Middle-East Journal of Scientific Research 23 (Sensing, Signal Processing and Security): 274-279, 2015
ISSN 1990-9233

DOI: 10.5829/idosi.mejsr.2015.23.ssps.108

Design of Hybrid Log-MAP Algorithm for Turbo Decoding Using QPP Interleaver

P. Deborah Roseline, G. Thavaseelan and R. Rani Hemamalini

Department of ECE, St. Peter's College of Engineering and Technology, Chennai, India

Abstract: Turbo codes are one of the most efficient forward error correcting codes (FEC) which was the first code that satisfied Shannon capacity theorem. The major drawback of turbo code is its high latency due to its iterative decoding process. The decoder consists of two SISO units parallely connected by an interleaver between them. Higher data rates can be achieved by parallel and synchronous operation of two soft input soft output (SISO) units on a data frame of uniform length. Parallel connection of SISO perturbs the functioning of the interleaver and creates a contention in accessing the memory; this bottleneck further delays the decoding process. This work proposes the adoption of an advanced interleaver, Quadratic permutation polynomial (QPP) interleaver. When interleaver is replaced with QPP the memory access contention is resolved. In this work, Hybrid Log-MAP algorithm for decoding turbo code is proposed. It approximates the exact Log-MAP accurately. The BER vs. SNR simulation were carried out to analyze the performance of the proposed system. The result of other Log-MAP based algorithm is also presented.

Key words: Turbo Encoder · Turbo Decoder · QPP interleaver · Hybrid Log-MAP Algorithm

INTRODUCTION

Turbo codes are one of the most powerful types of Forward- Error-Correcting (FEC) channel codes. Since the emergence of digital communication systems, there has been a need for error correction. This is due to the non-ideal nature of practical communication channels, which are often corrupted by noise. Error correction attempts to compensate for the errors introduced by noise. Turbo codes has been first introduced in 1993 By Berrou, Gavieux and Thitimajshima, [1] and provide near optimal performance approaching the Shannon limit. Turbo decoders suffer from high decoding latency due to the iterative decoding process, the forward- backward recursion in the maximum a posteriori (MAP) decoding algorithm and the interleaving and de-interleaving between iterations. Generally, the task of an interleaver is to permute the soft values generated by the MAP decoder and write them into random or pseudo-random positions. The Turbo encoding scheme is a parallel concatenated convolutional code with one interleaver. The function of the interleaver is to take a block of N-bit data and Produce a permutation of the input data block. This QPP interleaver is used to produce addresses recursively throughout the decoding process. The $f_c(x)$ function is the basic function and the corresponding addresses for all the SISO decoders are generated recursively.

The job of the turbo decoder [2] is to restore the transmitted data from the received systematic bit stream and the two parity check bit streams, even though these were corrupted by noise. The iterative turbo decoder consists of two constituent SISO decoders serially connected via an interleaver, identical to the one in the encoder and a corresponding de-interleaver. When data arrives, it is first stored in memory. Turbo decoder uses various iterations for decoding. Initially for the first iteration, the a-priori1 data is not available and it is set to zero. Thus, only the systematic and the parity1 data are used by decoder1 to calculate the a-posteriori 1 data. a-posteriori 1 data from decoder1 becomes a-priori2 data after interleaving and it together with parity2 data and the interleaved systematic data, are used by decoder2 to calculate a-Posteriori 2 data. Again the de-interleaved a-posterior i2 data becomes the a-priori data for decoder1 and the first iteration is finished. A high speed Turbo decoder can be realized by parallelizing several MAP decoders, where each MAP decoder operates on a segment of the received code word.

Corresponding Author: P. Deborah Roseline, Department of ECE, St. Peter's College of Engineering and Technology, Chennai, India

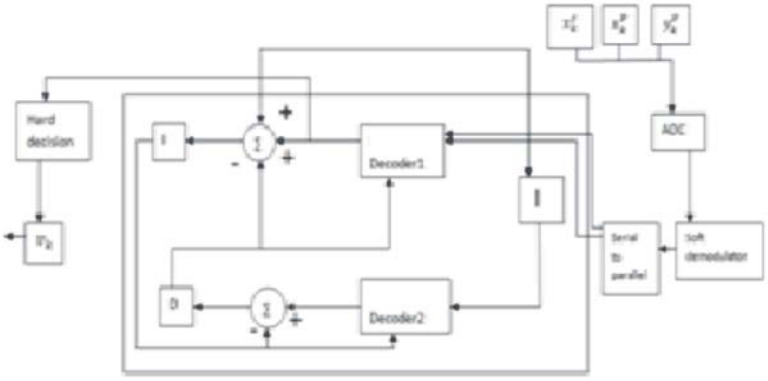

Fig. 1: Turbo Decoder

Due to the randomness of the Turbo interleaver, two or more MAP decoders may access the same collision. As a result, the decoder has to be stalled which consequently delays the decoding process.

