Middle-East Journal of Scientific Research 24 (S1): 278-285, 2016
ISSN 1990-9233
© IDOSI Publications, 2016
DOI: 10.5829/idosi.mejsr.2016.24.S1.58

Effect of Interleaver Algorithm on the Performance of Modified Turbo Codes

[1]I. Poonguzhali and [2]C. Arun

[1]Department of Electronics and Communication Engineering,
Panimalar Institute of Technology , Chennai, India
[2]Department of Computer Science and Engineering , RMK Engineering College, Chennai, India

Abstract: Turbo Convolutional Codes (TCC) are the most powerful error-correcting codes that can approach the Shannon limit through iterative decoding process and an efficient interleaver. Each decoding iteration results in an increase of computation and decoding delay. Therefore, this gives rise to a need for low complexity codes called Modified Turbo Codes (MTC). The performance of MTC is close to TCC with reduced decoding complexity. MTC encode information bits arranged in 2-dimensional array by using 2-stage interleavers. This paper proposes two algorithms, viz. Proposed Interleaver Algorithm 1 (PIA-1) and Proposed Interleaver Algorithm 2 (PIA-2) for the design of 2-stage interleavers. Quadratic Permutation Polynomial (QPP) interleaver adopted by 3GPP LTE standard is used in PIA-2. Simulation results show that Bit Error Rate (BER) performance and error convergence of MTC with PIA-1 and PIA-2 is better than that of Existing Interleaver Algorithm (EIA). From the BER comparison, it is observed that the performance of MTC with PIA-2 is almost equivalent to TCC with QPP interleaver but with reduced decoding complexity.

Key words: Bit Error Rate · Decoding Complexity · Modified Turbo Code · Multiple interleaver · QPP interleaver

INTRODUCTION

Since the introduction of Turbo Convolutional Codes (TCC) in 1993[1], they have received considerable attention as its performance is near the Shannon capacity limit [2]. TCC consists of an interleaver which separates the two parallel concatenated convolutional codes. The complexity involved in decoding structure of TCC hinders its usage. Therefore the necessity for low complexity gives rise to a class of Modified Turbo Codes (MTC) called Low Complexity Hybrid Turbo Codes (LCHTC) and Improved Low Complexity Hybrid Turbo Codes (ILCHTC), which is a parallel combination of component codes i.e., zigzag codes and convolutional codes[3,4]. However, simulation results show that BER of LCHTC & ILCHTC are comparable to that of TCC with ILCHTC having better error convergence than LCHTC with comparable decoding complexity [3,4,5]. LCHTC & ILCHTC has 50% less decoding complexity than TCC [5].

MTC encodes information bits arranged in two dimensional arrays, but TCC encodes information bits arranged in one dimensional array. Therefore, the interleavers designed for TCC are not suitable for MTC as they require row or column spread and dispersion of interleaved bits for 2D information array. For better error performance an interleaver with high values of spreading factor and dispersion is required [6]. MTC uses multiple 2-dimensional interleavers as they encode bits arranged in two dimensional arrays. Therefore it is an essential factor to design 2-dimensional interleaver. In this paper two different 2-dimensional interleaver algorithms are proposed and BER of TCC and MTC are compared. Simulation results shows that for the proposed interleaver algorithms BER of 10^{-5} is achieved with less decoding complexity than the existing interleaver algorithm. This paper is organized as follows. In section II description of MTC is presented. Multiple and QPP interleavers are explained in section III. Simulation results are presented and analyzed in section IV. In section V conclusion is outlined.

Description of MTC

MTC Encoder: MTC is a parallel concatenation of component codes. A combination of zigzag and

Corresponding Author: I. Poonguzhali, Department of Electronics and Communication Engineering,
Panimalar Institute of Technology, Chennai, India.

