Deterministic Interleaver Design for Turbo Codes

Kwame Ackah Bohulu, 1631133

22-06-2017

1 Abstract

input weight 2m error events with the distance between the bit '1' seperated by a multiple of the componet codes cycle length ($a-\tau$ seperated input weight 2m error events) tend to produce low-weight turbo codewords if present in both component codes. In this research paper, we introduce a method for designing deterministic interleavers for turbo codes in such a way that $a-\tau$ seperated input weight 2m error events in the first component encoder are not mapped unto the second component encoder. Using this method, we find good interleavers for specified frame lengths and component codes. The performance of the designed interleavers is tested against the Quadratic Permutation Polynomial (QPP) Interleaver and is shown to be better especially for long frame sizes.

2 Introduction

Turbo Codes are amongst the class of capacity approaching FEC codes that were discovered in 1993 by Claude Bearoux. They are constructed by the parallel concatenation of 2 Recursive Systematic Convolutional (RSC) Encoders via an interleavers. Diagram for

Decoding of Turbo codes is done using the Turbo Decoder. It is made up 2 Soft Input Soft Output (SISO) Decoders. The interleaver plays a very important role in Turbo codes as it reduces the number of low-weight codewords(multiplicity) of the Turbo code [4]. However, the existence of the low-weight codewords causes the turbo codes to have a high error floor in the high SNR region. A lot of research has been done concerning interleavers for turbo codes and they are generally put into 2 groups, Random and Deterministic interleavers.

Turbo codes implemented using Random interleavers have been shown to have good performance, especially for large frame sizes [3]. The disadvantage of using Random interleavers stems from required use of interleaver tables in both the encoder and decoder, which is undesirable in many applications. Deterministic interleaver on the other hand require no such interleaver tables and the logic behind interleaving and deinterleaving can be executed by means of algorithms. Due to this advantage, a lot of Turbo codes employ Deterministic interleavers in their construction. Most noticable amongst these is the Quadratic Permutation Polynomial (PP2) interleaver [3] which is used in LTE applications.

Despite this advantage, Deterministic interleavers that outperform Random interleavers, especially for large frame sizes have yet to be discovered. The aim of this research is to design an interleaver that has a performance that is at least as good as that of random interleavers for large frame sizes.

2.1 Turbo Encoding and Decoding

The Turbo encoder is shown in figure (1).

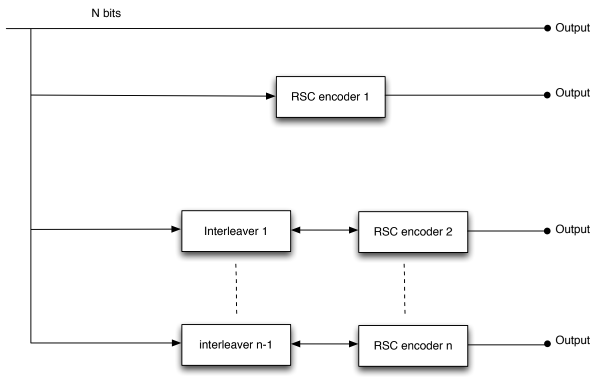

Figure 1: Turbo Encoder

The component encoders (CEs) used are identical (N+m_{RSC}, N) RSC encoders with the generator matrix given in octal form $\frac{d}{e}$ where e is the feedback vector ,d is the feedforward vector and m_{RSC} is the memory order of the RSC encoder. In our encoding scheme, we wish to return the CEs to the all-zero state.This is acheived by using m_{RSC} non-zero tail bits. The turbo encoding process is as follows. The binary input information (systematic) sequence $\mathbf{u} = \{u_1, u_2, ..., u_{N-1}, u_N\}$ of length N is fed into each CE. The upper CE receives the input as is and produces the upper parity bit sequence $\mathbf{v} = \{v_1, v_2, ..., v_{N-1}, v_N, ..., v_{N+m_{RSC}}\}$ of length N+m_{RSC} . The input to the lower CE is an interleaved version of the input information sequence $\mathbf{u}' = \{u'_1, u'_2, ..., u'_{N-1}, u'_N\}$. This is used to generate the lower parity bit sequence $\mathbf{v}' = \{v'_1, v'_2, ..., v'_{N-1}, v'_N, ..., v'_{N+m_{RSC}}\}$ also of length N+m_{RSC}. The output rate of the turbo code using this encoding scheme is $\frac{N}{3N+2m_{RSC}}$. The systematic, upper and lower parity bits are multiplexed and modulated using BPSK and transmitted over the AWGN channel.

