Complexity Analysis of Iterative Decoders in Mobile Communication Systems

Archana Bhise and Prakash Vyavahare

***Abstract*—Turbo Convolutional Codes (TCC) and Low Density Parity Check (LDPC) codes, based on iterative decoders, are widely used to reduce bit error rate in mobile communication systems. The key implementation issue for these codes is the overall higher decoding complexity. Modified Turbo codes are low complexity turbo-like codes with error performance which is equivalent to that of Turbo codes. In this paper, decoding complexity of LDPC, TCC and various low complexity modified Turbo codes, such as Low Complexity Hybrid Turbo Codes (LCHTC) and Improved Low Complexity Hybrid Turbo Codes (ILCHTC), are presented. Further, modified Turbo codes are compared with TCC on the basis of error performance and decoding complexity. It is shown that the error performance of modified Turbo codes is almost equivalent to that of TCC. Moreover, modified Turbo codes require significantly lower decoding complexity than that of TCC and LDPC codes.**

***Index Terms*—Decoding complexity, bit error rate, hybrid Turbo codes, iterative decoders.**

I. Introduction

Error correcting codes with low Bit Error Rate (BER) are used for reliable data transmission in wireless communication system. A large number of error correcting codes providing reduced BER have been proposed in the literature. Convolutional codes [1], Low Density Parity Check (LDPC) codes [2] and Turbo codes [3] are widely used in wireless systems. For the given implementation complexity of the code, convolutional codes outperform linear block codes. Convolutional codes are used for Second Generation (2G) cellular systems. LDPC codes are linear block codes and are characterized by a sparse parity check matrix [4]. LDPC codes are based on iterative decoding and have performance close to the channel capacity. Several communication standards, such as IEEE 802.16e in wireless local area network [5] and DVB-S2 for digital television [6] have included LDPC codes. However, LDPC codes with large block of data require large number of computations with increased time delay.

Turbo Convolutional Codes (TCC), also called as Turbo codes, exhibit BER which is close to Shannon's limit [3]. The rate-1/2 Turbo codes achieve BER of 10^{-5} at Bit Energy to Noise (E_b/N_o) ratio of 0.7 dB which is 0.7 dB above Shannon capacity limit for 18 decoding iterations. Turbo codes are used in cellular communication systems and are included in the specifications for Universal Mobile Telecommunications System (UMTS) and cdma2000, which are 3G wireless standards [7]. However, Turbo codes require decoders with large computational complexity.

The number of mathematical operations required by Turbo decoders increases exponentially with the increase in constraint lengths of constituent convolutional codes [8].

Codes with large computational complexity adversely affect the power consumption, memory requirements and total delay of a receiver. For receivers with limited power, powerful processors and large memory are not always available. In past few years, several approaches to achieve low complexity Turbo-like code designs have appeared in the literature [9]-[12]. Modified Turbo codes provide good compromise between complexity and error performance. It has been shown that the low complexity modified Turbo codes using efficient interleavers provide error performance which is equivalent to Turbo codes. Concatenated zigzag codes are low complexity modified Turbo codes with error performance which is equivalent to Turbo codes [9]. However, concatenated zigzag codes require large interleaver length. An interleaver with large length requires more time latency as decoders require longer delay in deinterleaving the received bits. Turbo-SPC codes [10] are low complexity modified Turbo codes with error performances which are equivalent to the error performance of Turbo codes. However, at low E_b/N_o, errors do not converge for all the combinations of bits for Turbo-SPC. Recently, a class of modified Turbo codes termed as Low Complexity Hybrid Turbo Codes (LCHTC) has been proposed [11]. However, LCHTC decoder requires large number of decoding iterations. The structure of LCHTC encoder is modified to construct the Improved Low Complexity Hybrid Codes (ILCHTC) [12], [13]. It is shown that for the given E_b/N_o, ILCHTC decoders require less number of iterations than LCHTC decoders. Moreover, the decoding complexity of ILCHTC is 50% less than that of Turbo codes.

Decoder complexity of a code depends on the number of Multiplication Equivalent Operations per Information Bit per Iteration (MEO/IB/I) and Addition Equivalent Operations per Information Bit per iteration (AEO/IB/I). Additionally, for iterative decoders, computational complexity is dependent on the number of iterations. Decoding complexity in terms of mathematical operations for various iterative decoders is presented in the paper.

This paper is organized as follows. In section II, decoding complexity of LDPC and Turbo codes using Soft In Soft Out decoders is presented. Various low complexity Modified Turbo codes are described in section III. Decoding complexity of various modified Turbo codes are also

Manuscript received October 5, 2013; revised December 11, 2013.

Archana Bhise is with the Department of Electronics and Telecommunication Engineering, M P School of Technology, Management and Engineering, Mumbai (e-mail: archana.bhise@ yahoo.co.in).

Prakash Vyavahare is with the Department of SGSITS, Indore, India (e-mail: prakash.vyavahare@gmail.com).

DOI: 10.7763/IJIEE.2014.V4.420

computed in this section. Comparison of error performances and decoding complexity of various codes is presented in section IV. Section V outlines the conclusion of investigation

II. LDPC AND TURBO CODES

LDPC and Turbo codes employ iterative decoders which are based on Soft Input Soft Output (SISO) decoding algorithms. SISO decoders use max* operator to reduce decoding complexity of the codes [14]. Max* operator is defined as,

$$\begin{aligned}\max^*(f,g) &= \ln\left(e^f + e^g\right) \\ &= \max(f,g) + f_m\left(|g - f|\right)\end{aligned} \tag{1}$$

where, $f_m(.)$ is a nonlinear correction function. Max* operator is used recursively for multiple number of arguments. Let $C_{m,\max}$ be the number of Multiplication Equivalent Operations (MEO) and $C_{a,\max}$ be the number of Addition Equivalent Operations (AEO) to implement max* operator [8]. Let C_m, $L_{\max}$ be the number of MEO and $C_{a,L\max}$ be the number of AEO to implement max* using log-MAP algorithm. Neglecting the operation for maximum value selection and assuming that the logarithm and exponential operations are equivalent to multiplication operations, the number of operations to implement Log-MAP algorithm are given by,

