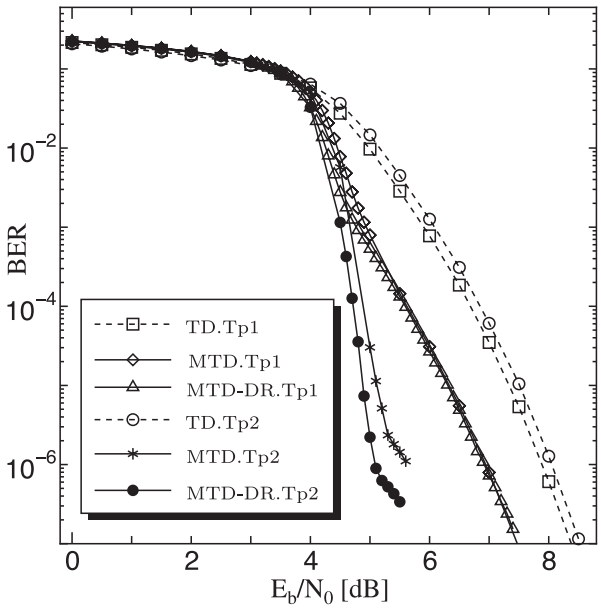


Fig. 6 Performance of MTDs with $N = 2100$, $J = 10$, $K = 1000$ for SOCC Type 1 and $J_{xy} = 5$, $K \approx 500$ for SOCC type 2.

For finding the best codes, generated SOCCs are driven to simulation with a fixed E_b/N_0 and bit error performance of them are observed and the best code is selected. Since J and K are the parameters of codes, this paper optimizes **J** (among $J_{xy} = 4$ to 6) for SOCCs type 2 and J (among $J = 8$ to 12) for SOCCs type 1 with certain K value. The tap weight $J_{xy} = 5$ ($J = 10$ for SOCCs type 1) gives the best BER among other codes with different J_{xy} values. Table 1 shows the best SOCC type 1 and 2 which are selected from 1000 codes. The best SOCC type 1 is found by setting $E_b/N_0 = 5.5$ dB. The best SOCCs type 2 with $J_{xy} = 5$ and $K \approx 500$ (shorter code) and $K \approx 10000$ (longer code) are found by setting $E_b/N_0 = 5.0$ dB and $E_b/N_0 = 4.0$ dB, respectively. This paper searches the best SOCCs around the bit error rate 10^{-5}.

The simulation is done by setting total information length $N = 2100$ bits for shorter codes and $N = 40000$ bits for longer codes in this paper. Figure 6 shows the hard decoding bit error performance of TD, MTD and MTD-DR with SOCCs type 1 and 2 for shorter codes. MTD gives better bit error performance than that of TD, for both types of codes. MTD-DR achieves 1.4 dB and 2.6 dB more coding gain than that of TD at the BER 10^{-5} for SOCC type 1 and 2, respectively. Since, DR can help to calculate the Hamming distance, between the decoded codeword and a received word, MTD-DR gives better error performance than MTD. The MTD-DR.Tp2 achieves 4.7 dB coding gain at the BER 10^{-5}. Furthermore, the DR improves error flooring effect significantly and elevates overall error performance.

Figure 7 illustrates the average number of iterations for decoding both type of codes by MTD and MTD-DR. When no information bit is flipped, MTDs terminate their decoding, regardless the DR and their decoding algorithms. If

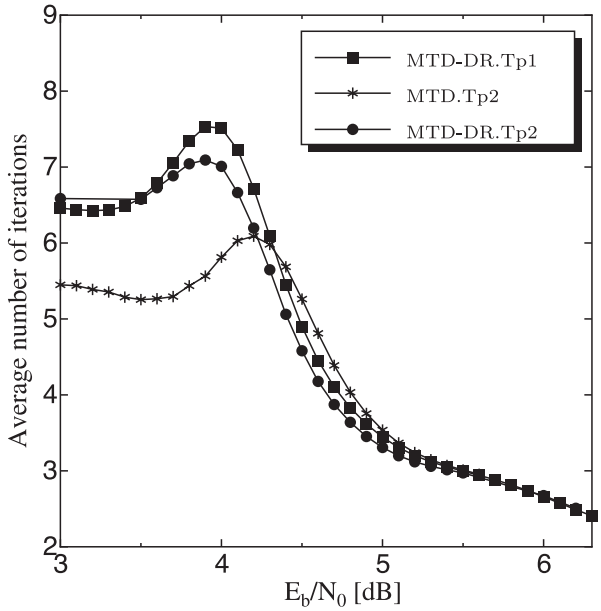


Fig. 7 Average number of iterations for hard decoding MTD-DR.Tp1, MTD.Tp2 and MTD-DR.Tp2 with the codes in Table 1.