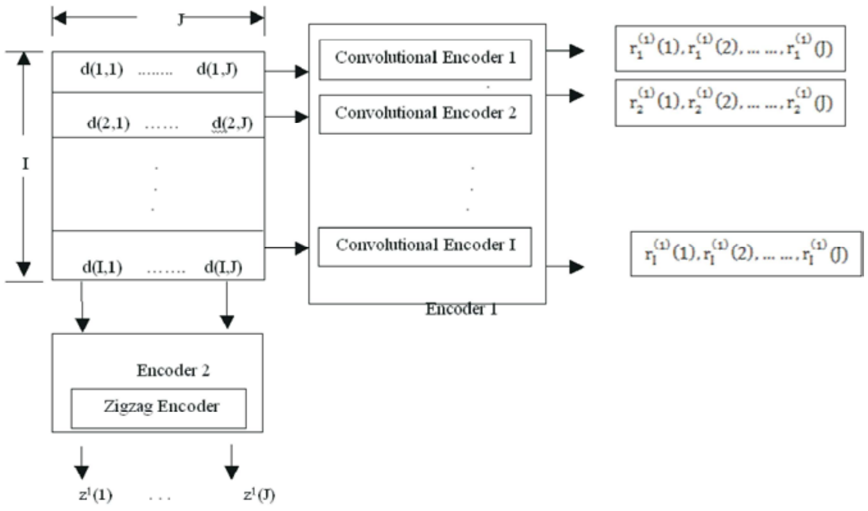

Fig. 1: First Constituent Encoder of a Modified Turbo Code

convolutional codes are used as component codes. A good compromise between complexity and error performance is provided by MTC [4].

MTC is constructed by arranging a sequence of N information bits in an array of size I x J [5]. Let two dimensional information arrays d, be given by

d={d(i,j)}, 1 = i = I and 1 = j = J. (1)

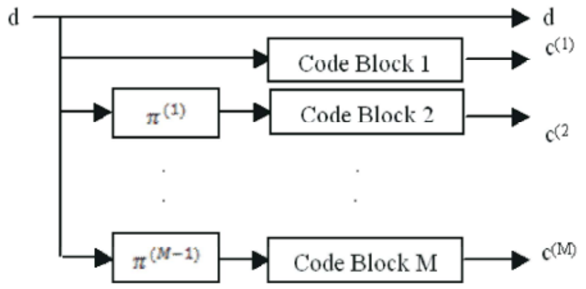

Fig. 2: General Encoder structure of a MTC

The first constituent encoder of a MTC is shown in Fig. 1. Row bits are encoded by encoder 1 and column bits are encoded by encoder 2. The encoder 1 is a convolutional encoder and encoder 2 is a zigzag encoder. The encoder structure of a general MTC consists of parallel concatenations of M constituent codes and M-1 interleavers as shown in Fig. 2. LCHTC and ILCHTC are types of MTC.

ILCHTC Encoder: The modified version of LCHTC is ILCHTC. Information bits are encoded by (i) only zigzag codes in LCHTC and (ii) both zigzag and convolutional codes in ILCHTC. Therefore, ILCHTC's error convergence is better than that of LCHTC. In ILCHTC, first L rows of information array bits are encoded using rate 1/2 RSC code (encoder 1). Then for each column of information array the zigzag parity bits are computed (encoder 2). The other constituent encoders are zigzag encoders which is same as that of LCHTC. The encoder structure of ILCHTC consists of parallel concatenation of M constituent codes and M-1 interleavers as shown in Fig. 3 with L=I for first constituent encoder and L=0 for other constituent encoders.

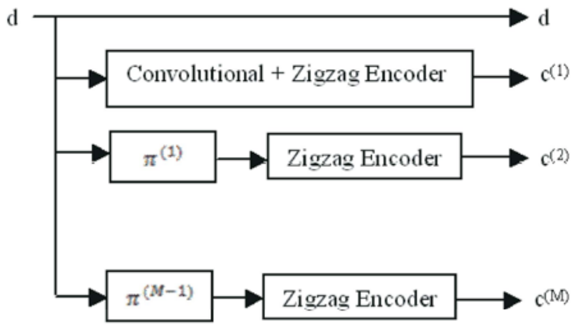

Fig. 3: ILCHTC Encoder

The code word C_{IL} and code rate R_{IL} for ILCHTC is given by

$$C_{IL} = \{d, c^{(1)}, c^{(2)}, \ldots\ldots\ldots, c^{(M)}\}. \tag{2}$$

$$C_{IL} = \left\{d, r_1^{(1)}, r_2^{(1)}, \ldots, r_L^{(1)}, z^{(1)}, z^{(2)}, \ldots, z^{(M)}\right\}. \tag{3}$$

$$R_{IL} = J/(J + L + M) \tag{4}$$

Where $r_L^{(1)}$ is the convolutional parity vector for the Lth row of information array in first constituent encoder and $z^{(M)}$ be the zigzag parity vector of Mth zigzag encoder for each column of information array? The encoders and decoders used in ILCHTC are described below.