Decoding of turbo codes is done using the turbo decoding algorithm. This algorithm is based on the iterative use of the BCJR algorithm or variations of it. In this research we use the Max-Log-APP algorithm as described in [].

Assuming BPSK modulation and transmission over the AWGN channel the a posteriori LLR values are calculated using the equation below.

$$L(u_i) = L_c y_i^s + L_a(u_i) + L_e(u_i) \tag{1}$$

variable	definition	formula	initial values
$L_c y_i^s$	$0\,0 \rightarrow 1\,0$	$1\,0 \rightarrow 1\,1$	$1\,1 \rightarrow 0\,1$
$L_a(u_i)$	$0\,0 \rightarrow 0\,0$	$0\,0 \rightarrow 1\,0$	$1\,0 \rightarrow 0\,1$
$L_e(u_i)$	$0\,0 \rightarrow 1\,0$	$1\,0 \rightarrow 0\,1$	$0\,1 \rightarrow 1\,0$

Table 1: $d_k = 0$

The meaning and definitions of all variables related to (1) are summarized in Table 1.

The Turbo decoder is shown in Figure 2. it utilises 2 Soft Input Soft Output

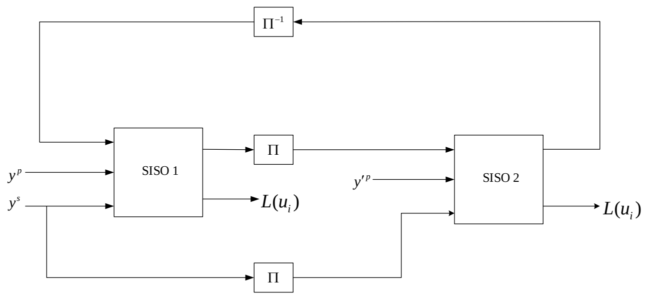

Figure 2: Turbo Decoder

(SISO) decoders(one for each encoder) connected in such a way that the $L_e(\mathbf{u})$ from one encoder is fed into the other as $L_a(\mathbf{u})$.

The turbo code transmitted over the AWGN channel is received as $\mathbf{y} = \{\mathbf{y_u}, \mathbf{y_v}, \mathbf{y_{v'}}\}$ of length $3N + 2m_{RSC}$, where $\mathbf{y_u}, \mathbf{y_v}, \mathbf{y_{v'}}$ correspond to the systematic, upper and lower parity bits respectively.

$$\mathbf{y_u} = \{y_{u_1}, y_{u_2}, ..., y_{u_{N-1}}, y_{u_N}, ..., y_{u_{N+m_{RSC}}}\}$$

$$\mathbf{y_v} = \{y_{v_1}, y_{v_2}, ..., y_{v_{N-1}}, y_{v_N}, ..., y_{v_{N+m_{RSC}}}\}$$

$$\mathbf{y_{v'}} = \{y_{v'_1}, y_{v'_2}, ..., y_{v'_{N-1}}, y_{v'_N}, ..., y_{v'_{N+m_{RSC}}}\}$$

The decoding process is as follows.