the termination condition does not fulfil, MTDs terminate their decoding by the maximum number of iterations. It is necessary to set sufficiently large maximum iterations for extracting maximum benefit. In this case 30 iterations is sufficiently large. It is seen that, at a particular E_b/N_0 region, the average number of iterations increase unexpectedly. Authors have not yet known the reasons why it occurs. To achieve the BER 10^{-5}, MTD-DR.Tp1 expends E_b/N_0 = 6.2 dB and 2.6 average number of iterations whereas MTD-DR.Tp2 and MTD.Tp2 expend 5.1 dB and 5.3 dB E_b/N_0 with the average number of iterations 3.2 and 3.3, respectively.

In terms of bit error performance, MTDs for SOCCs type 2 are better than TD (for both types of codes) as well as MTDs for SOCCs type 1. Therefore, this paper compares the error performance of MTD-DR for SOCCs type 2 among hard and soft decoding algorithms hereafter.

5. Soft Decoding MTDs

5.1 Soft MTD-DR

For decoding j-th information bit by the soft decoding MTD-DR (SMTD), the j-th checksum value L_j is calculated by

$$L_j = \sum_{s_{j,k} \in \{\mathbf{S_j}\}} w_{j,k}(1 - 2s_{j,k}) + w_{dj}(1 - 2d_j) \tag{12}$$

where $s_{j,k}$ is the k-th bit of a syndrome set $\{\mathbf{S}_j\}$ and d_j is the j-th bit in DR. The $w_{j,k}$ is an absolute value of received parity signal related to the syndrome bit $s_{j,k}$. The value w_{dj} is an absolute value of j-th received information signal. When $L_j < 0$, flipping decision is done. After flipping each information bit, the Euclidian distance between the decoded codeword and a received word becomes closer. (proof is given below).

In Eq. (12), $s_{j,k}$ and d_j are binary valued. It defines $r_{j,k\neq 0} \triangleq (1 - 2s_{j,k})$ and $r_{j,k=0} \triangleq (1 - 2d_j)$, $\forall r \in \pm 1$ and then, Eq. (12) is simplified to

$$L_j = \sum_{k=0}^{J} \gamma_{j,k} r_{j,k} \tag{13}$$

where $\gamma_{j,k\neq 0} = w_{j,k}$ and $\gamma_{j,k=0} = w_{dj}$. The squared Euclidian distance between a received word and the decoded codeword is the summation of the squared Euclidian distance between the parity parts and the information parts of them. Let y be a received signal and r is the antipodal representation of that decoded information bit. Then the squared Euclidian distance between them can be defined by

$$E_d^2 = (y - \rho r)^2 = y^2 + \rho^2 r^2 - 2\rho y r \tag{14}$$

where ρ is the channel gain. When the term yr in Eq. (14) increases, the squared Euclidian distance decreases. If sign(y) = sign(r), the DR's value $(1 - 2d)$ or the syndrome register's value $(1 - 2s)$ will be '1,' otherwise they will be '-1.' Then the term yr can be represented by

$$yr = |y|(1 - 2d) \qquad \text{or} \qquad yr = |y|(1 - 2s) \tag{15}$$

For decoding j-th information bit, the partial squared Euclidian distance $E_{d_j}^2$ can be calculated by

$$\begin{aligned} E_{d_j}^2 &= \sum_{k=0}^{J} \{y_{j,k}^2 + \rho^2 (r_{j,k})^2\} - 2\rho \sum_{k=0}^{J} |y_{j,k}| r_{j,k} \\ &= \sum_{k=0}^{J} \gamma_{j,k}^2 + \rho^2 (J+1) - 2\rho \sum_{k=0}^{J} \gamma_{j,k} r_{j,k} \\ &= \sum_{k=0}^{J} \gamma_{j,k}^2 + \rho^2 (J+1) - 2\rho L_j \end{aligned} \tag{16}$$

