Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/13S120

Error Rate Analysis of Sliding Window Algorithm in Turbo Decoding

V. Pushpa[1,∗], *H. Ranganathan*[2] *and M. Palanivelan*[3]

[1] ECE, Sakthi Mariamman Engineering College, Chennai, India
[2] Gojan School of business and Technology, Chennai, India
[3] Rajalakshmi Engineering College, Chennai, India

Received: 20 May 2019, Revised: 19Jul 2019, Accepted: 20 Jul. 2019
Published online: 1 Aug. 2019

Abstract: The performance analysis of coding is the error rate analysis which finds the probability of bit being error in the data transmission. The errors should be minimized for the better reception in the wireless communication. This paper presents a different approach of turbo decoding by implementing and sliding window algorithm in iterative turbo decoding. Sliding window algorithm is combined with turbo codes to improve the latency problem and large memory requirement. In this method the entire trellis is divided into small length sequence of several windows and the turbo decoding algorithms are applied only on those windows instead of running over the entire trellis. The performance of the coding is measured particularly by Bit Error Rate. The performance degrade in the sliding window turbo decoding algorithm is measured and it is improved by introducing proposed method of overlap sliding window algorithm is implemented in iterative decoding. By using this method the BER performance of the turbo code is improved over sliding window turbo decoding method. The number of computations is computed and compared for both sliding window turbo decoding and overlap sliding window turbo decoding method. The experimental results showed that the proposed method of overlap sliding window algorithm has better error performance than the sliding window turbo decoding method.

Keywords: Bit error rate, binary phase phift keying, parallel concatenated convolutional code, sliding window, overlap sliding window.

1 Introduction

A new class of error controlling code is the convolutional turbo code which is proposed by Berrou et al in 1993 [1]. Turbo codes substantially improve the Bit Error Rate performance. A parallel concatenated convolutional code (PCCC) with iterative decoding methods provides excellent coding gain and the BER (Bit Error Rate) performance of the code achieves near Shannon's theoretical limit for error free transmission in the data communication [2] by using non uniform interleave BER performance achieves 10^{-5} from 1 dB from Shannon limit.

Sliding window algorithm is the method of flow control for data transfer. Sliding window is used on a wide area of applications such as machine learning models [3,4]. The sliding window BCJR algorithm is proposed by Benedetto et al. In this algorithm information bits need not to be divided into blocks and the trellis termination does not required. The entire trellis length is divided into small size of blocks .So finite memory is sufficient for working with the short trellis and memory span reduces considerably. The decision depth or window size uses short trellis instead of long trellis of the entire frame sequence. The performance loss of the turbo decoder due to windowing is measured in terms of Bit Error Rate. Implementation of sliding window algorithm in the turbo decoding allows continuous decoding and this is suitable for deep space communication.

This paper is organized as follows: Section 2 provides literature survey of various methods of turbo decoding with implementing sliding window. Section 3 presents turbo encoder, and decoder structure. Section 4 explains the decoding algorithm of turbo code used. Section 5 presents the implementation of sliding window algorithm in turbo code. Section 6 describes the overlap sliding window method in iterative turbo decoding. All the simulation results are shown in section VII .The important points are concluded in section 7.

∗ Corresponding author e-mail: pushpa.paulvictor@gmail.com

© 2019 NSP
Natural Sciences Publishing Cor.

2 Literature Survey

Hyuntack Lim et al employed adaptive guard window in the normal sliding window based turbo decoder. By using this technique achieved best reduction in complex calculations for smaller window but not in frame error performance [10]. Lee, Y et al proposed a NII metric compression based sliding window to reduce the complexity calculations in the turbo decoder. The memory storage is reduced in this method but the BER performance is degraded by 0.06 dB in the lower rate of turbo decoder and it performs well at higher rate [14]. Maunder, R.G proposed a parallel method of turbo decoding based on the log BCJR algorithm. This method is suitable for the Wimax and LTE environment. The parallel mechanism is achieved through processing the odd indexed of first component and even indexed bits of second component using odd-even interleavers. This method achieves the same error rate performance with the traditional method. But, it increases the cost for the resource requirement to perform the parallel processing [15].

Roth, C et al., proposed to implement a parallel mechanism using sliding window Map algorithm on SISO decoder. It operates in all range of code rates. The SMP based architecture produce better throughput with less memory area [16]. Martina M., et al., proposed a window skipping technique for MAP of BCJR turbo decoding method. By using this window skipping technique the memory is reduced due to fewer computations with no degradation in BER. This technique is able to skip 20% of memory requirements with zero degradation in BER. [17].For cellular networks an adaptive based sliding code modulation is proposed by Kim, K.T., et al., to improve the gain of the systems. It employed a soft decoding technique in the SSCM rather than the successive cancellation technique. This soft decoding technique improved its performance by 6.4% compared to the traditional decoding in the symmetric region for 16 iterations. The traditional decoding is not be able to achieve the sum-rate pair due to the modulation schemes. [18] Liu, D, et al., addressed a cross-sliding window for reducing the latency period of 5G turbo decoder. In this approach lower latency is achieved with reasonable BER, but the BER is degraded after certain limits of latency [19].

3 Turbo Coding and Decoding Systems

Parallel concatenated convolutional code (PCCC) is a powerful channel coding method mainly used for deep space communication working at low signal to noise ratio. Turbo codes are generated by using two parallel concatenated recursive systematic convolutional (RSC) encoders with one interleaver (3GPP) as shown in Figure 1. These two encoders are separated by an interleaver and it is used to permute the input sequence of the second encoder [1]. The code generator $G(D)$ of the turbo code is given as $\left[1, G_f(D)/G_b(D)\right]$. Where $G_f(D)$ is the feed forward polynomial which is equal to $1+D+D^3$ and backward polynomial is $1+D+D^3$.

Each constituent encoder has eight states and the constraint length (K) of the encoder is four. When the switches are connected to lower position 12 tail bits from these encoder are generated to the input bits, if there are n number of bits $n+12$ bits are generated. The turbo encoder yields three different sequences of output. One output sequence as the actual information bit stream $y1$ called systematic bits plus tail bits, while the second sequences are without interleaved encoder output $y2$, which is named first parity check bits. The third sequence are the output from interleaved encoder $y3$ are the second parity check bits.