The input to the SISO1 is $\mathbf{y_u}, \mathbf{y_v}$ and $\mathbf{L_a}$. For the first iteration, it is assumed that the input information bits have equal probability and $\mathbf{L_a}$ is an all-zero vector. These are used to calculate $\boldsymbol{\gamma}, \boldsymbol{\alpha}, \boldsymbol{\beta}$ and finally $\mathbf{L}$ using (1) and $\mathbf{L_e^{(1)}}$ is obtained by subtracting $L_c y_i^s$ from each element in $\mathbf{L}$. $\mathbf{L_e^{(1)}}$ is then interleaved and fed into SISO2 as the value for $\mathbf{L_a}$ along with an interleaved version of $\mathbf{y_u}$ and $\mathbf{y_{v'}}$ which correspond to the interleaved systematic bits and the lower parity bits. These are used to calculate $\boldsymbol{\gamma}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{L}$ and finally the extrinsic LLR values of the second component decoder, $\mathbf{L_e^{(2)}}$. $\mathbf{L_e^{(2)}}$is deinterleavedand fedback into the first component encoder as the new $\mathbf{L_a}$ value for SISO1.

The process is either repeated for a predetermined number of times, or untill a certain condition is met. At the final iteration $\mathbf{L}$ (from the second component decoder) is deinterleaved and used to estimate the values of $\mathbf{u}$.

3 a-τ seperated weight 2m error events

The component encoder of choice for Turbo codes is the Recursive Systematic Convolutional (RSC) encoder. The system diagram is shown in the figure below. The generator matrix is of the form

$$[1 \ \frac{F(D)}{G(D)}]$$

where F(D) represents the feed-forward portion of the encoder and G(D) the feed-back portion of the encoder."1" represents the systematic input which is also present in the output of the code. In this paper we will describe the RSC encoders generator matrix by the feed-foward and feed-back portion and in the octal form.

Each RSC encoder has a cyclic length, which is defined as the parity output of the encoder when the input is [1,0,0,...]. For the 5/7 RSC encoder, the output is [1,1,1,0,1,1,0,1,1,0...]. We observe an output cycle of [1,1,0] which gives a cycle length(τ) of 3. Now, if a weight-2 input sequence with the "1" bits separated by aτ -1 "0" bits (a-τ separated input weight -2 error event, a=1,2,3) , a low-weight codeword is output. An example is shown in the figure below. In the case where an error event in CE1 is mapped unto CE2 by the interleaver ,a low-weight turbo codeword is produced. Since the performance of turbo codes at high SNR is dependent on the lowest weight codeword, we wish to avoid such cases. To acheive this, it is necessary to design an interleaver that prevents the mapping of a-τ separated input weight -2 error event in CE1 unto CE2. For simplicity sake, we shall refer to a-τ separated input weight -2 error event as a-τ error event from here onwards.

4 BER Performance Bounds for Turbo Codes via Union Bound

Assuming BPSK modulation and the transmission of the all-zero sequence over the AWGN channel, the union bound BER performance of a turbo code is bounded by[Proakis]

$$P_b \leq \frac{1}{N}\sum_{m=1}^{2^N-1} w_m^{(s)} Q\left(\sqrt{2R_c w_m^{(c)} \frac{E_b}{N_o}}\right)$$

where $w_m^{(s)}$,$w_m^{(c)}$,R_c , $\frac{E_b}{N_o}$are the weight of the mth information sequence, weight of the codeword generated from the mth information sequence, the overall coding rate and the bit energy to noise ratio of the channel respectively. N is the length of the information sequence.

The above equation may be written to group infromation sequences of the same weight. This gives rise to the equation below

$$P_b \leq \frac{1}{N}\sum_{m=1}^{N}\sum_{l=1}^{\binom{N}{m}} mQ\left(\sqrt{2R_c w_{ml}^{(c)} \frac{E_b}{N_o}}\right) \tag{2}$$

5 Methodology

Turbo codes have error floor in the high SNR region. This has been attributed to the prescence of low-weight codewords. The error floor of the Turbo codes can be raised by increasing the interleaver size whiles maintaining the effective free distance. Alternatively increasing the effective free distance of the Turbo while maintaining the multiplicity serves a similar purpose [2]. The effective free distance of a code is the minimum distance associated with an input of weight 2.

In RSC encoders, weight 2 inputs of the form $(1+D^{t\tau})(D^u), 0 \leq u \leq N-\tau, t = \{1, 2, 3, ...\}$ tend to produce low-weight codeword [1]. τ is the cycle length of the RSC encoder. We shall call such low-weight producing weight 2 inputs $a-\tau$-seperated input weight 2 errors. If the $a-\tau$-seperated input weight 2 errors are input into the Turbo codes component encoders a low-weight codeword will be produced.