where $y_{j,k=0}$ is the value of j-th received information signal and $y_{j,k\neq 0}$ is the value of a received parity signal regarding to the syndrome bit $s_{j,k}$. In this case, $\gamma_{j,k}$ and $|y_{j,k}|$ are identical. When $L_j < 0$ (in Eq. (16)), the flipping decision turns L_j from negative value to positive and the Euclidian distance reduces.

5.2 Weighted Bit Flipping MTD-DR

The weighted bit flipping (WBF) algorithm is proposed for decoding LDPC codes [10]. This paper formulates a WBF algorithm for MTD-DR called weighted bit flipping MTD-DR (WBF.MTD). It is known that each syndrome bit is derived from $(J + 1)$ received signals. The weighting value $w_{j,k}$ of WBF.MTD is the minimum absolute value among the received signals regarding to the syndrome bit $s_{j,k}$. The weighting value $w_{j,k}$ is put into Eq. (12) and the decoding decision is done accordingly.

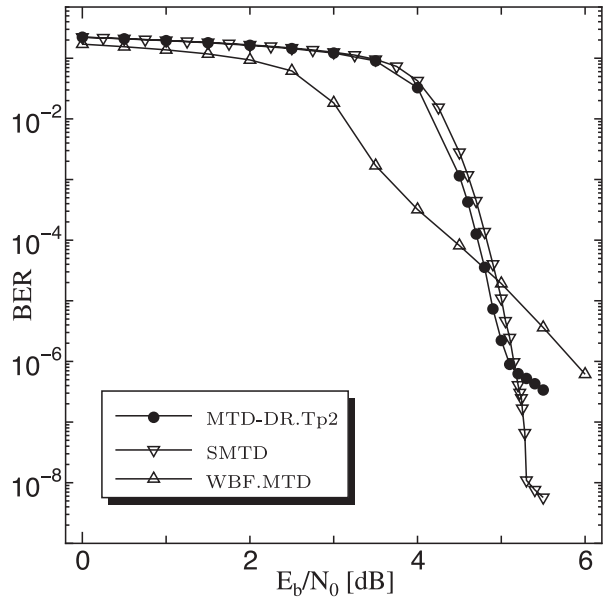


Fig. 8 Performance of hard and soft decoding MTDs for SOCC Type 2 with $J_{xy} = 5$, $K \approx 500$, $N = 2100$.

5.3 Performance of Soft Decoding MTDs

Figure 8 illustrates the bit error performance of hard and soft decoding MTDs for a SOCC type 2 with $K \approx 500$ and $J_{xy} = 5$. The bit error performance of SMTD and MTD-DR.Tp2 are superior than that of WBF.MTD at the BER less than 10^{-5}. The error flooring effect of SMTD is better than that of MTD-DR.Tp2 and WBF.MTD. The WBF.MTD gives better BER at the lower E_b/N_0 region. By observing the error performance of SMTD and WBF.MTD, this paper proposes two new kinds of soft decoding algorithms called combined soft decoding algorithms for MTD-DR which are discussed in the next section.

6. Combined Soft Decoding MTDs

Combined soft decoding MTD with DR (CMTD) is a serial concatenation of WBF.MTD and SMTD. Two types of combined soft decoding MTDs are given:

1. Combined soft decoding MTD-DR without feedback (CMTD.NFB)
2. Combined soft decoding MTD-DR with feedback (CMTD.Feed)

6.1 CMTD without Feedback

Figure 9(a) shows a schematic diagram of CMTD.NFB. In this case WBF.MTD works first. After some iterations (maximum Γ_w iterations), WBF.MTD terminates its decoding. Then SMTD starts its decoding. After some iterations (maximum Γ_s iterations), SMTD terminates it decoding and final output is done. Since, WBF.MTD and SMTD give better error performance at lower and higher E_b/N_0, respectively, the CMTD.NFB may improves overall decoding performance. For CMTD.NFB case, it sets sufficiently large number of iterations such as $\Gamma_w = \Gamma_s = 30$.