These memory collisions can be reduced by using a parallel interleaver. Such a type of interleaver is called Quadratic Permutation Polynomial (QPP)[3] interleaver which can generate destination addresses. An enhanced QPP interleaver is proposed which is recursive in nature and it can permute the data without receiving the entire block of data. A QPP de-interleaver is just the inverse of QPP interleaver. The major advantages of turbo codes are its high BER because of the iterative algorithm used in turbo decoder. Turbo codes Come closest to approaching the shanon capacity limit on maximum achievable data transfer rate over a noisy channel. Soft In Soft Out (SISO) Decoder is used in turbo codes which enables us to get soft decisions rather than hard decisions.

The rest of the paper is organized as follows: Section II Decoding Algorithm. Section III, Approximation of the correction function. Section IV, present proposed method. In Section V, Simulation Parameter and result. Section VI, Conclusion.

Decoding Algorithm: The turbo decoding concept is functionally illustrated in Fig. 1. The decoding algorithm is called the maximum a-posteriori (MAP) algorithm and is usually calculated. During the decoding process, each SISO decoder receives the intrinsic log-likelihood ratios (LLRs) from the channel and the extrinsic LLRs from the other constituent SISO decoder through interleaving or deinterleaving. Consider a decoding process of the MAP decoder computes the LLR of the a posteriori probability (APP) of each information bit.

Log-MAP Algorithm: The MAP algorithm is too complex for practical implementation in a real system. To avoid complicated operations, the entire MAP algorithm can be computed in the log domain. By taking the logarithm of $\alpha(s_k)$, $\beta(s_k)$, $\gamma_k(S_{k-1}, S_k)$ the MAP algorithm reduces to addition and multiplication operations. However, the computation of forward state metric, $\alpha(s_k)$ = ln $\alpha(s_k)$ involves the log exponential sum which is complicated to implement in hardware:

SISO Unit: Using two SISO block in a turbo decoder, can calculate following metrics:

Branch Metric Computation Unit: In the algorithm for turbo decoding [4] the first computational block is the branch metric computation. The branch metrics is computed based on the knowledge of input and output associated with the branch during the transition from one state to another. There are four states and each state has two branches, which gives a total of eight branch metrics. The branch metrics is given by [5] as;

$$\gamma_k(S_{k-1}, S_k) = 1/2[L(u_k)x_k^s + y_k^s x_k^s + y_k^p y_k^p] \quad (1)$$

where S_k is the previous state and S_{k+1} is the next state, $x^k = (x_k^3, x^p{}_k)$ is the input/output symbol of the encoder for each branch between state S_k to S_{k+1} $y^k = (y_k^3, y^p{}_k)$ is the received channel symbol and $L(u_k)$ is the a-priori information.

Forward State Metrics: Forward recursion is given to compute the alpha coefficient.

$$\alpha(s_k) = max*[\alpha_{k-1}(s_{k-1}) + \gamma_k(s_{k-1}, s_k) \quad (2)$$

Backward State Metrics: Backward recursion is given to compute the beta coefficient.

$$\beta(s_k)^* = max^*[\beta_{k+1}(s_{k+1}) + \gamma_k(s_k, s_{k+1})] \quad (3)$$

LLR Computation Unit: To compute the extrinsic LLR value,

$$L_k = max^*_{(s_k, s_{k+1}),1}[\alpha_{s_k} + \gamma_{k+1(s_k, s_{k+1})} + \beta_{k+1(s_{k+1})} - max^*_{(s_k, s_{k+1}),0}[\alpha_{s_k} + \gamma_{k+1(s_k, s_{k+1})} + \beta_{k+1(s_{k+1})} \quad (4)$$

The max star operator employed in the above descriptions is defined as follows;

$$max^*(a,b) = \log(e^a + e^b) = \max(a,b) + \log(1 + e^{-|a-b|}) \quad (5)$$

MAX-LOG-MAP Algorithm: With the Max-Log-MAP algorithm the max* operation is loosely approximated using,

$$max^*(a, b) \approx max(a,b) \quad (6)$$

The Max-Log-MAP simplifies the Log-MAP algorithm by simply omitting the correction function fc(x) altogether. The Max-Log-MAP algorithm is the least complex of all the existing methods but offers the worst BER performance.

Approximation of the
Correction Function: This section gives a brief review of existing algorithms which approximates the correction function in order to achieve a simple implementation yet improved performance as compared to Max-Log-MAP.