Turbo decoding is like turbo engine. The decoder output is feedback to the input and the process is repeated as the prescribed number of iteration. The turbo decoder uses iterative decoding method [5]. PCCC employing two convolutional codes as constituent codes with iterative coding algorithm provides good coding gain close to theoretical value of Shannon limit [2]. Turbo coding uses iterative decoding of Soft Input Soft (SISO) output is a maximum a posteriori MAP decoding algorithm [17]. Turbo decoder as shown in Figure. 2 consists of two identical decoders, two interleaver and one de-interleaver using with decoding rule [1]. The interleaver used at the decoder is perfectly matched with the interleaver which is used at the encoder. The de-interleaver performs the reverse operation of the interleaver. The outputs of the encoder $y1$, $y2$, $y3$ are received at the decoder as $y1'.y2'$, $y3'$ due to the presence of AWGN noise in the channel.

Decoder 1 and 2 are the SISO decoders whose input and outputs are soft values that is real values. $Y2'$ and $Y1'$ are given to decoder 1. It gives as extrinsic information as output and this is interleaved before giving to the decoder 2. The output of the decoder 2 is fed to decoder 1 through de-interleaver and this process is repeated until prescribed number of iteration is achieved. The output of the SISO consists of three LLR values. One LLR is the priori LLR of the data bit second LLR is the channel measurement made at the receiver and the LLR is the extrinsic LLR for the decoding process. The sign of LLR gives the hard decision. For the positive values of LLR decides data bit as $+1$ and negative values of LLR decides data bit as -1.

4 Turbo Decoding Algorithm

Turbo iterative decoding is proposed by C.Berrou et al which is basically a modification of the Bahl decoding algorithm. This modification is necessary due to RSC encoders. A simple algorithm weighted soft decision is

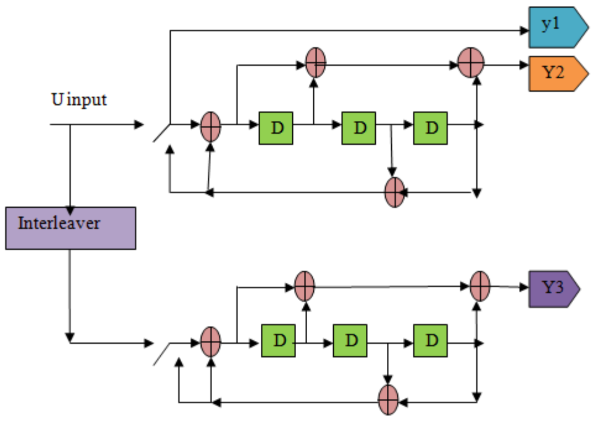


Fig. 1: Structure of the turbo encoder

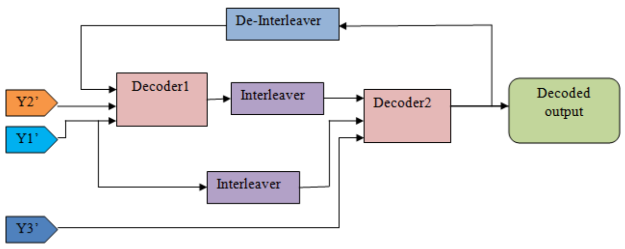


Fig. 2: Structure of Turbo decoder

used for turbo codes decoding whose complexity is twice the complexity of viterbi algorithm [1]. An optimal decoding algorithm is used to minimize the symbol error rate [6]. Here iterative decoding schemes of Log MAP and Max Log MAP are the two methods have been used for parallel concatenated codes.

4.1 LOG MAP and Max LOG MAP Algorithm

The turbo decoding method, Maximum a posteriori algorithm (MAP) is proposed by Bahl et al in 1993 [1]. The decoding method is based on trellis structure. Posteriori is an non theoretical knowledge and priori is an theoretical knowledge [9]. A posteriori probabilities of the states are estimated from the state transition. Let U be the information sequence containing zeros and ones and y be the received sequence. The Log Likelihood ratios or L values can be calculated at the decoder by Eq. (1)

$$L(U) = \log \frac{P(U = +1\,|y)}{P(U = -1\,|y)} \quad (1)$$

The decoder output can be calculated from the L value is equal to $+1$ for $L(U) > 0$ and L value is equal to -1 for $L(U) < 0$ [9].

The turbo decoding algorithms are all based on the trellis structure as shown in Figure 3. The information bits are encoded by eight states of convolution code $s = 0, 1.., 7$. The dotted line represents the branch receiving the message bit as $+1$ for bit 1 and the solid line represents the branch receiving the message bit as -1 for 0 bit. The received bit sequence can be divided into three kinds of past, present, and future sequences and these sequences are defined as state metric at time $t < 0, t = 0$ and $t > 0$, respectively. The received bit at the percent time is s state. The received bit at time $t-1$ is s' state. The integer time variable $t = 1, 2...k$. The complete received sequence comprises of three ranges of sequences that is past, present, and future. Solid line represents the surviving path. At time $t-1$ the state of the encoder $St-1 = s'$ is previous state, at time t the encoder state $St = s$ is current state and at time $t+1$ is the future state of the encoder known as next state. The beginning and end state is fixed as all zero state by the decoder. The end state of the encoder is achieved by using tail bits which makes the encoder forced to zero state and terminate the trellis sequence.

As shown in the Figure 3 $\alpha(t-1)$, $\beta(t-1)$, $\gamma(t-1)$ are the forward metric, backward metric, and branch metric at time '$t-1'$ respectively. Similarly $\alpha(t)$, $\beta(t)$,