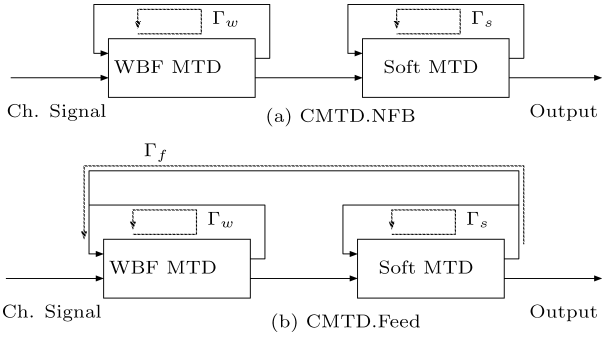


Fig. 9 Schematic diagrams of (a) CMTD without Feedback and (b) CMTD with feedback systems.

6.2 CMTD with Feedback

Figure 9(b) shows a schematic diagram of CMTD.Feed. In this structure, CMTD.Feed first decodes the information bit streams just like as CMTD.NFB by a few number of maximum iterations, due to reduce decoding complexity, and then feedback to decode again. The feedback decoding may correct again some bits and gives better error performance. When both component decoders satisfy their termination condition simultaneously, feedback does not continue and then the CMTD.Feed gives final output. In this case, outer iteration, Γ_f, and inner iterations, Γ_w and Γ_s, highly influence the complexity of decoding. When it sets the inner iterations more than 2 and outer iterations more than 10, CMTD.Feed does not save E_b/N_0 more than 0.05 dB at water fall region and at error floor region the improvement is zero. So, the outer maximum iteration $\Gamma_f = 10$ and inner maximum iterations $\Gamma_w = \Gamma_s = 2$ may be considered as the sufficient setting of iterations.

6.3 Performance of Combined Soft Decoding MTDs

Figure 10 shows the BER performance of soft decoding and combined soft decoding MTDs. CMTDs successfully use the advantages of WBF.MTD and SMTD together and give better BER performance. CMTD.NFB achieves 2.6 dB and 3.1 dB more coding gain than that of SMTD and WBF.MTD, respectively at the BER 10^{-5}. In addition, CMTD.Feed gives the best BER performance among other MTDs. At the BER 10^{-5}, CMTD.Feed achieves 0.4 dB more coding gain than that of CMTD.NFB and achieves coding gain of 6.5 dB over the AWGN channel. Though, the error flooring of CMTD.Feed starts at the earlier BER than that of CMTD.NFB, their error performance converge at the higher E_b/N_0 region. The dotted line shows the slope of estimated error performance of maximum likelihood decoding [11] considering the minimum Hamming distance term only. The minimum Hamming distance for systematic SOCCs type 1 is $(J + 1)$ [8] and it is $\min_x (\sum_y J_{x,y} + 1)$ for

SOCCs type 2. The error flooring slopes of all MTDs follow the theoretical error performance curve at higher E_b/N_0 region. This error flooring occurs due to minimum Hamming distance of codes.

Figure 11 shows the average number of iterations for soft decoding MTDs. For CMTDs, the average numbers of iterations are the summation of the average numbers of iterations of each component decoder. The CMTD.NFB gives better BER than that of SMTD as well as WBF.MTD by a little more average number of iterations of them. For achieving the BER 10^{-5}, CMTD.NFB demands E_b/N_0 = 3.5 dB with the average number of iterations 7.5 whereas CMTD.Feed demands 3.1 dB with 35.0 average number of iterations. Therefore, the CMTD.NFB may be useful for faster decoding and the CMTD.Feed may be suitable for the lower E_b/N_0 transmission.