$$C_{m,\max} = C_{m,L\max} = 2 \tag{2}$$

$$C_{a,\max} = C_{a,L\max} = 2 \tag{3}$$

Max-Log-MAP algorithm is the approximation of Log-MAP algorithm for the implementation of max* operator. Using Max-Log MAP algorithm, max* is approximated as,

$$\max^*(f,g) \approx \max(f,g) \tag{4}$$

Max-log-MAP algorithm requires only selection operation. Assuming that the number of comparisons are equivalent to addition operations, the number of computations to implement max* operator is given by,

$$C_{m,\max} = C_{m,\mathrm{Mmax}} = 0 \tag{5}$$

$$C_{\mathrm{a,max}} = C_{\mathrm{a,Mmax}} = 1 \tag{6}$$

A. LDPC Codes

LDPC codes are a class of linear block codes and are characterized by their parity check matrix, $\boldsymbol{H}$, which contains much lesser number of 1's than 0's. Tanner graph provides complete representation of the LDPC code. Tanner graph of LDPC uses two types of nodes, namely, variable nodes and check nodes. For a ($N+P$, N) LDPC code, Tanner graph consists of P check nodes and ($N+P$) variable nodes. Fig. 1 illustrates an example of a (8, 4) LDPC code. Let h_{pn} be an element in p^{th} row and n^{th} column of $\boldsymbol{H}$, where $1 \le p \le P$ and $1 \le n \le (N+P)$. A Check node, c_p is connected to variable node v_n if the element h_{pn} is 1. For the received code word, $\boldsymbol{r}$, if $\boldsymbol{rH^T} = \boldsymbol{0}$, then $\boldsymbol{r}$ is a valid code word.

LDPC decoder employs belief propagation algorithm which is based on SISO decoding [14]. Let Likelihood Ratio (LR) of a received bit be given by

$$R\left(\bar{b}_n \mid b_n\right) = \frac{P\left(\bar{b}_n \mid b_n = +1\right)}{P\left(\bar{b}_n \mid b_n = -1\right)} \tag{7}$$

where $P(\bar{b}_n \mid b_n = \pm 1)$ is the probability of receiving $\bar{b}_n$ if transmitted bit b_n, is ± 1, $1 \le n \le N+P$. Equivalently, Logarithm of Likelihood Ratio (LLR) is given by

$$L\left(\bar{b}_n \mid b_n\right) \equiv Log\left(R\left(\bar{b}_n \mid b_n\right)\right) = \tilde{b}_n \tag{8}$$

For Additive White Guassian Noise (AWGN) channel with variance, σ,

$$\tilde{b}_n = \frac{2\bar{b}}{\sigma^2} \tag{9}$$

Let $q_{np}^{(t)}$ be the LLR of the message sent by the variable node, v_n to the check node c_p in tth iteration and $r_{pn}^{(t)}$ be the LLR of the response message sent by cp to the variable node vn in tth iteration, $1 \le n \le (N + P)$ and $1 \le p \le P$.

The process of decoding starts with the selection of A Posteriori Probability (APP) for each data bit, which is followed by choosing the data bit value that corresponds to the Maximum A Posteriori (MAP) probability for that data bit. Belief propagation algorithm is described below:

For the first iteration, $\tilde{b}_n$ is the *a priori* probability of data bits. Initialize $t = 1$ and $q_{np}^{(0)} = \tilde{b}_n$. Let w_r and w_c be the number of 1's in each row and each column of $\boldsymbol{H}$.

- Step 1: The response message at the check node in t^{th} iteration is given by,

$$r_{np}^{(t)} = \max^*\left(q_{1p}^{(t-1)}, q_{2p}^{(t-1)}, ..., q_{ip}^{(t-1)}, ..., q_{w_r p}^{(t-1)}\right) \tag{10}$$

where, $q_{ip}^{(t-1)}$ is the response message for the branch of Tanner graph which connects i^{th} variable node to p^{th} check node. Since each check node is connected to maximum w_r variable nodes, the max* in (10) operates on w_r number of parameters.

- Step 2: The variable nodes update their response message to the check nodes. LLR of the message at the check node in t^{th} iteration is given by,

$$q_{np}^{(t)} = \sum_{p' \in P \mid p} r_{p'n}^{(t)} + \tilde{b}_n \tag{11}$$

where summation of response messages is taken for all the check nodes except p^{th} node.

- Step 3: LLR of the received bits at variable nodes are computed to estimate the transmitted bits. LLR of the n^{th} received bit, $q_n^{(t)}$, is given by,

$$q_n^{(t)} = \sum_{p=1}^{P} r_{pn}^{(t)} + \tilde{b}_n \tag{12}$$

Current estimation, $\hat{b}_n$ of the received bit $\tilde{b}_n$ is given by,

$$\begin{aligned}\hat{b}_n &= 1 \quad \text{for } q_n^{(t)} > 0 \\ \hat{b}_n &= 0 \quad \text{for } q_n^{(t)} \le 0\end{aligned} \tag{13}$$

- Step 4: Updated LLR of received bits are used to compute syndrome, S^t, for the t^{th} iteration. For a linear block code, S^t is given by

$$\boldsymbol{S^t = \hat{b}H^T} \tag{14}$$

If $\boldsymbol{S^t = 0}$ then go to Step 5 else $t = t+1$ and go to *Step 2*.

- Step 5: End.