To increase effective free distance of the turbo code, we design an interleaver in such a way that the input to the second is of the form

$$(1 + D^{c\tau})(D^u), 0 \leq u \leq N - \tau \tag{3}$$

where c is a large number.

5.1 Linear interleaver design

The mapping function for the linear interleaver is given by

$$\Pi_{\mathbf{L}_n}(i) \equiv bi \mod N,\ 0 \le i \le N \tag{4}$$

where b is a positive integer that is co-prime to N. The simplest $a-\tau$-seperated input weight 2 error (t=1) is shown in the figure below.

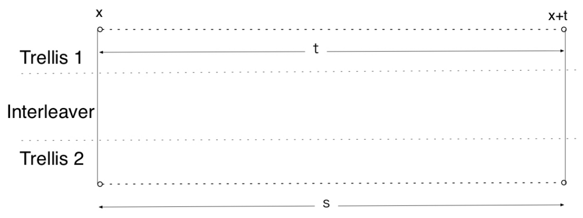

Figure 3: $t = s = \tau$

s is calculated using the equation below.

$$\begin{aligned} s &= \Pi_{\mathbf{L}_n}(x+t) - \Pi_{\mathbf{L}_n}(x) \\ &= b(x+t) - b(x) \mod N \\ &= bt \mod N \end{aligned} \tag{5}$$

The weight of the codeword can be calculated using the equation below.

$$d_{(t_i,s_j)} = w_o\left(3 + \left(\frac{|t_i|}{\tau} + \frac{|s_j|}{\tau}\right)\right) \tag{6}$$

Substituting (5) into (6) and rewriting t as τ gives

$$d_{(t_i,s_j)} = w_o\left(3 + \left(1 + \frac{b\tau \mod N}{\tau}\right)\right) \tag{7}$$

It should be noted that values of b that are considered should satisfy the condition

$$((b\tau \mod N) \mod \tau) \neq 0 \tag{8}$$

The process for choosing the value of b that changes the input to the second component encoder into the form in (3) is outlined below.

1. For a given value of b , $1 \le b \le N/2$ which satisfies (8) and all possible inputs of the form $(1+D^{t\tau})(D^u), 0 \le u \le N-\tau, t=1$, calculate corresponding s using (5)

2. Calculate the Hamming distance for the codeword using equation (6) and select min $d_{(t_i,s_j)}$

3. After min $d_{(t_i,s_j)}$ is selected for all possible values of b, the value of b which corresponds to max(min $d_{(t_i,s_j)_v}$)

The tables below show the values for b selected for $t = \tau$ and $t = 2\tau$ for various frame sizes.

Table 2: $N = 2^m$, $m = \{10, 11, 12, 13, 14\}$, $t = \tau$

b	min(s)	max(s)
511	509	515
1023	1021	1027
2047	2045	2051
4095	4093	4099
8191	8189	8195

Table 3: $N = 2^m$, $m = \{10, 11, 12, 13, 14\}$, $t = 2\tau$

b	min(s)	max(s)
427	510	514
853	1022	1026
1707	2046	2050
3413	4094	4098
6827	8190	8194

6 References

1 Oscar Y. Takeshita, Member, IEEE, and Daniel J. Costello , "New Deterministic Interleaver Designs for Turbo Codes",IEEE Trans. Inform. Theory, vol. 46,pp. 1988-2006,Nov. 2000

2 L. C. Perez, J. Seghers, D. J. Costello, Jr., "A distance spectrum interpretation of turbo codes", IEEE Trans. Inform. Theory, vol. 42, pp. 1698-1709, Nov. 1996.

3 Jing Sun, Oscar Y. Takeshita Interleavers for Turbo Codes Using Permutation Polynomials over Integer Rings, IEEE Trans. Inform. Theory, vol. 51, pp. 101 - 119 Jan. 2005

4 John G. Proakis, Masoud Salehi. ”Digital Communications”, Fifth Edition,Chapter 8, McGraw-Hill
.