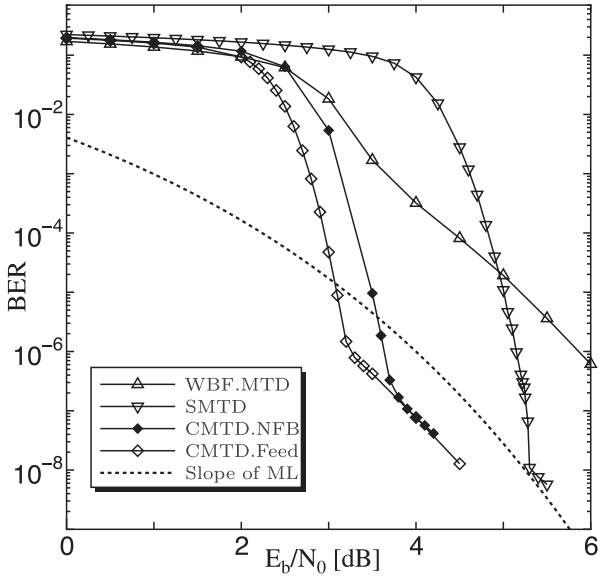


Fig. 10 Performance of CMTDs for SOCC Type 2 with $J_{xy} = 5$, $K \approx 500$, $N = 2100$, ML = Maximum Likelihood Decoding.

7. CMTDs with Parity Check Codes

7.1 Serial Concatenation with Parity Check Codes

For further improving overall error performance, parity check codes are concatenated with CMTD.Feed (or with CMTD.NFB), in short CMTD.Feed.Parity (or CMTD.NFB.Parity). Before encoding for SOCCs, information bit streams are segmented by the predefine size. Each segment of the information bit stream is defined as an information sub-block. A parity check encoder generates the parity check codes against each information sub-block and then the encoder of SOCC (in Fig. 3) generates a codeword. The received signals are decoded by the CMTD first and then the parity check decoder decodes for the final output. When a decoded information sub-block does not satisfy the parity check, the parity check decoder finds the minimum checksum value L_i in the information sub-block and then the i-th information bit of that sub-block is flipped and the final output is done.

7.2 Performance of CMTDs with Parity Check Codes

Figure 12 shows the BER performance of CMTD.Feed, CMTD.NFB, CMTD.Feed.Parity and CMTD.NFB.Parity. For finding these error performance, information sub-block length is set to 50 bits. Therefore, CMTD.Feed.Parity as well as CMTD.NFB.Parity give the coding rate 0.4901 <

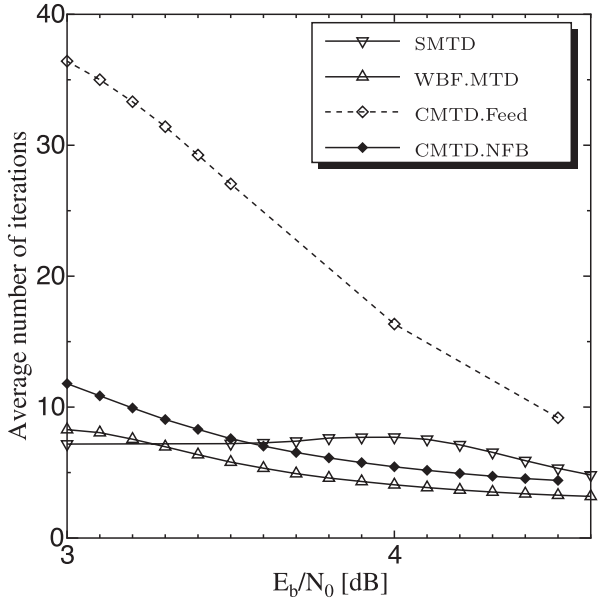


Fig. 11 Average number of iterations for CMTD.Feed and CMTD.NFB for SOCC Type 2 with $J_{xy} = 5$, $K \approx 500$, $N = 2100$.

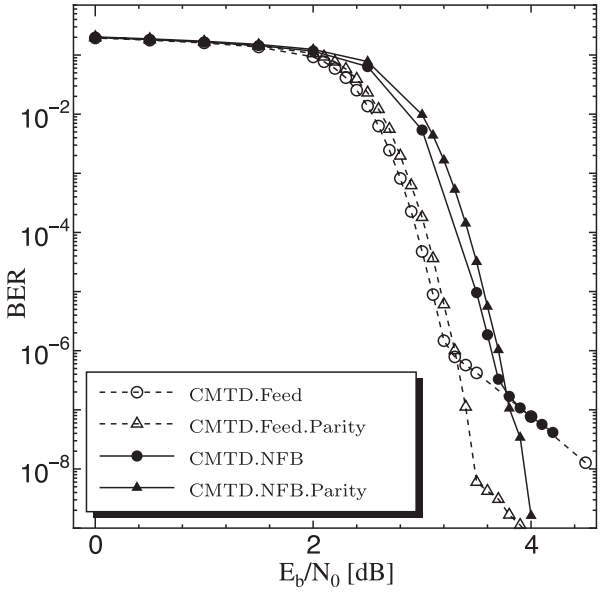


Fig. 12 Performance of CMTDs with parity check codes for SOCC type 2, $J_{xy} = 5$, $K \approx 500$, $N = 2100$.