Decoding complexity of LDPC is computed using the number of mathematical operations in each step of the decoding algorithm. The computation of response messages of check nodes in *Step 1* requires max* on w_r variable nodes. Let $C_{m,1}$ and $C_{a,1}$ be the number of MEO/Iteration and AEO/Iteration required to compute response messages in *Step 1* respectively. Using (2), (3) and (10),

$$C_{m,1} = P \times (w_r - 1) \times C_{m,\max} \tag{15}$$

$$C_{a,1} = P \times (w_r - 1) \times C_{a,max} \tag{16}$$

In *Step 2*, response message at $(N+P)$ variable nodes are updated using the message from P check nodes. Let $C_{m,2}$ and $C_{a,2}$ be the number of MEO/Iteration and AEO/Iteration required to update response messages at variable nodes in *Step 2* respectively. Using (11),

$$C_{m,2} = 0 \tag{17}$$

$$C_{a,2} = P \times (N+P) \tag{18}$$

Further, estimation of transmitted bits requires computation of LLR of the received bit in *Step 3*. Let $C_{m,3}$ and $C_{a,3}$ be the number of MEO/Iteration and AEO/Iteration required to update response messages at variable nodes in *Step 3* respectively. Using (12),

$$C_{m,3} = 0 \tag{19}$$

$$C_{a,3} = (P+1) \times (N+P) \tag{20}$$

Let $C_{m,4}$ and $C_{a,4}$ be the number of MEO/Iteration and AEO/Iteration required to compute the syndrome in Step 4. Using (14),

$$C_{m,4} = (N+P) \times P \tag{21}$$

$$C_{a,4} = (N+P-1) \times P \tag{22}$$

Using (15) to (22), the number of Multiplication Equivalent Operations per Information Bit per Iteration (MEO/IB/I), $C_{m,\mathrm{LDPC}}$ and Addition Equivalent Operations per Information Bit per Iteration (AEO/IB/I), $C_{a,LDPC}$ for LDPC are given by,

$$C_{m,LDPC} = (C_{m,1} + C_{m,2} + C_{m,3} + C_{m,4})/N$$

$$= \frac{P \times C_{m,\max}(w_r - 1) + (N+P) \times P}{N} \tag{23}$$

$$C_{a,LDPC} = (C_{a,1} + C_{a,2} + C_{a,3} + C_{a,4})/N$$

$$= \frac{P \times C_{a,\max}(w_r - 1) + (N+P) \times P + (P+1)(N+P) + (N+P-1) \times P}{N}$$

B. Turbo Codes

Turbo encoder consists of parallel concatenation of two rate-1/2 Recursive Systematic Convolutional (RSC) encoders [3]. Fig. 2 shows the encoder structure of TCC used for UMTS system, with generator matrix, $G = [1, 15/13]_8$. For TCC, a sequence of N data bits is encoded by the first encoder in its natural order. Second encoder encodes interleaved data bits. Let the sequence of data bits be represented by $\boldsymbol{d} = \{d_1, d_2, \ldots, d_N\}$ and parity bits generated by the first and second RSC encoders be represented by $\boldsymbol{r^{(1)}}$ and $\boldsymbol{r^{(2)}}$ respectively. Where, $r^{(1)} = \{r_1^{(1)}, r_2^{(1)}, \ldots, r_N^{(1)}\}$ and $r^{(2)} = \{r_1^{(2)}, r_2^{(2)}, \ldots, r_N^{(2)}\}$. Code word, $\boldsymbol{C_{TCC}}$, for TCC is given by,

$$\boldsymbol{C_{TCC}} = \{\boldsymbol{d}, \boldsymbol{r^{(1)}}, \boldsymbol{r^{(2)}}\} \tag{25}$$

Turbo decoder consists of two constituent decoders. Each decoder is a Maximum *A Posteriori* (MAP) decoder which is based on LLR of received bits [14]. First RSC decoder operates during the first half iteration and generates extrinsic probability $\boldsymbol{e^{(1)}}$. Second RSC decoder operates during the second half iteration and generates extrinsic probability, $\mathbf{e^{(2)}}$. SISO decoding is used in Turbo decoders. The architecture of TCC decoder is illustrated in Fig. 3.

Let $C_{m,c}$ and $C_{a,c}$ be the number of MEO/IB/I and AEO/IB/I required by convolutional decoder respectively. It is shown that the number of MEO/IB/I and AEO/IB/I for convolutional decoder, using S_t states, are given by [14],

$$C_{m,c} = (4S_t - 2) \times C_{m,max} \tag{26}$$

$$C_{a,c} = (4S_t - 2)\, C_{a,max} + 12S_t + 1 \tag{27}$$

First RSC decoder uses two inputs; LLR of received convolutional parity bits, $\tilde{r}^{(1)}$ and LLR of data bits, which is the addition of LLR of received data bits, and extrinsic probability generated by the second decoder. Let $C_{m,frst}$ and $C_{a,frst}$ be the number of MEO/IB/I and AEO/IB/I respectively for the first RSC decoder of TCC. The number of computations required by the first RSC decoder is given by,

$$C_{m,frst} = C_{m,c} = (4S_t - 2) \times C_{m,max} \tag{28}$$

$$C_{a,frst} = C_{a,c} + 1$$

$$= (4S_t - 2)\, C_{a,max} + 12S_t + 1 \tag{29}$$

The number of computations for the second RSC decoder of TCC is same as that for first decoder. Therefore, the number of computations for the second decoder is given by,

$$C_{m,scnd} = C_{m,frst} \tag{30}$$

$$C_{a,scnd} = C_{a,frst} \tag{31}$$

Output of second decoder is used to generate extrinsic information for the next iteration. The number of MEO/IB/I, $C_{m,TCC}$ and the number of AEO/IB/I, $C_{a,TCCC}$ for TCC decoder is given by,

$$C_{m,TCC} = C_{m,frst} + C_{m,scnd} = 4(2S_t - 1) \times C_{m,max} \tag{32}$$

$$C_{a,TCC} = C_{a,frst} + C_{a,scnd} + 3 = 4(2S_t - 1) \times C_{a,max} + 24S_t + 5 \tag{33}$$

For a general TCC decoder with M constituent codes, the number of computations is given by,

$$C_{m,TCC} = M \times (C_{m,frst}) = M \times (4S_t - 2) \times C_{m,max} \tag{34}$$

$$C_{a,TCC} = M \times (C_{a,frst} + 1) + 1 = M\,[\,((4S_t - 2) \times C_{a,max} + 12S_t + 2)] + 1 \tag{35}$$

III. Modified Turbo Codes

TCC and LDPC codes are excellent error correcting codes for wireless channels. However, TCC decoders require large number of mathematical operations and computational complexity of LDPC encoders is large [14]. MTC are multiple concatenations of low complexity convolutional codes and block codes. It is shown that low complexity Modified Turbo Codes (MTC) with iterative decoding offer error performance which is equivalent to Turbo codes [9]-[12].