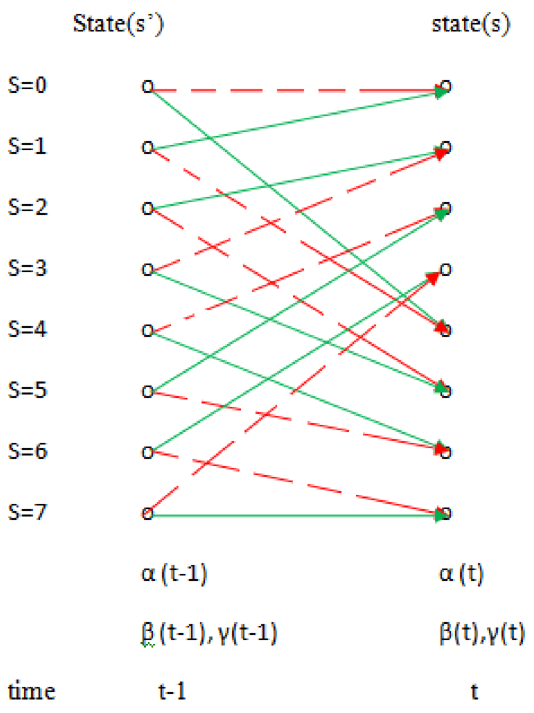


Fig. 3: Turbo code trellis diagram with eight states

$\gamma(t)$ are the forward metric, backward metric, and branch metric at time 't' respectively. The turbo decoding algorithm of log map uses exact formulas for calculating these metrics. It is the logarithmic version of BCJR algorithm. It is simpler and easy to implement. The parameters in Log map are the function of signal to noise ratio (SNR) and supports wide range of operating SNR. The computation of Log MAP and Max Log MAP algorithm [11] for 1 to N number of bits as shown in Figure 4 are calculated by the following steps

Fig. 4: Steps for computing Log MAP and Max Log MAP algorithm for N bits.

1.Computing the forward metric and branch metric from 1 to N bits
2.Computing the backward metric from N to 1 bits
3.Computing APP value from 1 to N bits

A posteriori probabilities are estimated by using dual maximum with forward and backward metric [7]. Logarithmic version of MAP reduces the number of computation and memory used by the decoder. Log MAP turbo decoding algorithm is less complex compared to MAP algorithm can be used to compute values of metrics by logarithm operation. Max function is used to compute these metric values, The Log MAP algorithm are computed by using these Eq. [10]. Forward metric $Ao(s)$ and backward metric BN are initialized as

$$\mathrm{Ao}(\mathrm{s}) = \ln(\alpha t(s)) = \begin{cases} 0, & if\ s=0 \\ -\infty, & if\ s \neq 0 \end{cases} \tag{2}$$

$$\mathrm{BN}(\mathrm{s}) = \ln(\beta t(s)) = \begin{cases} 0, & if\ s=0 \\ -\infty, & if\ s \neq 0 \end{cases} \tag{3}$$

The branch metric $\Gamma t(s',S)$ can be used in both forward and backward metric calculations. So it can be computed first. This branch metric is transition states from state s to s'.

$$\Gamma t(s',S) \ = \ ln\gamma t(s',s) \tag{4}$$

The initial state of the encoder is considered as all zero state and the forward metric probability can be calculated at time $t = 0,1,2,K-1$.

$$\begin{aligned} At(s) &= \ ln\ \alpha t(s) \\ &= max[At_{-}(s') + \Gamma t(s',s)], \end{aligned} \tag{5}$$

where $At(s)$ is the max operation is on overall previous states s' to current state s. This forward metric can be calculated as the sequence is received. It can be computed from the forward direction 1 bit to N bits. The end state of the encoder is all zero state and the backward metric probability can be calculated at time $t = k, k-1, k-2....0$ by using

$$\begin{aligned} Bt_{-1}(s') \ &= \ lnBt_{-1}(s') \\ &= max[Bt(s) \ + \Gamma t(s',s)] \end{aligned} \tag{6}$$

It can be computed after received the whole sequence and calculating from backward N bits to first bit. The branch metric max operation is on overall next states s to s'. The APP values in Max Log MAP decoding is

$$\begin{aligned} L(U/y) = \max_{(+1)}\left[A_{t-1}(s) + \Gamma t(s',s) + Bt(s)\right] \\ - \max_{(-1)}\left[A_{t-1}(s') + \Gamma t(s',s) + Bt(s)\right] \end{aligned} \tag{7}$$

A posteriori probabilities of the information sequence for trellis code can be computed using Log MAP and Max Log MAP decoding algorithms.

$$\begin{aligned} \max(a,b) &= In(\exp(a) + \exp(b)) \\ &= \max(a,b) + In(1 + \exp(|a-b|)) \end{aligned} \tag{8}$$

for Log Map

$$max(a,b) = max(a,b) \text{ for Max Log Map} \tag{9}$$

For Log MAP max function can be comprised of two terms. Max function and correction function. This correction function can be computed either by using look up table or by using a threshold detector.

5 Implementation Of Sliding Window Algorithm In Turbo Decoding

The main disadvantage of SISO map decoder is very long latency and high memory requirement. To avoid this drawback sliding window algorithm is proposed in turbo decoding method. The main purpose of using sliding window algorithm in turbo decoding is to reduce the memory requirement for the decoder [14]. The latency in turbo decoder is decreased by implementing sliding window algorithm. This will increase the number of computations and decreases the memory requirement of the decoder [12]. By employing sliding window algorithm in turbo decoding of Log MAP and Max Log MAP the entire frame length sequences are divided into a small block called window or decision depth whose width is chosen to be ten times of the constraint length of the encoder.

In sliding window algorithm the forward and backward metrics are calculated at each window instead of computing for the entire length of frame as shown in Figure 5. By combining both Log MAP and sliding window technique code words are divided into small blocks using with small memory [13]. While decoding the sequences are divided into a small block called window 'W' which is equal to ten times the constraint length of the encoder. At each window initial values of forward and backward metrics are computed. The initial value of forward metric can be calculated from the last values of α of the previous window. Then the initial value of backward metric are calculated from the β of the next window K is the sum of information bits and tail bits.

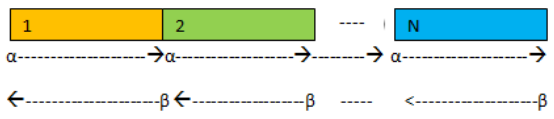


Fig. 5: computing α and β in sliding window for N bits

The original idea and computational steps of sliding window algorithm, taken from [11], are described by the following steps. The sliding window Log MAP is defined from the MAP algorithms allows continuous decoding without the requirement of trellis termination. The computation of sliding window method in Log MAP algorithm is shown in Figure 6 and calculated by the following equations.