Zigzag Encoder: Zigzag encoder is encoder 2 which computes parity bits for each column of information bits as in Fig. 4. Let $z^{(m)}$ be the zigzag parity vector for the mth constituent encoder, where

$$z^{(m)} = \{z^{(m)}(j)\},\ 1 = j = J \text{ and } 1 = m = M. \quad (5)$$

The zigzag parity bits for the information bits arranged in an array of size I x J can be computed progressively by taking even parity of data bit in each column and parity bit of the previous column as given below [7].

$$z^{(m)}(0) = 0 \quad (6)$$

$$z^{(m)}(j) = [\textstyle\sum_{i=1}^{I} d(i,j) + z^{(m)}(j-1)],1 = j = J. \quad (7)$$

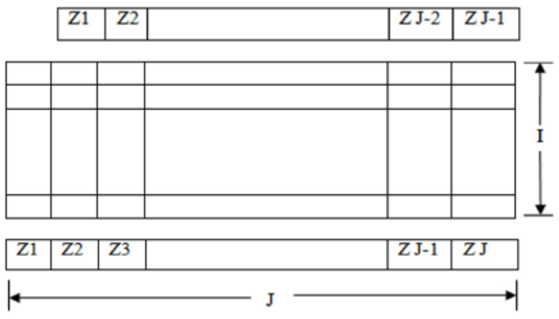

Fig. 4: Zigzag Encoder

Here, XOR operation on the information bit is represented by summation. For concatenated zigzag code with M constituent encoders the code word c_z and code rate, R_z are given by

$$c_z = \{d, z^{(1)}, z^{(2)}, \ldots\ldots, z^{(m)}\}. \quad (8)$$

$$R_s = J/(J+M) \quad (9)$$

Convolutional Code: A rate ½ Recursive Systematic Convolutional (RSC) code is encoder 1 which computes parity bits for each row of information bits as in Fig. 5. Let $r_I^{(1)}$ be the convolutional parity vector for the Ith row of information bits, where

$$r_I^{(1)} = \{d, r_i^{(1)}(j)\},\ 1 \le i \le I. \quad (10)$$

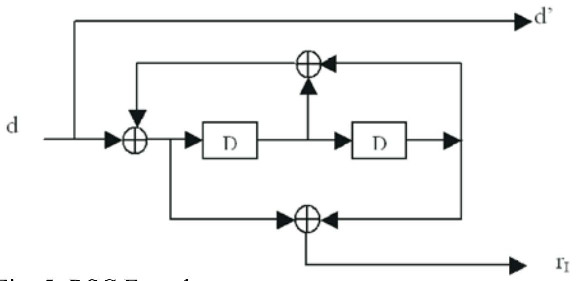

Fig. 5: RSC Encoder

ILCHTC Decoder: Let the transmitted binary codeword c be with values in {-1, 1}. The transmitted code vector is denoted by c = {d, r, z} and the received code vector is denoted by $\tilde{c} = \{\tilde{d}, \tilde{r}, \tilde{z}\}$ where $\tilde{d}, \tilde{r}, \tilde{z}$ are the received information vector, received convolutional parity vector and the received zigzag parity vector respectively [4].

The received data bits are partitioned into blocks of size IxJ and their corresponding convolutional parity bits (a priori values of LLR) are identified. Then the decoding process starts as follows in the first decoder.

- Each L-rows of the array are decoded using a priori LLRs as input to SISO convolutional decoder to produce a posteriori LLR as output. Each convolutional decoder is accomplished by using Max-Log-MAP algorithm [8, 9]. The computational complexity of Max-Log-MAP algorithm is less when compared to the MAP algorithm.
- Zigzag decoding of each column of array is performed by taking the results of step (a) as a priori LLRs.