For a general MTC, a sequence of N information bits is arranged in an array of size $J \times K$. Therefore,

$$N = J \times K \tag{36}$$

Depending on the concatenation scheme and the type of constituent code, MTC encodes columns/rows of the array of information bits.

Fig. 4 shows the constituent code of a general MTC. Encoder 1 encodes columns and encoder 2 encodes rows of information array. Parity bits generated by either encoder 1 or encoder 2 are encoded by encoder 3. Depending on the type of MTC, one or more of these encoders are used. Particular combination of encoder 1, encoder 2 and encoder 3 forms a constituent code of MTC. Parallel concatenations of M constituent codes form the overall encoder structure of MTC. Each constituent code of MTC encodes interleaved array of information bits except first constituent code [15, 16]. Fig. 5 shows the overall structure of a general MTC, where, $\pi^{(m)}$ denotes m^{th} interleaver, $1 \le m \le M$-1. Some of the low complexity modified turbo codes are described below:

A. Concatenated Zigzag Codes

Concatenated Zigzag Codes (CZC) are low complexity codes which consist of parallel concatenation of zigzag codes [9]. In each constituent code, zigzag encoder computes parity bits for the columns of 2-dimensional information array. In Fig. 4, encoder 1 is a zigzag code. Encoder 2 and encoder 3 are not used for CZC. Transmitted code word, $\boldsymbol{C_z}$ for CZC is represented by,

$$\boldsymbol{C_z} = \{ \boldsymbol{d}, \boldsymbol{z^{(1)}}, \boldsymbol{z^{(2)}}, \ldots, \boldsymbol{z^{(M)}} \} \tag{37}$$

Constituent decoders of CZC use max* operator to compute LLR of information bits. Let $F[\tilde{z}(k)]$, $B[\tilde{z}(k)]$ and $L[\tilde{d}(j,k)]$ be the forward LLR and backward LLR and current LLR of received information bits respectively where, $1 \le k \le K$.

$$F[\tilde{z}(k)] = \tilde{z}(k) + \max^*(F[\tilde{z}(k-1)], \tilde{d}(1,k), \ldots, \tilde{d}(J,k)) \tag{38}$$

$$B[\tilde{z}(k-1)] = \tilde{z}(k-1) + \max^*(\tilde{d}(1,k), \ldots, \tilde{d}(J,k), B[\tilde{z}(k)]) \tag{39}$$

$$L[\tilde{d}(j,k)] = \tilde{d}(j,k) + \max^*(F[\tilde{z}(k-1), \tilde{d}(1,k), \tilde{d}(2,k), \ldots, \tilde{d}(j-1,k), \tilde{d}(j+1,k), \ldots, \tilde{d}(J,k), B[\tilde{z}(k)]) \tag{40}$$

Equations (38), (39) and (40) requires computation of max*. For a segment of J bits, max* operator in (38) is executed J times in addition to one addition operation. Let $C_{m,f}$ and $C_{a,f}$ be the number of MEO/IB/I and AEO/IB/I for computation of forward MLA and are given by

$$C_{m,f} = J \times C_{m,max} \tag{41}$$

$$C_{a,f} = (J \times C_{a,max}) + 1 \tag{42}$$

Result of (38) is used to evaluate backward LLR in (39). Therefore, backward MLA requires computation of max* only once. Let $C_{m,b}$ and $C_{a,b}$ be the number of MEO/IB/I and AEO/IB/I required to compute backward MLA. Then

$$C_{m,b} = C_{m,max} \tag{43}$$

$$C_{a,b} = C_{a,max} + 1 \tag{44}$$

Let $C_{m,d}$ and $C_{a,d}$ be the number of MEO/IB/I and AEO/IB/I required to decode an information bit in zigzag decoder. Using (40), the number of computations required for LLR of a bit in a column of an array is given by,

$$C_{m,d} = (J+1) \times C_{m,max} \tag{45}$$

$$C_{a,d} = ((J+1) \times C_{a,max}) + J \tag{46}$$

For CZC with M constituent codes, extrinsic information is computed for each decoder. Therefore, an addition operation is required to compute extrinsic information before the next constituent decoder in each iteration. Let $C_{m,czc}$ and $C_{a,czc}$ be the number of MEO/IB/I and AEO/IB/I required by CZC to decode an information bit. Using (41) to (46), the decoding complexity of CZC with M constituent decoder is given by,

$$C_{m,czc} = M(C_{m,f} + C_{m,b} + C_{m,d})/J = M[(2J+2) \times C_{m,max}]/J \tag{47}$$

$$\begin{aligned} C_{a,czc} &= M[C_{a,f} + C_{a,b} + C_{a,d} + J]/J \\ &= 2M[(J+1) + (J+1)C_{a,max}]/J \end{aligned} \tag{48}$$

B. Low Complexity Hybrid Turbo Codes

Low Complexity Hybrid Turbo Codes (LCHTC) is a new class of modified Turbo codes [11]. LCHTC uses hybrid concatenation of zigzag code and convolutional code. In each constituent encoder, zigzag encoder computes parity bits for the columns of information bit array (encoder 1 in Fig. 4). Zigzag parity bits of first L constituent codes are encoded by rate-1/2 RSC code (encoder 3 in Fig. 4). Encoder 2 is not used for LCHTC.