The forward metric probability that is initialized as $Ao(s)$ and backward metric is initialised as $BN(s)$ can compute at each window defined in Eq. (2) and Eq. (3), respectively. $B_N(s) = \alpha(s)$ for all states of s The forward metric probability at $K > W$ can be computed by

$$At_{-1}(s) = max[At(s') + \Gamma t(s', s)] \tag{10}$$

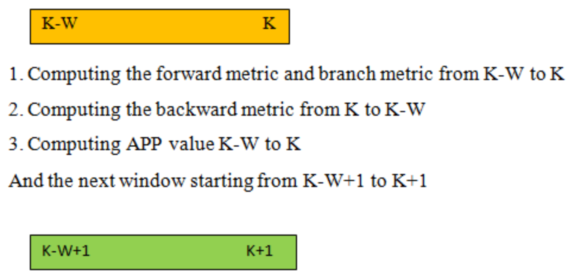
The backward metric probability is initialised at $K > W$ by

$$Bt_{-1}(s') \;= max[Bt(s) + \Gamma t(s', s)] \text{for all states } s \tag{11}$$

And branch metric for all states s is given by $\Gamma t(s', s)$ The APP $L-$value at $K - W$ can be computed by Eq. (12)

$$L(U/y) = \max_{(+1)}\left[A_{K-W-1}(s) + \Gamma_{K-W}(s', s) + B_{K-W}(s)\right] \\ - \max_{(-1)}\left[A_{K-W-1}(s') + \Gamma_{K-W}(s', s) + B_{K-W}(s)\right] \tag{12}$$

Computing APP value in Log MAP algorithm using sliding window is shown in three steps



1. Computing the forward metric and branch metric from K-W to K
2. Computing the backward metric from K to K-W
3. Computing APP value K-W to K

And the next window starting from K-W+1 to K+1

Fig. 6: Steps for sliding window in Log MAP algorithm.

At each sliding window the backward metric is computed first, then the forward metric and LLR are computed in the iterative decoding Trellis diagram for the sliding window algorithm in iterative decoding with eight states is shown in Figure 3 sold lines in the trellis represent the surviving path from time $t-1$ to t.

6 Overlap Sliding Window Algorithm In Turbo Decoding

By employing sliding window algorithm in the turbo decoding of Log MAP and Max Log MAP the entire trellis are divided into small size of block called window or decision depth. The size of the window is chosen to be ten times of the constraint length of the convolutional encoder. Each small block or window calculates the forward metric, backward metric, branch metric and computes the LLR values. The BER performance of the turbo code using sliding window algorithm does not provide good performance in the Log MAP and Max Log MAP decoding. To improve the BER performance further overlapping is done between the windows as shown in Figure 7.

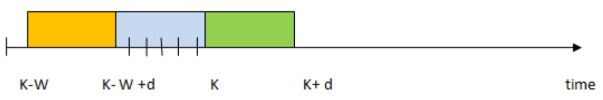


Fig. 7: Overlap Sliding Window in Log MAP algorithm, $d = 5$ bits.

The first window starting from $K-W$ to K and the second window starting from $K-W+d$ to $K+d$. All the windows are equal size of 'W' bits. Here W is the window size of small block length of received sequence which is equal to ten times of the constraint length of the convolutional encoder and 'd' refers to the overlapping width. Length of W-d is the overlapping region with the next window. Second window overlap with the previous first window with 5 bits is the overlap bits $d = 5$ bits in the blue coloured region. At each window APP values are computed for both MAP and Log MAP algorithm. The initial value of forward metric can be calculated from Eq. (2). Then the initial value of backward metric are calculated from received symbols and the branch metric value as given by Eq. (13).

$$\mathrm{B}N(\mathrm{s}) = \ln(\beta ot(s)) = max\,[Bt(s)\ \ + \Gamma t(s', s)] \qquad (13)$$

In this, the data and the interleaver data are portioned based upon the size of the sliding and overlap sliding window. This windowed data follows the procedure for decoding by calculating the branch metric values of both the algorithms separately, and bits are decoded separately for both the algorithms. Allowing one to several bits in a single decoding window reduces the complexity in computations, also the latency of decoding reduces and increases the speed of turbo decoding. The BER performance is much better and it is further improved when the width of the sliding window is increased.

The overlap sliding window algorithm in turbo decoding involves the following steps.

1.The received soft input symbols at the decoder are divided into small blocks called windows or decision depth $(W = 10XK)$ Where K is the constraint length of the convolutional encoder.
2.At each window turbo decoding algorithm is applied.
3.Initialise the forward metric at the beginning of the window. Compute the forward metric in forward direction from the received soft input symbols.
4.Initialise the backward metric at the end of the window. Compute the backward metric in reverse direction from the received soft input symbols.
5.The metric values are not stored for the initial value of the second window metric computations. No storage is required for dummy metric values of forward and backward. Dummy metric values are calculated directly from the input symbols and branch metric values.
6.The second window started forward, backward, and branch metrics computations 'd'number of bits from the previous window ('d' bits define overlap size).
7.This procedure is repeated for the entire block length.

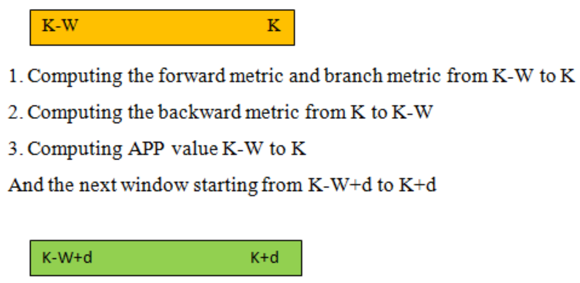

K-W
K
1. Computing the forward metric and branch metric from K-W to K
2. Computing the backward metric from K to K-W
3. Computing APP value K-W to K
And the next window starting from K-W+d to K+d
K-W+d
K+d

Fig. 8: Steps for overlap sliding window in Log MAP algorithm.

At each window forward metric (A) in the forward direction and backward metric (B) can be calculated in the reverse direction. The received soft input symbols and the priori information forward metrics are computed. Dummy backward metric (βo) can be calculated directly from the input symbols without using any memory storage. To compute backward state metric at the instant soft input symbol from previous and next input symbol is used. Backward branch metric are calculated by Eq. (11). There is no beta metric storage buffer. Dummy beta values are calculated directly from soft input symbols and branch metric values.