The overall decoding is completed by iterative decoding strategy in which steps (a) and (b) are repeated in first constituent decoder and step (b) is repeated in other constituent decoders [9]. The iterative decoder consists of M decoding modules. Each decoder has 2 inputs viz. i) LLR of parity of zigzag code and ii) A Posteriori Probability (APP) of information bit which is updated iteratively. Also by subtracting the extrinsic information of previous iteration, the extrinsic information is prevented from circulating back to its generator as suggested in [1].

Max* operator is used to reduce decoding complexity in turbo codes [10]. It is defined as

$$\max{}^*(x,y) = \ln(e^x+e^y) = \max(x,y) + \ln(1+e^{(-|x-y|)}). \quad (11)$$

Let C_a and C_m be the number of Addition and Multiplication Equivalent Operation (AEO and MEO) required to implement max* in Log-MAP and Max-Log MAP. The number of computations required to implement max* operator in Log-MAP is given by $C_{a,max} = 2$ and $C_{m,max}=2$ by neglecting the operation for maximum value selection and assuming logarithm and exponential values are equivalent to multiplication operations.
For Max-Log MAP algorithm, max* is approximated as

$$\max{}^*(x,y) \sim \max(x,y). \quad (12)$$

The number of computations required to implement max* operator is given by $C_{a,max} = 1$ and $C_{m,max}=0$ by assuming that the number of comparisons are equivalent to addition operations.

Let $C_{a,IL}$ and $C_{m,II}$ be the number of AEO per Information Bit per Iteration (AEO/IB/I) and MEO per Information Bit per Iteration (MEO/IB/I) required by ILCHTC decoder [10]. The decoding complexity of ILCHTC decoder is given by,

$$C_{a,II} = (L/J)[(4S_t - 2) C_{a,max} + 12 S_t + 1] + C_{a,zc} + M \quad (13)$$

$$C_{m,II} = [(L/J) \times (4S_t - 2) \times C_{a,max}] + 1 + C_{a,zc} + M \quad (14)$$

Where,
$C_{a,ZC}= M[2J+2+(2J+2)C_{a,max}]/J$ and $C_{m,ZC}=M[(2J+2)C_{m,max}]/J$

Table 1:

		Log-MAP		Max-Log-MAP	
Code	Parameter	C_m	C_a	C_m	C_a
TCC	M=2,S_t=4	56	157	0	129
ILCHTC	M=3,J=4,L=4, S_t=4	44	103	0	81
	M=2,J=4,L=2, S_t=4	25	56	0	44

The decoding complexity is computed for Log-MAP and Max-Log-MAP algorithms and is given in table I.

Multiple and Qpp Interleaver: Since there are M constituent encoders in MTC, M-1 interleavers are required except for the first one. In this paper two 2-stage interleaver algorithms (Proposed Interleaver Algorithm–1, Proposed Interleaver Algorithm–2) are proposed and their comparison with the existing 2-stage interleaver is given. An interleaver has to spread either column or row positions of information bits as the constituent encoders used in MTC encode either columns or rows of an information array [3, 4]. The spreading factor S, of an interleaver with 2-dimensional array of size IxJ can be obtained by considering both uninterleaved and interleaved information bits of same size. The spread of π is the largest integer S such that $|p-q|<S \Rightarrow |\pi(p)-\pi(q)|=S$, for 1=S=K, where p, q and $\pi(p),\pi(q)$ are column positions of information bits before and after interleaving respectively, such that p?q.

Design of Two-Stage Multiple Interleavers: The proposed method changes the rule used to scramble the information array in MTC [11]. Figure 6 shows the steps to design the two stage interleaver.

Proposed Interleaver Algorithm-1: The uninterleaved N bits are arranged in an array A of size I x J and array A is divided into K blocks, each of size I x U bits. For N=1536, I=4, U=4 and number of blocks K =96 (i.e. N=K x I x U).

The algorithm for the first stage of the interleaver is as follows:

Step 1: Initialize two arrays, B and C with the same size and number of blocks as that of A.

Step 2: Replace all the elements of K blocks in the array B by b. Starting from b=1, repeat this for all values of b (1¡Üb¡ÜK).

Step 3: The first element of the odd blocks in the array B is grouped together into blocks with the same size of I x U and stored in array C, followed by the blocks formed by grouping the first elements of the even blocks.

Step 4: Repeat Step 3 for all the elements in each block.