Decoding complexity of LCHTC is dependent on the number of computations required by convolutional and zigzag decoder. For LCHTC decoder, convolutional decoders are used to decode K zigzag parity bits in first L constituent decoders. LLRs of the decoded bits of first L constituent codes are much larger than that of remaining constituent decoders. Therefore, LLRs of information bits decoded by L decoders are multiplied by the damping factor to achieve optimum speed of error convergence for the overall LCHTC decoder. For LCHTC, the number of MEO/IB/I required by L number of convolutional decoders, $C_{m,LCC}$, is given by,

$$C_{m,LCC} = (L \times K \times (C_{m,c} + 1))/N \tag{49}$$

Therefore, using (24) and (34),

$$C_{m,LCC} = (L/J) \times (C_{m,max} \times (4S_t - 2) + 1) \quad (50)$$

Similarly, number of AEO/IB/I required in LCHTC by L convolutional decoders, $C_{a,LCC}$, is given by,

$$C_{a,LCC} = L \times (K \times C_{a,c}/N) \quad (51)$$

Using (25) and (35),

$$C_{a,LCC} = (L/J) [(4S_t - 2) C_{a,max} + 12S_t + 1)] \quad (52)$$

LCHTC uses zigzag codes for all M constituent codes. Let $C_{a,zCC}$ and $C_{m,zCC}$ be the total number of AEO/IB and MEO/IB required by zigzag and convolutional code of LCHTC. Using (47), (48), (50) and (52), the number of computations are given by,

$$C_{m,zCC} = C_{m,LCC} + C_{m,czc} \quad (53)$$

$$C_{a,zCC} = C_{a,LCC} + C_{a,czc} \quad (54)$$

The overall LCHTC decoder is shown in Fig. 6. For overall LCHTC decoder, extrinsic probability is computed at the output of a constituent decoder and is given at the input of next constituent decoder. Mathematically, the extrinsic information vector generated by the mth decoder in the tth iteration is defined by

$$\tilde{x}_t^{(m)} = \tilde{d}_{o,t}^{(m)} - \tilde{d}_{i,t}^{(m)} \quad (55)$$

The *a priori* LLR of $\tilde{d}_{i,t}^{(m)}$, is given by

$$\tilde{d}_{i,t}^{(1)} = \tilde{d}_{o,t}^{(M)} - \tilde{x}_{t-1}^{(1)}, \quad m = 1$$
$$\tilde{d}_{i,t}^{(m)} = \tilde{d}_{o,t}^{(m-1)} - \tilde{x}_{t-1}^{(m)}, \quad m > 1 \quad (56)$$

The LLRs of the information bits, $\tilde{d}_{i,t}^{(m)}$, are updated iteratively using (55) and (56). In the first iteration, $\tilde{d}$, is used as *a priori* information for the first decoder. In the subsequent iterations, extrinsic probability generated by the Mth decoder is used to update the *a priori* information of the information bits for the first decoder. Therefore,

$$\tilde{d}_{i,t}^{(1)} = \tilde{d}, \text{ for } t = 1 \quad (57)$$
$$\tilde{d}_{i,t}^{(1)} = \tilde{d}_{o,t}^{(M)} - \tilde{x}_{t-1}^{(1)}, \text{ for } t > 1$$

Let $C_{a,LC}$ and $C_{m,LC}$ be the number of AEO/IB/I and MEO/IB/I multiplications required by overall LCHTC decoder. Decoding complexity of LCHTC decoder using (53) and (54) is given by,

$$C_{m,LC} = C_{m,zCC} = \frac{((2 \times L \times S_t) + (J \times M) + M - L) \times 2C_{m,\max}}{J} + \frac{L}{J} \quad (58)$$

$$C_{a,LC} = C_{a,zCC} + 2M$$

$$= \frac{L\left((4S_t - 2)C_{a,\max} + 12S_t + 1\right) + M\left((2J + 2)C_{a,\max} + 2J + 2\right)}{J} + 2M$$

Decoding complexity of a code is dominated by the number of computations required by convolutional decoders [8]. Given the number of states, S_t, of convolutional code, decoding complexity of convolutional code is dependent on the trellis length of convolutional code [14]. Trellis length of Turbo code is $(2 \times N)$ whereas trellis length of LCHTC, T_r, is given by,

$$T_r = L \times (N/J) \quad (60)$$

For $J = 4$, $L = 4$ *and* $M = 4$, maximum trellis length of a rate-1/3 LCHTC is N.

C. Turbo-SPC Code

Encoder structure of Turbo-SPC is similar to LCHTC except that zigzag codes in LCHTC are replaced by SPC codes. For Turbo-SPC codes, encoder 1 (in Fig. 4) is a Single Parity Check (SPC) code and encoder 3 is a rate-1/2 RSC code [10]. Encoder 2 is not used for Turbo-SPC codes. Let $C_{m,SPC}$ and $C_{a,SPC}$ be the number of MEO/IB/I and AEO/IB/I for SPC decoder respectively. SPC decoder of Turbo-SPC code decodes J information bits of each column. Therefore, decoding complexity of SPC decoder in the constituent code of Turbo-SPC is given by,

$$C_{m,SPC} = (J\text{-}1)\, C_{m,max} \quad (61)$$

$$C_{a,SPC} = (J\text{-}1)\, C_{a,max} + 1 \quad (62)$$

Let $C_{m,TSPC}$ and $C_{a,TSPC}$ be the number of MEO/IB/I and AEO/IB/I required by the overall Turbo-SPC decoder. Moreover, M addition equivalent operations are required to compute extrinsic probability for the constituent decoders. Therefore, decoding complexity of Turbo-SPC decoder using (50), (52), (61) and (62) is given by,

$$C_{m,TSPC} = C_{m,LCC} + C_{m,SPC}$$
$$= (M/J) \times [(4S_t - 2)C_{m,max} + 1] + (J - 1)C_{m,max} \quad (63)$$

$$C_{a,TSPC} = C_{a,LCC} + C_{a,SPC} + M$$
$$= (M/J) \times [(4S_t - 2)C_{a,max} + 12S_t + 1] + [(J - 1)C_{a,max} + 1]$$

D. Improved Low Complexity Hybrid Turbo Codes

Encoder structure of LCHTC is modified to construct Improved Low Complexity Hybrid Turbo Codes (ILCHTC) [12]. For ILCHTC, first E rows of information array are encoded using rate-1/2 RSC code (encoder 2). Only Zigzag parity bits are computed for each column of the information array in other constituent encoders except the first one (encoder 1). Other constituent encoders of ILCHTC are same as that of LCHTC. In first constituent decoder of ILCHTC, convolutional decoder decodes E rows of information array. Let $C_{m,ILCC}$ and $C_{a,ILCC}$ be the number of MEO/IB/I and AEO/IB/I required by the convolutional decoder in ILCHTC. Using (24) and (25), decoding complexity of convolutional codes in ILCHTC is given by,

$$C_{m,ILCC} = \frac{C_{m,c} \times E \times K}{N} + 1$$
$$= [(E/J) \times (4S_t - 2) \times C_{m,max}] + 1 \quad (65)$$

$$C_{a,ILCC} = \frac{C_{a,c} \times E \times K}{N}$$
$$= (E/J)\, [(4S_t - 2)\, C_{a,max} + 12S_t + 1] \quad (66)$$