The APP values for trellis code can be calculated by using two equations.

$$max(a,b) \;=\; max(a,b) + ln(1 + exp(|a-b|) \qquad (14)$$

for Log MAP

$$max(a,b) = max(a,b) \qquad (15)$$

for Max Log Map

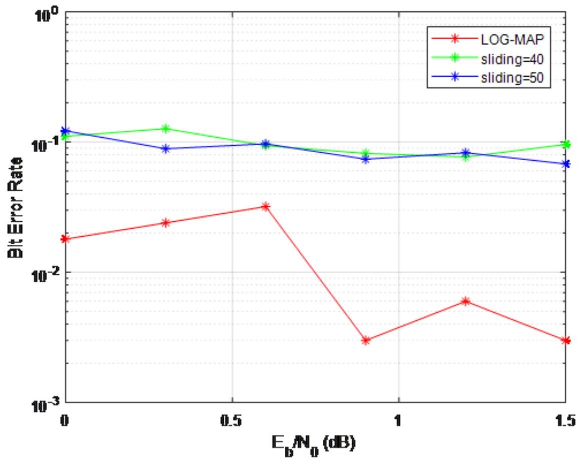


Fig. 9: BER performance of turbo code Log MAP decoding and with sliding window algorithm of window size 40 and 50 bits.$(SW = 40 and SW = 50)$

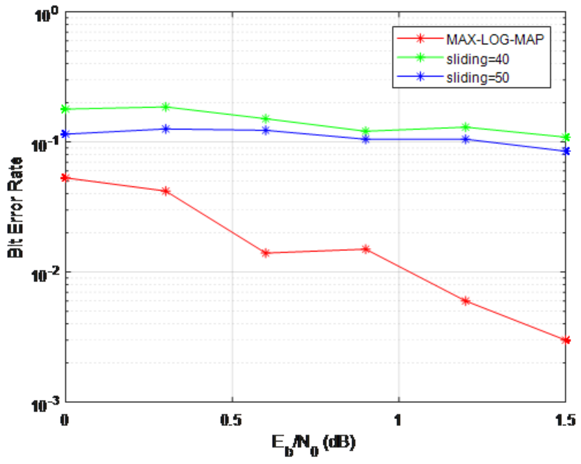


Fig. 10: BER performance of turbo code Max Log MAP decoding and with sliding window algorithm of window size 40 and 50 bits respectively. $(SW = 40 and SW = 50)$

Table 1: Performance comparison of turbo decoding Log Map Algorithm with sliding window and Overlap Sliding Window methods at Eb/No=1.5 dB

Performance parameters	LogMap	SW Log Map		OSW Log Map With window size(40 bits)		OSW Log Map With window size (50 bits)	
		Window size (40 bits)	Window size(50 bits)	Overlap size(10 bits)	Overlap size (15 bits)	Overlap size (10 bits)	Overlap size (15 bits)
BER	0.0030	0.096	0.068	0.038	0.035	0.037	0.035
Number of errors	3	96	68	38	35	37	35

The number of errors is reduced in OSW Log MAP algorithm than the SW Log MAP algorithm. This result is shown in Figure 13.

© 2019 NSP
Natural Sciences Publishing Cor.

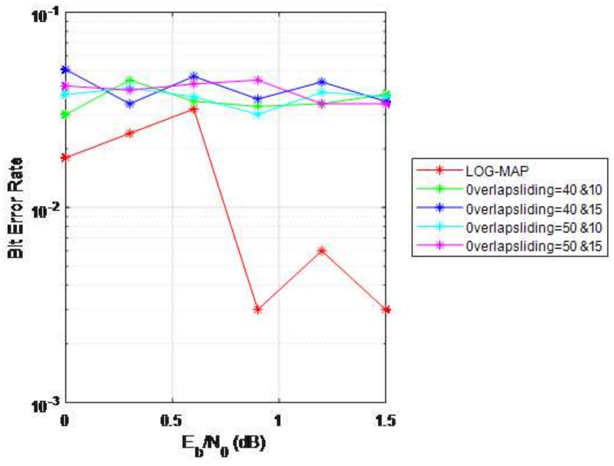


Fig. 11: shows the BER performance of turbo code Log Map decoding and with overlap sliding window algorithm. OSW $w = 40$ $d = 10$ and $d = 15$ bits, OSW $w = 50$ $d = 10$ and $d = 15$ bits *SW* (Sliding Window), *w* (window size), *OSW* (Overlap Sliding window), *d* (Overlap size)

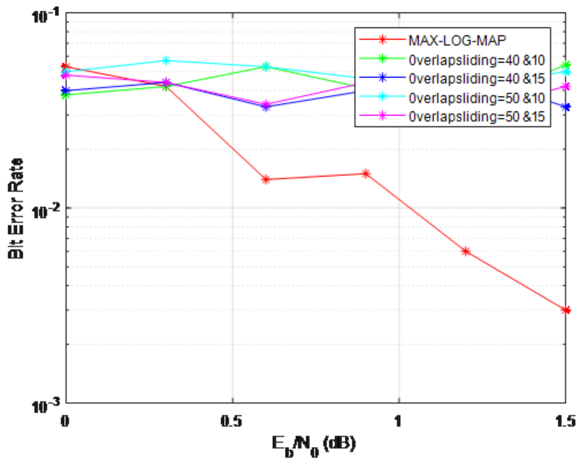


Fig. 12: shows the simulation results of turbo code Max Log Map decoding with overlap sliding window algorithms.(OSW $w = 40$ $d = 10$ and $d = 15$ bits, OSW $w = 50$ $d = 10$ and $d = 15$ bits.)