QPP Interleaver: The QPP Interleaver used for generating pseudo random number sequences is adopted by 3GPP LTE standard due to its simple formula and outstanding performance like avoidance of interleaver table storage in a turbo decoder and excellent reduction of decoding latency of parallel turbo decoding by maximum-contention-free property [12,13]. For an information block of size N, the QPP interleaver is specified by

$$F(x) = f_2x^2+f_2x \quad \text{Mod N}. \quad (15)$$

where, x is the original address, F(x) is the interleaved address, the f_1 and f_2 are integers and depends on the block size N ($0 < x,f_1,f_2 < N$). All the block sizes in LTE are even numbers and divisible by 4 & 8. If N=512, N=1024 and N=2048, then the block size N is divisible by 16, 32 and 64 respectively. When N is even, f_1 is always an odd number and f_2 is always an even number [13]. Throughout the decoding process, the interleaved address is produced recursively on the fly by this QPP Interleaver.

b=1	2			96
1 2 3 4	5 6 7 8			381 -- 384
385 -- 388	389 -- 392			765 - - 768
1153 - 1156	1157 - 1160			1533 - 1536

a,b

b=1	2		96
1 1 1 1	2 2 2 2	…	96 96 96 96
1 1 1 1	2 2 2 2	…	96 96 96 96
1 1 1 1	2 2 2 2	…	96 96 96 96
1 1 1 1	2 2 2 2	…	96 96 96 96

b=1	2	3		96
1 3 - 7	33 - 37 39	- 67 69 -	…	66 - - 72
	41 43 - 47		…	- - 78 80
- 19 - 23		81 - 85 -	…	
- - 29 31	57 - - 63	- 91 – 95	…	90 - - 96

c

b=1	2		96
1 9 - 25	- - 145 -	…	1416- - -
33 - 49 -	161 - - -	…	
		…	- - - 1504
- -113 121	- - - 249	…	1512 - - -

d

b=1	2		96
161 - - -	…		33 - 49 -
	…	- - - 1504	
- - - 249	…	1512 - - -	- -113 121
- - 145 -	…	1416- - -	1 9 - 25

e

Fig. 6: Steps in designing two stage interleaver

a: Uninterleaved Array A for N=1536,I=4,J=384,U=4,V=4 and K=96
b: Array B with N=1536,I=4,J=384,U=4,V=4 and K=96
c: Array C – partially filled with scrambled bit positions from array B
d: Interleaved Array D – partially filled
e: Interleaved Array E – after 1row and 4 column shifting

QPP de-interleaver is needed for turbo decoding. The inverse polynomial of a QPP is a permutation polynomial but not quadratic always [14]. A simple algorithm to compute the inverse of QPP was described in [14]

The Algorithm for the Second Stage of the Interleaver Is Given as Follows:

Step 1: Initialize k=1. Search elements identified by k in C and replace these elements with the elements of A.

Step 2: Repeat Step 1for all the elements of array C for k=1.

Step 3: Repeat Step 1 and Step 2 for the remaining blocks of

C, i.e. for k=2, 3…K.

Step 4: Rearrange either the columns or rows or both columns and rows of array C. Let this array be D.

Step 5: Do Armenien Shuffle Permutation (ASP) for the array D. The ASP is done by dividing the J number of columns into 16 equal blocks. If J=384, each block contains 24 columns. Each set of block is shifted according to the permutation given below:

[9 12 13 16 3 2 7 6 11 10 15 14 1 8 5 4]

i.e. the input block 1 is shifted to output block 9 in the interleaved array, input block 2 to output block 12 and so on.Let this array be interleaved array F. Multiple Interleavers can be obtained by varying the rearranging order of columns and rows in step 4. For example, if j=8, then eight columns are circular shifted (i.e. either left or right). The value of j can be varied from 1 to J-1, where J is the total number of columns. In the similar way the rows can be shifted up or down I-1 number of times, where I is the total number of rows of the interleaved array. Since the first stage is deterministic, trials are not required for the selection of blocks. M-1storage elements are required to store the permutation pattern of M-1 interleavers. Armenian Shuffle is a fast diffusing coordinate permutation. The spreading factor and the dispersion of the interleaved array gets increased by row, column rearrangement and Armenien shuffle Permutation.