For the remaining $(M - 1)$ constituent codes, only zigzag

decoder is used. Let $C_{m,zILCC}$ and $C_{a,zILCC}$ be the number of MEO/IB/I and AEO/IB/I required by zigzag decoder of ILCHTC respectively. The number of computations required by M decoders of ILCHTC is, therefore, given by

$$C_{m,zILCC} = C_{m,ILCC} + C_{m,czc} \quad (67)$$

$$C_{a,zILCC} = C_{a,ILCC} + C_{a,czc} \quad (68)$$

Overall ILCHTC decoder requires one addition equivalent operation to compute extrinsic probability before each decoder. Let $C_{m,ILC}$ and $C_{a,ILC}$ be the number of MEO/IB/I and AEO/IB/I multiplications required by overall LCHTC decoder. Using (67) and (68), decoding complexity of ILCHTC decoder is given by,

$$C_{m,ILC} = C_{m,zILCC}$$

$$= \frac{((2\times E\times S_t)+(J\times M)+M-E)\times 2C_{m,\max}}{J} + \frac{E}{J} \quad (69)$$

$$Ca,ILC = Ca,zILCC + \quad M$$

$$= \frac{E\left((4S_t-2)C_{a,\max}+12S_t+1\right)+M\left((2J+2)C_{a,\max}+2J+2\right)}{J} + 2M$$

To compare decoding complexity of various codes in unified manner, number of MEO/IB/I and AEO/IB/I required for various rate-1/2 codes are computed. Table I shows decoding complexity of various codes using Log-MAP algorithm and Max-Log-MAP algorithm. It is observed that TCC requires maximum number of mathematical operations.

IV. Comparison of Codes

The BER performances of TCC, ILCHTC, LCHTC, Turbo-SPC and concatenated zigzag codes, at rate 1/3, are shown in Fig. 7 for comparison. TCC shows the best BER performance among these codes. The simulation of ILCHTC shows that it achieves a BER of 10^{-5} at an E_b/N_o = 1.9 dB which is 0.4 dB more than that of TCC adopted in the Third Generation Partnership Project (3GPP) for wireless communication systems [8]. However, as shown in Table I the decoding complexity of ILCHTC is considerably lower than that of TCC [13]. Therefore, ILCHTC is suitable for wireless communication applications. Among modified Turbo codes under evaluation, ILCHTC requires minimum E_b/N_o to achieve BER of 10^{-5}. BER performance of ILCHTC is better than that of LCHTC. However, for E_b/N_o < 1.2 dB, the BER performance of TCC is far better than that of modified Turbo codes under evaluation.

Decoding complexity of a code is dependent on the number of iterations and the number of computations per iteration. Moreover, the number of MEO/IB/I and AEO/IB/I increases with the trellis length of the convolutional codes used in the constituent codes. Average number of iterations required by Zigzag, Turbo-SPC, LCHTC, ILCHTC and TCC for different values of E_b/N_o for rate, R_c =1/2, is illustrated in Fig. 8. TCC requires least number of iterations whereas LDPC requires maximum number of iterations. ILCHTC requires 10% more iterations than TCC.

Let C_m and C_a be the number of MEO/IB and AEO/IB required for the various decoders respectively. Table II shows error correcting codes used in various standards, their decoding complexity and the possibility of using proposed codes in those standards.

TABLE I: Complexity Comparison of Iterative Decoders

Code	Parameter	Log-MAP			Max-Log-MAP	
		C_m	C_a	$C_m + C_a$	C_m	C_a
TCC	M = 2, S_t = 8	120	317	437	0	257
Turbo-SPC	M = 4, J = 4 S_t = 8	67	168	235	0	135
LCHTC	M = 2, J = 4 L = 2, S_t = 8	41	94	135	0	74
ILCHTC	M = 2, J = 4 E = 2, S_t = 8	41	94	135	0	74
LDPC	N=1440 w_r = 7, w_c = 6	38	38	76	0	12
C.Z.Code	M = 4, J = 4	20	30	50	0	20

TABLE II: Error Correcting Codes and Their Decoding Complexity for Various Wireless Standards

Wireless Standards	Error correcting codes					
	Code	R_c	S_t	Decoding Complexity		
				C_m	C_a	Total C_m+C_a
IS-95	Convolu.	1/3	9	2044	5373	7417
GSM	Convolu.	½	5	124	317	441
UMTS	Turbo	1/3	4	120	317	437
cdma2000	Convolu.	½	9	2044	5177	7161
		1/3	9	2044	5373	7417
		¼	9	2044	5629	7673
	Turbo	½	4	120	317	437
		1/3	4	120	317	437
		¼	4	180	475	655
	LCHTC	½	4	41	94	135
		1/3	4	81	195	276
	ILCHTC	½	4	41	94	135
		1/3	4	81	195	276

It is observed that the convolutional codes and Turbo codes are extensively used to reduce bit error rate in wireless standards for mobile communication. The selection between convolutional code and Turbo code is dependent on the type of logical channels and the application of the wireless standard for communication. It is observed that the constraint length of convolutional codes is larger than that of Turbo codes. This is because convolutional codes require more constraint length than that of Turbo codes to achieve the desired bit error rate. Since decoding complexity of these codes increases exponentially with the increase in constraint length, convolutional decoders require more mathematical operations than Turbo decoders.

Analysis shows that the decoding complexity of rate-1/2 Turbo codes is same as that of rate-1/3 Turbo codes. This is because parity bits of rate-1/3 TCC are punctured to change the code rate to 1/2 and puncturing does not affect the overall decoding complexity of Turbo codes. It is also observed that LCHTC and ILCHTC require the same number of computations per iteration. However, the number of iterations in ILCHTC is less than that of LCHTC. Therefore, overall decoding complexity of ILCHTC would be less than LCHTC.