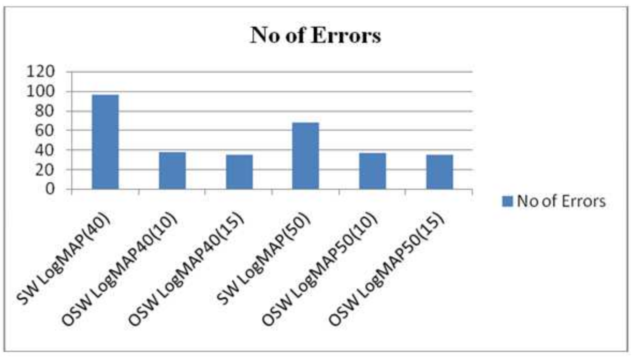


Fig. 13: BER performance comparison of Turbo code with SW Log MAP and OSW Log MAP for 1000 information bits

Table 2: Performance comparison of turbo decoding Max Log Map Algorithm with Sliding window and Overlapping Sliding Window methods at Eb/No=1.5 dB

Performance parameters	Max Log Map	SW Max Log Map		OSW Max Log Map With window size(40 bits)		OSW Max Log Map With window size (50 bits)	
		Window size (40 bits)	Window size(50 bits)	Overlap size(10 bits)	Overlap size (15 bits)	Overlap size (10 bits)	Overlap size (15 bits)
BER	0.0030	0.1090	0.0850	0.054	0.042	0.0500	0.033
Number of errors	3	109	85	54	42	50	33

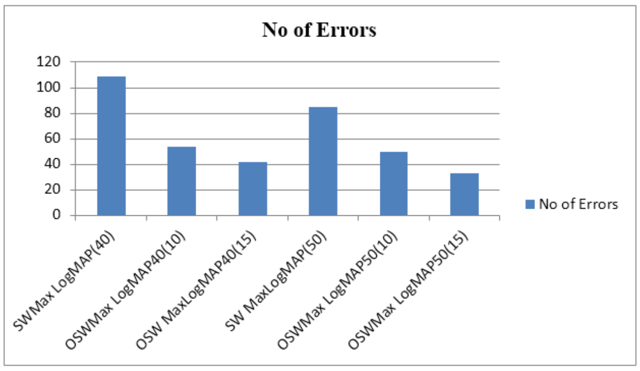


Fig. 14: BER performance comparison of Turbo code with SW Max Log MAP and OSW Max Log MAP for 1000 information bits

Performance parameters	SW Log Map		OSW Log Map With window size(4 bits)		OSW Log Map With window size (8 bits)	
	Window size(4bits)	Window size (8 bits)	Overlap size (1 bit)	Overlap size (2 bits)	Overlap size (1 bit)	Overlap size (2 bits)
Slice LUTs	3694	3694	3642	3562	3417	3424
Additions	93	85	147	97	89	89
Total No.Comp.	3787	3779	3789	3659	3506	3513

Table 3: Comparison of total number of computations between sliding window Log Map and overlap sliding window Log Map decode methods

Performance parameters	SW Max Log Map		OSW Max Log Map With window size(4 bits)		OSW Max Log Map With window size (8 bits)	
	Window size(4bits)	Window size (8 bits)	Overlap size (1 bit)	Overlap size (2 bits)	Overlap size (1 bit)	Overlap size (2 bits)
Slice LUTs	686	689	593	706	434	438
Additions	76	72	78	80	72	72
Total No.Comp.	762	761	671	786	506	510

Table 4: Comparison of total number of computations between sliding window Max Log Map and overlap sliding window Max Log Map decode methods

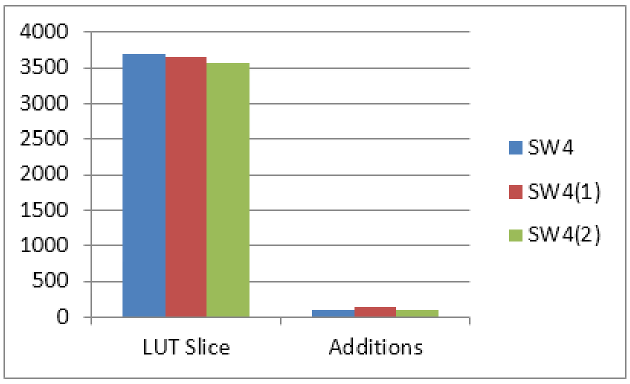


Fig. 15: Hardware utilization SW Log Map and OSW Log Map turbo decode methods for $W = 4$ bits with 1 bit and 2 bit overlap

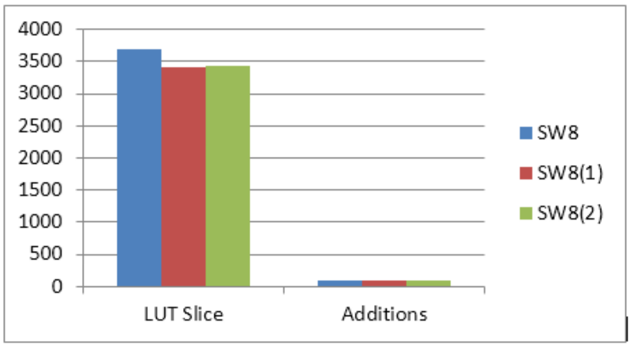


Fig. 16: Hardware utilization SW Log Map and OSW Log Map turbo decode methods for $W = 8$ bits with 1 bit and 2 bit overlap

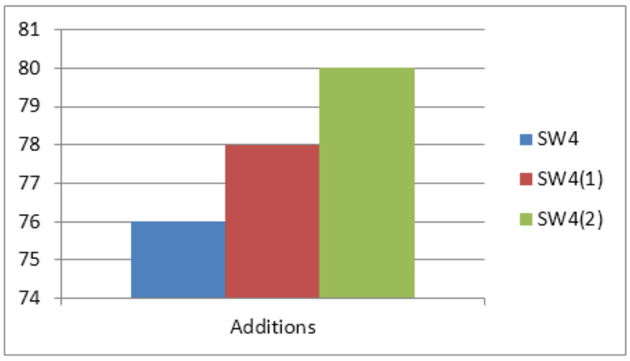


Fig. 17: Hardware utilization SW Max Log Map and OSW Max Log Map turbo decode methods for $W = 4$ bits with 1 bit and 2 bit overlap.