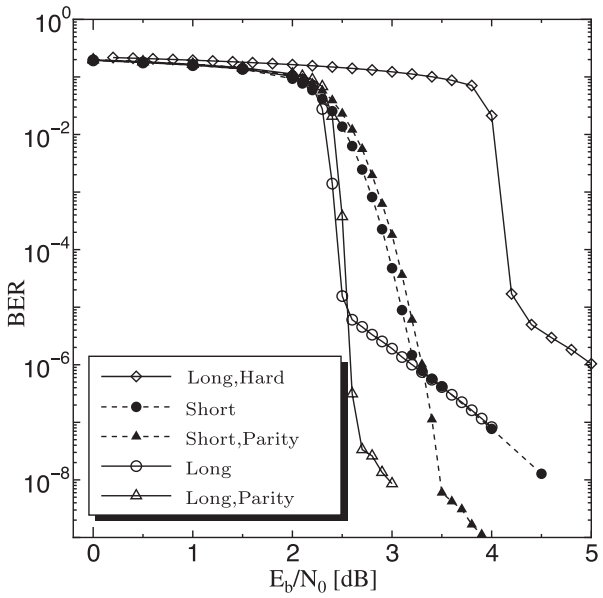


Fig. 13 Performance of hard decoding MTD-DR for the longer code as well as CMTD.Feed and CMTD.Feed.Parity for the shorter and longer codes (in Table 1).

0.5. Since the parity check decoding is not iterative, the average number of iterations for CMTD.Feed (or CMTD.NFB) and CMTD.Feed.Parity (or CMTD.NFB.-Parity) are the same. The bit error performance of CMTD.Feed and CMTD.Feed.Parity are similar at the lower E_b/N_0 region. The CMTD.Feed.Parity prevents error flooring effect and gives better BER performance at the high E_b/N_0 region. Similar incident is occurred between CMTD.NFB and CMTD.NFB.Parity also. So, the parity check code with CMTDs improves overall error performance without increasing the average number of iterations.

7.3 Performance of Longer Memory Length Code

Figure 13 gives the BER performance of SOCCs type 2 with memory length $K \approx 500$ and $K \approx 10000$ for CMTD.Feed and CMTD.Feed.Parity decoding schemes. Hard decoding performance of the longer code is also given. The hard decoding MTD-DR for longer code achieves 5.3 dB coding gain at the BER 10^{-5} over the AWGN channel. The CMTD.Feed with $K \approx 10000$ achieves 0.6 dB more coding gain than that of the CMTD.Feed with $K \approx 500$ at the BER 10^{-5}. The CMTD.Feed.Parity (for the longer code) achieves 7.1 dB coding gain over the AWGN channel at the BER 10^{-5} with 40 average number of iterations.

8. Conclusion

This paper has successfully reconstructed MTDs with coding gain 7.1 dB over the AWGN channel at the BER 10^{-5} for a longer code. MTDs may be considered as a low complex decoding method with excellent coding gain. MTDs give limited error performance with the SOCC type 1 due to make an irreducible error group at decoding. For SOCCs type 2, MTDs prevent to form that error group at decoding and give better error performance. The DR in MTD-DR improves overall error performance. So, the MTD-DR for SOCCs type 2 may consider as an important decoding method in the field of threshold decoding.