Table III presents coding gains and the required number of computations for LDPC, TCC, Turbo-SPC, ILCHTC, LCHTC and concatenated zigzag codes for R_c = 1/2, N = 1452 and E_b/N_o = 1.6 dB in AWGN channel. It is observed

that TCC and LDPC provide almost equivalent coding gains. Decoding complexity of LDPC codes is maximum among the codes under evaluation. Moreover, the number of computations required by ILCHTC decoder in one iteration is almost 30% of the number of computations required by the TCC. However, ILCHTC require 76% more iterations than TCC. Therefore, the overall decoder complexity of ILCHTC is 50% less than that of TCC. Simulation results show that the coding gain of ILCHTC is 0.4 dB less than that of TCC. However, decoding complexity of ILCHTC is considerably lower than that of TCC. Analysis shows that concatenated zigzag codes require 25% less computations than ILCHTC. However, coding gain of concatenated Zigzag code is 0.8 dB less than that of ILCHTC.

TABLE III: DECODING COMPLEXITY AND CODING GAINS OF RATE-1/2 CODES FOR N = 1452, BER= 10-5 AND EB/NO = 1.6 DB

Code	comput./ Iter	No. of iteration	Total no. of computations	Coding gain for BER of in dB
LDPC	76	32.0	2432	8.3
TCC	437	5.1	2228	8.5
Turbo-SPC	235	8.9	2091	7.8
LCHTC	135	12.0	1620	7.9
ILCHTC	135	9.0	1215	8.1
Con.Zigzag	50	18.2	910	7.3
Convolu. (soft deci.)	217	1.0	217	4.8
(31,26) Hamming	336 (binary)	1.0	336	3.0

Fig. 9 shows the comparison for the number of computations/iteration for the convolutional codes and Turbo codes used for 3G and the newly proposed LCHTC and ILCHTC. It is observed that the decoding complexity of convolutional codes is considerably larger than that of Turbo codes. This is because convolutional codes require larger constraint length than Turbo codes to achieve the same BER. It is also observed that there is a marginal increase in the decoding complexity for the convolution codes with the reduced code rate. This is because increase in the number of parity bits results in the reduction of the code rate of convolutional codes and increase in parity bits increase only addition equivalent operations. Analysis shows that the decoding complexity for a rate-1/4 convolutional code is 7% more than that for a rate 1/2 convolutional code which is very low. Decoding complexity of LCHTC and ILCHTC is 3% of the decoding complexity of convolutional codes.

V. CONCLUSION

LDPC and Turbo codes are Shannon capacity approaching codes. However, Turbo decoders are complex and error convergence of LDPC codes is slower than that of Turbo codes. Therefore, modified Turbo codes with lower decoding complexity than Turbo codes, are recently proposed in literature. Decoding complexity is one of the factors in the selection of error correcting codes for mobile communication systems. Decoding complexity of various codes in terms of multiplication equivalent and addition equivalent operations is presented to compare the number of computations. It is observed that decoding complexity of modified Turbo codes is considerably lower than that of Turbo codes. Simulation results show that LDPC and Turbo codes require almost same number of computations. The decoding complexity of LDPC and TCC is more than that of Turbo-SPC, ILCHTC, LCHTC and Zigzag codes. Based on the complexity analysis comparison presented in the paper, suitable choice can be made in selecting a code for an application depending on available E_b/N_o and the coding gain requirements.

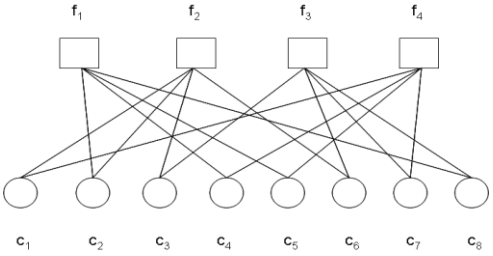

Fig. 1. Tanner graph for (8, 4) LDPC code.

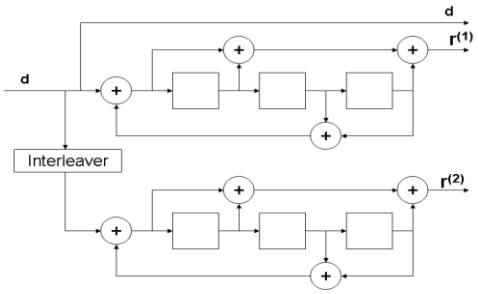

Fig. 2. Encoder structure of turbo convolutional code.

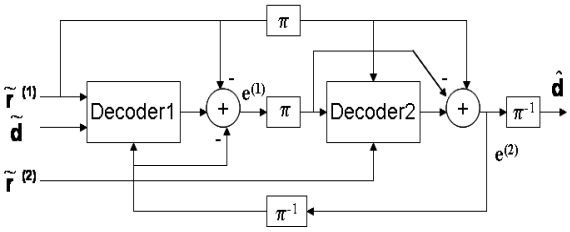

Fig. 3. Iterative turbo decoder.

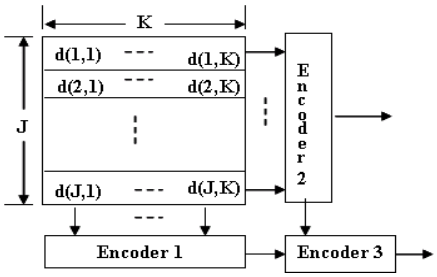

Fig. 4. Constituent code of a general modified turbo code.

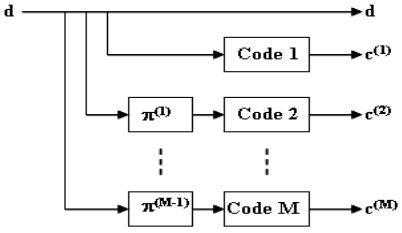

Fig. 5. Overall Modified turbo encoder.

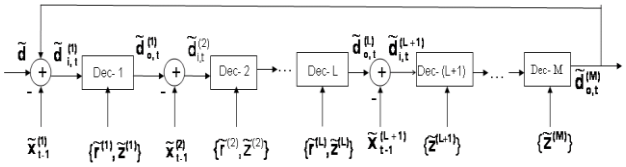

Fig. 6. Overall LCHTC decoder.