7 Simulation Results

The performance parameter of BER is calculated for the communication model with BPSK modulator, AWGN channel, turbo encoder and turbo decoding of Log Map, Max Log Map algorithm. The simulation results are obtained by applying sliding window algorithm in iterative turbo decoding. All the simulations are done by using a rate 1/3 PCCC with two equal recursive systematic coders having eight states and generator matrix is given by $G(D) = \left[1, \frac{1+D+D3}{1+D2+D3}\right]$ and with an interleaver of size 1000 bits is designed according to the 3GPPP standard. Each simulation is examined for 1000 bits. The performance of turbo code is evaluated in terms

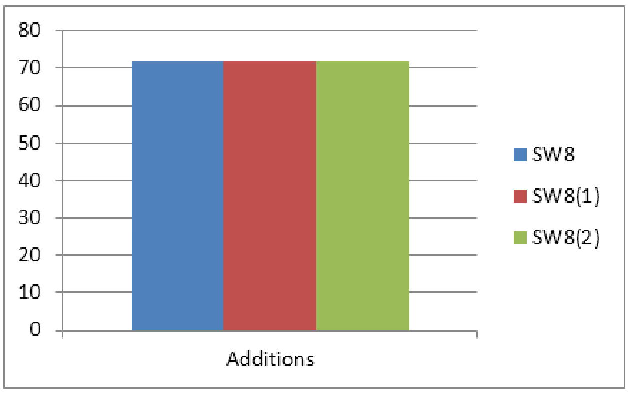


Fig. 18: Hardware utilization SW Max Log Map and OSW Max Log Map turbo decode methods for $W = 8$ bits with 1 bit and 2 bit overlap.

of Bit Error Rate (BER). BER is used to analyse the performance of the code. It is the probability of any particular bit being error in the transmission.

$$BER = \frac{\text{Number of errors at the receiver}}{\text{Total Number of bits transmitted}} \qquad (16)$$

Low BER represents less number of errors at the received bits. This is calculated by ratio of the number of errors determined at the receiver end to the total number of bits transmitted. By using the iterative method in the turbo decoding Log MAP provides the better performance than Max Log Map. Figure 9 and Figure 10 shows the simulation results of iterative decoding method of Log Map and Max Log Map with sliding window width 40 and 50 bits. Sliding window of size 40 and 50 bits BER performance is shown in Figure 9 with Log MAP turbo decoding Sliding window of size 40 and 50 bits BER performance is shown in Figure 10 with Max Log MAP turbo decoding. From the simulation results it is observed that the increasing window size improves the BER performance. When we introducing overlap in the sliding window and moving it in forward direction by increasing the overlap sizes 10 to 15 bits the BER performance are further improved. These are shown in Figure 11 and Figure 12. The values of BER at $Eb/No = 1.5$ dB are tabulated in Table 1 and Table 2 for Log Map and Max Log Map with sliding window(SW) and overlap sliding window(OSW). The research mainly concentrated on the analysis of conventional sliding window turbo decoding algorithm and proposed method of overlap sliding window turbo decoding algorithm. The window size is selected as ten times of the constraint length of the turbo encoder.

By using sliding window algorithm in Log Map turbo decoding with window size 40 the number of errors are 96. When overlap sliding window method in Log Map turbo decoding is implemented, the number of errors is reduced to 38. When the overlap size increases by 5 bits the number of errors further reduced.

When the window size is increased to 50 bits, the number of errors is 68. For the overlap sliding window Log Map method the number of errors is reduced to 37.

The number of errors is reduced in overlap sliding window algorithm compared to sliding window algorithm. When the overlap size is increased by 5 bits, the number of errors is further reduced to 35. All the results are tabulated in Table1.

In Max Log Map turbo decoding the number of errors are 3. When sliding window algorithm in Max Log Map is combined, the number of errors is 109. But in the overlap sliding window Max Log Map decoding method the number of errors is reduced to 54. That is, it is nearly half of previous method.

Increasing window size from 40 bits to 50 bits the number of errors is reduced from 109 to 85. In overlap sliding window method the increase of overlap size reduces the errors from 54 to 42. These are represented in Figure 14. When we increased the overlap size from 10 bits to 15 bits for the window size 50, the number of errors is reduced from 50 to 33. The number of errors and BER is calculated for transmitting 1000 bits and the results are tabulated in Table 2. The number of errors is reduced in OSW Max Log MAP algorithm than the SW Max Log MAP algorithm.

Table 3 and Table 4 listed the number of computations required for each algorithm. It is calculated based on the architecture design of the Max-log-map and Log-map algorithm. The input length of four (2^2) and eight bits (2^3) are given and decoded using sliding window, overlap sliding window in Log Map and Max Log Map The overlap size 1 and 2 represents the overlapping of one and two bits respectively. The term LUT denotes the lookup table which holds the information of all input and output operations. The number of additions is reduced when the window size is increased in both the sliding and overlap sliding window. But in the overlap sliding window uses only lower number of additions as compared to the sliding window approach in both the techniques. These are shown in Figure 15 and Figure 16 for the window size 4 and 8 bits with 1 bit, 2 bits overlap in Log MAP turbo decoding.

The number of computations required for Max Log Map is less compared to Log Map decoding. The total number of computations required for sliding window Max Log Map with window size of 4 bits is 762. In the proposed method of overlap sliding window in Max Log Map turbo decoding with window size of 4 bits and 1 bit overlap required number of computations is 671. The increase of overlap in sliding window from one bit to two bits requires the total number of computations to be 786.

The increase of window size to 8 bits in sliding window Max Log Map requires 761 computations. The overlap of 1 bit reduces the number to 506 computations and overlap of 2 bit requires 510 computations in overlap sliding window Max Log Map algorithm. These are shown in Figure 17 and Figure 18.

Therefore it can be concluded that the proposed method of overlap sliding window Max Log Map algorithm reduces the number of computations compared to sliding window Log Map algorithm of turbo decoding. The results are tabulated in Table 4.

The total number of computations is the sum of addition operation and number of Look up Table Values. It is concluded that without increasing much number of computations BER performance is improved in the overlap sliding window turbo decoding algorithm. Hence by using same number of computations in overlap sliding window turbo decoding performs better compared to sliding window method of turbo decode.

8 Perspective

The decoding of data is an important process in the receiver side to receive the bits as it is from the sender side. Due to the developments of turbo encoding and decoding, most of the receiver employed this technique to decode the data. The algorithm such as Log map and Max-Log-Map is implemented in this paper for the decoding of data. Both the techniques were implemented in two ways i.e., sliding window and overlap sliding window. In the sliding window, the data is processed in terms of the window by window to receive the bits. In OSW the data is processed in terms of the window by the window but the windows are overlapped based on the overlap method. Due to this overlap of windows, the decoding of data is improved which reduces the Bit error rate as compared to the sliding window approach. The decoding process by OSW produced better results when the size of the overlapping window size is increased. The utilization of devices is also less by using the overlap sliding window approach because it requires less number of LUTS and addition blocks for decoding. Hence, based on the bit error rate and the utilization of devices the overlap sliding window produced the best results when the window size is increased. With the implementation of overlap sliding window algorithm in the turbo decoding allows continuous decoding and this is suitable for deep space communication.