Proposed Interleaver Algorithm-2: In the proposed interleaver algorithm-2 QPP interleaver is used to interleave the sequence of N information bits and arranged in an array of size I x J in first stage. In the second stage, as if in the proposed interleaver algorithm-1 the rows and columns of the interleaved sequence are rearranged. Then this is followed by Armenian Shuffle permutation. Similar to the proposed interleaver algorithm-1, additional Interleavers can be obtained by varying the rearranging order of columns and rows.

The expression of QPP interleaver is very simple. It reduces the memory contention issues when several decoders are used in parallel. The interleaved address is produced recursively on the fly and reduces the time required to design the interleaver. Therefore this reduces the computation time in decoding when several numbers of frames are used.

RESULTS AND DISCUSSIONS

The proposed algorithms for the ILCHTC are simulated in AWGN channel. The parameters used in our matlab simulation are frame length =1536, frame count=500, Channel Type: AWGN, RSC Code generator: (7, 5), Modulation: Binary Phase Shift Keying (BPSK) [15].

Simulation is performed for ILCHTC with M=3, M=2 number of component codes and M-1 number of interleavers. The first constituent encoder in ILCHTC encoder uses 4 and 2 component RSC's (where L=J=4 and L=J=2) to encode row and a zigzag encoder to encode each column respectively. The remaining constituent encoder uses zigzag encoder alone for interleaved data bits.

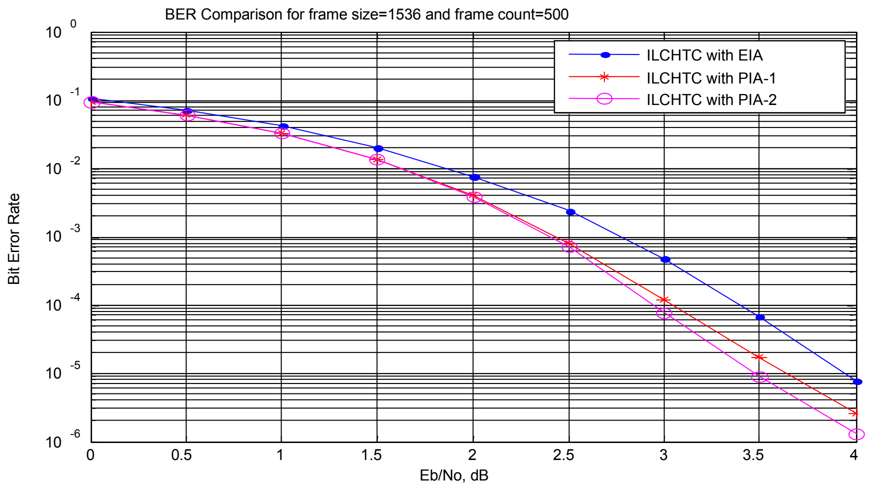

Fig. 7: BER plot of Proposed and Existing Interleaver algorithms after 9 decoding iterations for M=3, frame length=1536 and frame count=500

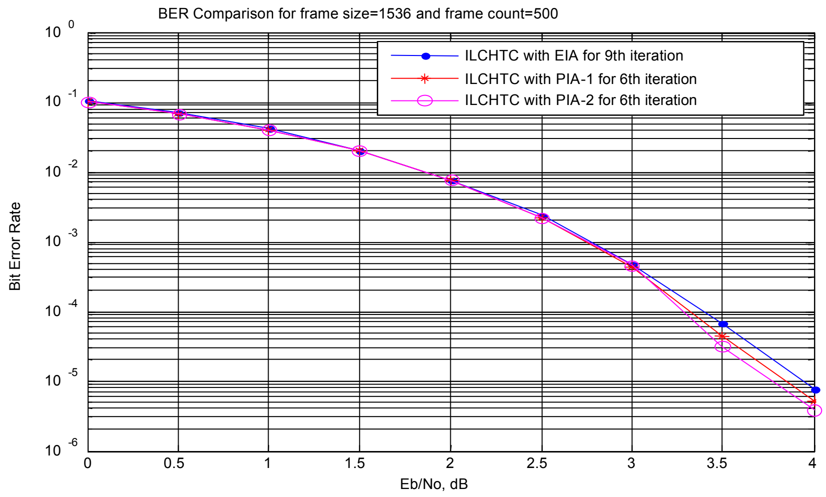

Fig. 8: Performance comparison of Proposed and Existing Interleaver algorithms for M=3, frame length=1536 and frame count=500