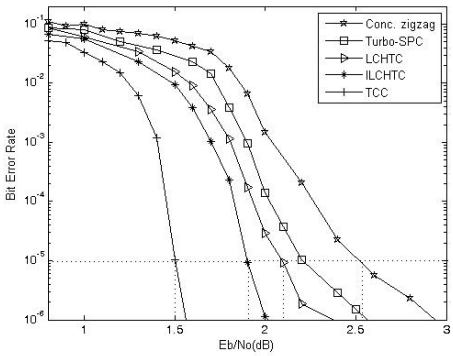

Fig. 7. Performance comparisons for rate-1/3 codes, N = 1452.

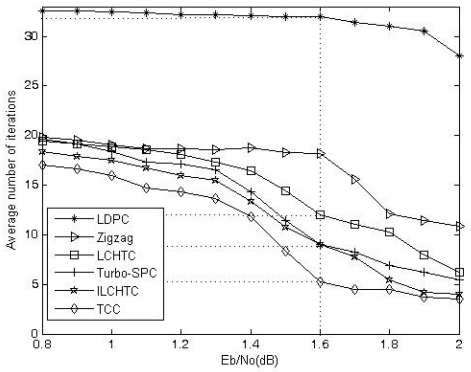

Fig. 8. Number of iterations required by rate-1/2 codes, N = 1452.

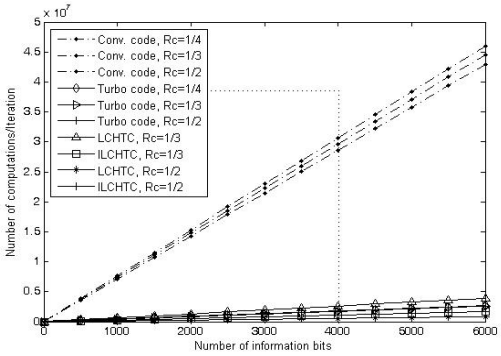

Fig. 9. Total number of computations/iteration for various codes.

REFERENCES

[1] G. D. Forney, "Convolutional codes: I. Algebraic Structure," *IEEE Transaction Information Theory*, vol. 16, no. 6, pp. 720-738, November 1970.

[2] T. J. Richardson and R. L. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE Transactions in Information Theory*, vol. 47, no. 2, pp. 638-656, February 2001.

[3] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon limit error correcting coding and decoding: Turbo codes," in *Proc. IEEE Proceedings of the International Conference on Communications (ICC 93)*, Geneva, Switzerland, May 1993, pp. 1064-1070.

[4] R. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21-28, January 1962.

[5] M. Zhuo, L. Ying and M. W. Xin, "A Quasi-parallel encoder Quasi-Cyclic LDPC Codes, IEEE 802.16e," in *Proc. International conference on Information Science and Engineering*, Dec 2009, pp. 2492-2495.

[6] J. Haeseong and T. Jong, "Implementation of LDPC decoder in DVB-S2 using Min-Sum algorithm," in *Proc. International Conference on Convergence and Hybrid Information Technology*, August 2008，pp. 359-362.

[7] 3GPP, "Technical Specification, Group Radio Access Network (Multiplexing and Channel Coding for FDD)," *Tech. Rep*., 3rd Generation Partnership Project, 1999.

[8] M. C. Valenti and J. Sun, "The UMTS Turbo Code and an efficient decoder implementation suitable for software-defined radios," *International Journal of Wireless Information Network*, vol. 8, no. 4, pp. 203-214, Oct. 2001.

[9] P. Li, X. L. Huang, and N. Phamdo, "Zigzag codes and concatenated zigzag codes," *IEEE Trans. on Information Theory*, vol. 47, vo. 2, Feb. 2001, pp. 800- 807.

[10] P. Li, "Turbo-SPC Codes," *IEEE Trans. on Communication*s, vol. 49, no. 5, pp. 754-759, May 2001.

[11] A. Bhise and P. D. Vyavahare, "Low complexity hybrid turbo codes," in *Proc. IEEE Wireless Communication and Networking Conference*, presented at the WCNC 2008, Las Vegas, 31st March – 3rd April, 2008, pp. 1525-1535.

[12] A. Bhise and P. D. Vyavahare, "Improved low complexity hybrid turbo codes and their performance analysis," *IEEE Transaction on Communications*, vol. 58, no. 6, pp. 1620-1622, June, 2010.

[13] A. Bhise and P. D. Vyavahare, "Improved low complexity hybrid turbo codes and union bound analysis," *IET journal on communications*, vol. 5, no. 4, pp. 512-518, March, 2010.

[14] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*, 2nd Edition, Pearson Education, 2011.

[15] A. Bhise and P. D. Vyavahare, "Multiple interleavers for modified turbo codes," presented at the International Conference on Wireless Communication and Sensor Networks, WCSN, 2009, IIIT, Allahabad, December 15-19, 2009, pp. 174-178.

[16] A. Bhise and P. D. Vyavahare, "Performance enhancement of modified turbo codes with 2-stage interleavers," *IET journal on communications*, vol. 5, no. 10, pp. 1336-1342, July, 2011.

Archana Bhise graduated from SGSITS, Indore, India in 1984, did her post graduation from VJTI, Mumbai, India in 1995 and PhD in electronics and telecommunication engineering from RGPV, Bhopal, India in 2012. She has 23 years of teaching experience in engineering discipline. Her main interests are channel codes and signal processing. Dr. Bhise is a member (ISTE), fellow (IETE) and member (IET).

Prakash Vyavahare received his M. Tech. and Ph. D. degrees from IIT Bombay in 1976 and 1995 respectively. He worked at Tata Institute of Fundamental Research, Bombay and Hindu-Hitachi scholar at Hitachi Ltd. Japan. His areas of interests include channel coding, channel modelling, cross layer design issues and secure communication. Dr. Prakash is a Senior Member (IEEE), Member (ACM), Fellow (IETE) and Fellow (IE).