The hard decoding MTD-DR gives 4.7 dB coding gain over the AWGN channel at the BER 10^{-5} with 3.8 average number of iterations for shorter codes. The MTD-DR for longer codes gives 5.3 dB coding gain at the same BER. In addition, the CMTD.Feed achieves 1.8 dB more coding gain than that of the hard decoding MTD-DR at the BER 10^{-5}. The CMTD.NFB gives coding gain 6.1 dB for shorter codes at the BER 10^{-5} with the average number of iterations 7.5 whereas the CMTD.Feed achieves 6.5 dB coding gain with 35.0 average number of iterations. Therefore, the CMTD.Feed may be considered as an E_b/N_0 conscious decoding method and the CMTD.NFB can decode with the lower average number of iterations. However, CMTDs experience error flooring effect from the earlier BER. The serial concatenation of parity check codes with CMTDs improves overall error performance and gives reliable decoding for the high speed communication.

References

[1] J. Massey, Threshold Decoding, MIT Press, 1963.
[2] J. Massey, "Catastrophic error-propagation in convolutional codes," Circuit Theory — 11th Midwest Symposium, University of Notre Dame, pp.583–587, 1968.
[3] J.P. Robinson, "Error propagation and definite decoding of convolutional codes," IEEE Trans. Inf. Theory, vol.IT-14, no.1, pp.121–128, Jan. 1968.
[4] V.V. Zolotarev and G.V. Ovechkin, "An effective algorithm of noise-proof coding for digital communication systems," Electrosvaz, no.9, pp.34–36, 2003.
[5] V.V. Zolotarev, "The multithreshold decoder performance in Gaussian channels," 7th Int. Symp. on Commn. Theory and Applications (ISCTA'03), pp.18–22, UK, July 2003.
[6] V.V. Zolotarev, Theory and Algorithms of Multithreshold Decoding, Under Scientific Edition of the Member Correspondent of the Russian Academy of Science, U.B. Zubarev, Moscow, Radio and Communications, Hot Line Telecom, 2006 (in Russian).
[7] http://www.mtdbest.iki.rssi.ru
[8] S. Lin and D.J. Costello, Jr., "Majority-logic decoding of convolutional codes," in Error Control Coding: Fundamentals and Applications, chap.13, Prentice-Hall, Englewood Cliffs, N.J., 1983.
[9] C. Cardinal, D. Haccoun, and F. Gagnon, "Iterative threshold decoding without interleaving for convolutional self-doubly orthogonal codes," IEEE Trans. Commun., vol.51, no.8, pp.1274–1282, Aug. 2003.
[10] Y. Kou, S. Lin, and M.P.C. Fossorier, "Low-density parity-check codes based on finite geometries: A rediscovery and new results," IEEE Trans. Inf. Theory, vol.47, no.7, pp.2711–2736, Nov. 2001.
[11] A.J. Viterbi and J.K. Omura, Principles of Digital Communication and Coding, McGraw-Hill Book Co., 1985.

Muhammad Ahsan Ullah was born in Jamalpur, Bangladesh on March 1, 1979. He received B.E. degree in Electrical and Electronic Engineering from Chittagong University of Engineering and Technology, Bangladesh in 2002 and M.E. degree in Electronic and Information Engineering from Kyung Hee University, Republic of Korea in 2007. Now, he is pursuing doctoral degree in Electrical Engineering at Nagaoka University of Technology from September, 2008. He is interested in convolutional codes and multi-stage threshold decoding.

Kazuma Okada was born in Okayama Prefecture, Japan on March 14, 1986. He received B.E. degree in Electrical Engineering from Nagaoka University of Technology in 2008, and received M.E. degree in Electrical Engineering from Nagaoka University of Technology in 2010. In his master course, he had interested in convolutional codes and multi-stage threshold decoding.

Haruo Ogiwara was born in Tochigi Prefecture, Japan on February 4, 1947. He received B.E. and M.E. in Control Engineering from Tokyo Institute of Technology in 1969 and 1971, respectively, and Dr.Eng. in Electronics from Osaka University in 1984. Since 1971 to 1986, he was a research engineer at Electrical Communication Laboratories of Nippon Telegraph and Telephone corporation, where he worked on researches and developments of Hologram memories, optical switching system, a digital subscriber loop, and communication theory. In 1986 he joined the Nagaoka University of Technology and he is now Professor in Department of Electrical Engineering. His current research interest includes turbo code, coded modulation especially for a non-Gaussian channel and a fading channel and adaptive equalization in digital mobile communication. He is a member of IEEE and SITA.