BER plot of ILCHTC with proposed interleaver algorithm 1 and algorithm 2 for M=3 is shown in Fig. 7 along with the comparison of existing interleaver algorithm. The existing interleaver algorithm achieves a BER of 10^{-5} in 9th iteration, so the proposed PIA-1 and PIA-2 are executed for the same 9 iterations to compare the BER performance. Simulation results show that for PIA-1, BER of 10^{-5} is achieved at Eb/No of 3.7dB and at Eb/No of 3.5dB for PIA-2 which is 0.3dB and.5dB away from Eb/No for the same BER in ILCHTC with existing interleaver algorithm respectively. Fig. 8 shows the number of iterations required to reach the BER of 10^{-5} for M=3. It is observed from the plot that to reach the BER of 10^{-5} 9 iterations are required for the EIA and 6 iterations for the proposed interleaver algorithms. This reduces the decoding complexity. At Eb/No of 3dB and above the proposed interleaver algorithms are better than the existing algorithm.

Table 2:

Code	No. of Comp./ Iter.	No. of Iter.	Total No. of Comp.
TCC	129	4	516
ILCHTC Existing	44	8	352
ILCHTC Proposed	44	6	264

The number of iterations and the number of computations per iteration determines the decoding complexity of a code. The number of computations required to achieve the BER of 10^{-5} for M=2 is given in Table II. It is observed that to achieve BER of 10^{-5} the existing interleaver algorithm requires 8 iterations but the proposed PIA-1 and PIA-2 requires only 6 iterations. Thus the reduction in number of iterations reduces the complexity.

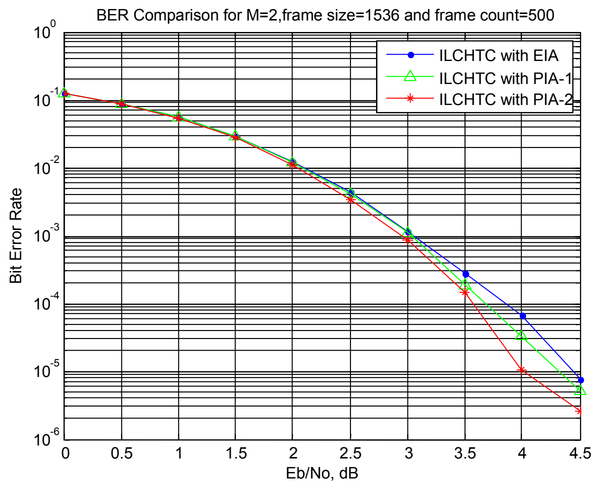

Fig. 9: BER plot of Proposed and Existing Interleaver algorithms after 10 decoding iterations for M=2,frame length=1536 and frame count=500

Figure 9 shows the BER performance of ILCHTC for M=2 after 10 decoding iterations. It is observed that BER of 10^{-5} is achieved at Eb/No of 4.3dB and 4dB for PIA-1 and PIA-2 which is 0.2dB and.5dB away from Eb/No for the same BER in ILCHTC with existing interleaver algorithm respectively.

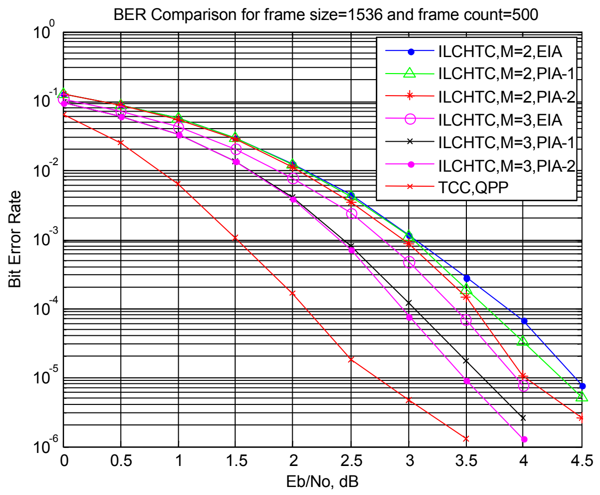

Fig. 10: BER Comparison for M=3,M=2,frame length=1536 and frame count=500

Figure 10. Shows BER comparison of ILCHTC with M=3 and M=2 for EIA, PIA-1, PIA-2 and TCC with QPP interleaver. The performance of TCC with QPP interleaver is good. Simulation results show that BER performances of the proposed interleaver algorithms are better when compared to the existing interleaver algorithm. But when proposed interleaver algorithms are compared PIA-2 (with QPP) is better than PIA-1 due to the on the fly computation of interleaving address. Also it is observed that the performance of ILCHTC with PIA-2 is almost equivalent to TCC with QPP interleaver from the BER comparison.