References

[1] C.Berrou and A.Glavieux, and P.Thitimajshima: Near Shannon limit error correcting coding and decoding turbo codes, in Proceedings of the IEEE international Conference on communications, Geneva, Switzerland), 1064-1070 (1993).

[2] S.Benedetto and G. Montorsi: Unveiling turbo codes: some results on parallel concatenated coding schemes, IEEE Transaction on Information Theory, vol.42, **2**, 409-428 (1996).

[3] A. Helwan and D. Uzun Ozsahin, Sliding Window Based Machine Learning System for the Left Ventricle Localization in MR Cardiac Images, Applied Computational Intelligence and Soft Computing, v 2017, 3048181, pp. 9 , 2017.

[4] A. Sagheer, M. Zidan and M. M. Abdelsamea, A Novel Autonomous Perceptron Model for Pattern Classification Applications, Entropy, 21(8), 763, 2019.

[5] C.Berrou and A.Glavieux: Near optimum error correcting coding and decoding: Turbo codes, IEEE Trans. Commun, vol.44, 1261-1271 (1996).

[6] L.R. Bahl, J.Cocke, F.Jelinek, and J.Raviv: Optimal Decoding of Linear codes for Minimising symbol error rate, IEEE Transactions on Information Theory, vol.IT-20, 284-287 (1974).

[7] A.J.Viterbi: An Intuitive Justification and Simplified Implementation of the MAP Decoder for convolutional codes, IEEEJournal on Selected Areas of Communication 1, 260-264, (1998).

[8] S.ten Brink: Convergence behaviour of iteratively decoded parallel concatenated codes, IEEE Trans. Commun., vol. 49, 1727-1737 (2001)

[9] Turbo Coding and MAP Decoding from www.complextoreal.com.

[10] Hyuntack Lim, Yongsang Kim and Kyungwhooncheun: An Efficient sliding window Algorithm using Adaptive - length Guard window for Turbo decoder, Journal of Communications and Networks, vol 14, **2**, (2012).

[11] S.Benedetto, D.D.ivsalar, G.Motorsi, F.Pollara: A Soft-Input Soft Output Maximum A Posteriori(MAP) Module to decode parallel and serial Concatenated Codes, TDA progress Report 42-127 (1996).

[12] Justif: Interleaver Design for Minimum Latency Turbo Coding, Diploma thesis Institute of Communications and Radio-Frequency Engineering , Vienna University of Technology, (2003).

[13] J.Gwar, S.K Shin, H.M Kim: Reduced complexity sliding window BCJR decoding algorithms for turbo code, European Signal Processing Conference, (2004).

[14] Y., Li, M., & Van der Perre, L. Memory-Reduced Turbo Decoding Architecture Using NII Metric Compression. IEEE Transactions on Circuits and Systems II: Express Briefs, 63(2), 211-215 (2016).

[15] Maunder, R. G. A fully-parallel turbo decoding algorithm. IEEE Transactions on Communications, 63(8), 2762-2775 (2015).

[16] Roth, C., Belfanti, S., Benkeser, C., & Huang, Q. Efficient parallel turbo-decoding for high-throughput wireless systems. IEEE Transactions on Circuits and Systems I: Regular Papers, 61(6), 1824-1835 (2014).

[17] Martina, M., Condo, C., Roch, M. R., & Masera, G. Computation reduction for turbo decoding through window skipping. Electronics Letters, 52(3), 202-204 (2015).

[18] Kim, K. T., Ahn, S. K., Kim, Y. H., Park, H., Wang, L., Chen, C. Y., & Park, J. Adaptive sliding-window coded modulation in cellular networks, In Global Communications Conference (GLOBECOM),IEEE, 1-7 (2015).

[19] Liu, D., Zuo, C., & Wu, Z: Benefit and cost of cross sliding window scheduling for low latency 5G Turbo decoding. In Communications in China (ICCC), 2015 IEEE/CIC International Conference on IEEE, 1-4 (2015).

V. Pushpa , received the B.Sc degree in physics from Madras University, Tamil Nadu, India in 1993, and the B.Tech degree in Electronics Engineering from Madras Institute of Technology, Chennai, Tamil Nadu, India in 1993 and M.E degree in Applied Electronics Engineering from Anna University, Tamil Nadu, India in 2005. She is currently pursuing the Ph.D. in Information and Communication Engineering at Anna University. Her research interests include digital communication, wireless communication, Error control coding methods and channel coding. She is currently working as an Assistant professor in the Department of Electronics & Communication, Sakthi Mariamman Engineering college,Chennai,Tamil Nadu,India. Author may be reached at pushpa.paulvictor@gmail.com.

Ranganathan H., received his BE in Electronics and Communication Engineering from College of Engineering, Guindy, Chennai, India in 1975 ,ME Communication system from the same institution in 1978 and PhD from Anna university Chennai. He has over 20 years of experience in providing hardware and software solutions to various customers through India. He has around 18 years of experience in education field in various levels. Currently he is Professor in ECE department of Gojan School of Business and technology, Chennai. His research interests are biomedical signal processing, RFID, Wireless networks, embedded systems, Digital image processing. He has more than 50 technical papers to his credit published in referred International Journals or proceedings of reputed International Conferences. He has been in the Program committee of many International Conferences and has been Session Chair in some of them. He is in the Editorial Board and in the review committee in International Journals of repute. He has reviewed Doctoral Theses.

M. Palanivelan, currently working as Professor & Head in the department of Electronics and Communication Engineering at Rajalakshmi Engineering College, Chennai, India. He has completed his Ph.D. at Anna University, Chennai at 2015. He received his Master Degree from College of Engineering (CEG), Anna university, Chennai at 2001 and Bachelor's degree in Engineering from MS university, Tirunelveli, India at 1995. He has published several Research papers in International Journals and Conferences. He is a Senior member IEEE and also a life Member of ISTE and ACM. Presently guiding many research scholars under his supervision. His research interest includes Peak power problems in Multicarrier modulation systems, Multiple antenna wireless communication systems, Optical Communication, Internet of Things and Signal processing.