CONCLUSION

Interleavers play a major role in the performance of TCC. In this paper, QPP interleavers are used for TCC and MTC as they offer contention free memory accessing capability for parallel turbo decoding. Memory is not required to store the permutation pattern in PIA-1 as both the stages of 2-stage interleaver uses deterministic methods. The randomness of the interleavers is maintained by row and column shifting and Armenian Shuffle Permutation. Time required to design the PIA-2 is less, as the interleaved address is generated recursively on the fly. Simulation results shows that the number of computations required for the MTC with the proposed interleaver algorithms is less when compared to the MTC with existing interleaver algorithm and TCC with QPP interleaver. Therefore the decoding complexity of the MTC with PIA is 50% less than that of TCC.

REFERENCES

1. Berrou, C., A. Glavieux and P. Thitimajshima, 1993. Near Shannon limit error-correcting coding and decoding: Turbo codes, in proc. IEEE ICC'93, Geneva, Switzerland, pp: 1064-1070.
2. Berrou, C. and A. Glavieux, 1996. Near optimum error correcting coding and decoding: Turbo codes, IEEE Transactions on Communications, 44: 1261-1271.
3. Archana Bhise and Prakash D. Vyavahare, 2008. Low Complexity Hybrid Turbo codes, in Proc. IEEE Wireless Communication and Networking Conference, WCNC 2008, pp: 1525-1535.
4. Archana Bhise and Prakash D. Vyavahare, 2009. Improved Low Complexity Hybrid Turbo codes, in Proc. Jubilee Conference for Discrete Mathematics,
5. Archana Bhise and Prakash D. Vyavahare, 2009. Multiple Interleavers for Modified Turbo Codes, International Conference on Wireless Communication and Sensor Networks, WCSN 2009, pp: 174-178.
6. Moon, T.K., 2006. Error Correction Coding: mathematical methods and algorithms, Wiley, India, 1st edition.
7. Li Ping, Xiaoling Huang and Nam Phamdo, 2001. Zigzag Codes and Concatenated Zigzag Codes, IEEE Trans. On Information Theory, 47(2): 800-807.
8. Robertson, P., E. Villebrun and P. Hoeher, 1995. Comparison of optimal and sub-optimal map decoding algorithms operating in log domain, in IEEE International Conference on Communications,
9. Li ping, S. Chan and K.L. Yeung, 1998. Iterative Decoding of Multidimensional Concatenated Single Parity Check Codes, IEEE Int. Conf. On Communications, pp: 131-135.
10. Archana Bhise and Prakash D. Vyavahare, 2012. Modified Turbo Codes for Next Generation Wireless Networks, International Conference on Wireless and optical Communication Networks, WOCN 2012,
11. Archana Bhise and Prakash D. Vyavahare, 2011. Performance Enhancement of Modified Turbo Codes with Two stage interleavers, IET Journal on Communications, 5(10): 1336-1342.
12. Takeshita, O.Y., 2006. On maximum contention-free interleavers and permutation polynomials over integer rings, IEEE Trans. On Information Theory, 52: 1249-1253.
13. Jing Sun and O.Y. Takeshita, 2005. Interleavers for Turbo codes using permutation polynomial over integer rings, IEEE Trans. On Information Theory, 51(1): 101-119.
14. John Hoon Ryu, 2007. Permutation polynomial based interleavers for Turbo codes over integer rings: Theory and Applications, The Ohio state university,
15. Krishnamoorthy, R., N.S. Pradeep and V. Arthi, 2013. Effect of Various CODEC Parameters on the Performance of Modified Max-Log-MAP Turbo Decoding Algorithm, WSEAS Transactions on Communications. Issue 8, Volume 12.