Constant Log-MAP Algorithm: In this algorithm proposed by [10],the correction function should be simple and accurate,

$$f_c(x) = \begin{Bmatrix} \frac{3}{8}, -2 < x < 2 \\ 0, \ otherwise \end{Bmatrix} \quad (7)$$

$$\max^*(a,b) \approx \max(a,b) + \begin{Bmatrix} 0, b-a < T \\ C, b-a > T \end{Bmatrix} \quad (8)$$

The implication of the above rule is that the lookup table for $f_c(x)$ can be reduced to a simple logic circuit which either adds or does not add a constant to the output of the maximum selection circuit. This improve the performance of Max-log-MAP, to find a good implementation of correction function. The constant Log-MAP algorithm offers a simple implemen- tation in hardware but with trade off in performance.

Linear Log-MAP Algorithm: The Linear-Log-MAP algorithm uses the following linear approximation[11], It should be noted that the logarithmic term of equation (5) is effective when |a-b| is around zero, otherwise the effect of this term is negligible. Therefore McLaurin Series expansion can be employed to describe the logarithmic term around zero. By neglecting orders greater than one we come up with:

$$\log(1+\exp(-x)) \approx \log 2 - 1/2x \quad (9)$$

Since the logarithmic term of equation (5) is always greater than zero, we rewrite equation (9) using (10) as follows,

$$\max^*(a,b) \approx \max(a,b) + \max(0, \log 2 - 1/2(b-a)) \quad (10)$$

This approximation offers better performance than the Constant log-MAP algorithm.

Multistep Log MAP Algorithm: This approximation which is accurate and exact correction term,

$$f_c(x) = \ln 2/2^{[x]} \quad (11)$$

where x is the largest integer that is smaller or equal to x. These correction terms are more accurate than previous. They are also competitive when considering the complexity. Note that dividing by 2 is a very simple job for digital circuits. With the correction term can be obtained by shift the register storing the constant log2. The required shift time is determined by x which is in fact represented by binary integers.

Proposed Method: In this work, consider Turbo decoding, where Recursive Convolutional component codes are decoded with the five algorithms along with an advanced interleaver known as Quadratic Permutation Polynomial Interleaver.

Hybrid Method: This approximation has the advantage over the linear and multistep Log-MAP algorithm in terms of accuracy and simplicity.

$$f_c(x) = \begin{Bmatrix} 0.693 - 0.5x, for x < 1.5 \\ \dfrac{0.1693}{2^{[x]}}, otherwise \end{Bmatrix} \quad (12)$$

When plotted against the exact correction function, the hybrid approximation proves to be a better fit to the correction term. The hybrid approximation is divided into two regions i.e., |x| < 1.5 and |x| ≥ 1.5. In the region of |x| < 1.5 the hybrid algorithm employs a linear polynomial fit. The constant ln2 is replaced with the constant 0.1635 where it is the corresponding value for |x| ≥ 1.5 for the linear region. The replacement gives a double advantage. The replacement with the constant 0.1693 requires lesser number of bitwise shifts for larger values of |x| and this reduces the number of shift operations needed to perform computation for f_c(x ≥ 1.5).

QPP Interleaver: When the degree of parallelism increases in a turbo decoder, memory contention issue arises and it may lead to extra delay in the circuit. Hence there is a need for parallel interleaver,which should be capable of generating addresses on the fly for all the parallel SISO decoders.

The quadratic permutation polynomial (QPP) interleaver guarantees the desirable contention-free property for parallel memory access and has been adopted in the 3GPP LTE for turbo coding. The QPP interleaver[8] can be expressed via a simple mathematical formula. Given an information block length N, the xth interleaving output position is specified by the quadratic expression, parameters f1 and f2 are integers and depend on the block size N (0 ≤ x, f1, f2 < N). For each block size, a different set of parameters f1 and f2 are defined. In this, all the block sizes are even numbers and are divisible by 4 and 8. Moreover, the block size N is always divisible by 16, 32 and 64 when N >= 512, N >= 1024 and N >= 2048, respectively. By definition, parameter f1 is always an odd number whereas f2 is always an even number. To perform de-interleaving, when the interleaved data is read, the original location of this data can also be retrieved in the same time and become the de-interleaving destination address. QPP interleaver can support a block size from 40bits to 6144bits.

RESULTS

The Proposed turbo decoder utilizes Hybrid Log-MAP algorithm for decoding the received code word.RSC turbo encoder gives the optimum performance under AWGN channel. Binary Phase Shift Keying (BPSK) modulation has been employed. The parallel turbo decoder has been implemented in Matlab and the BER performance was studied. Simulations have been carried out for an Additive White Gaussian Noise(AWGN) channel. I have used N-1024 bits of information and varied SNR from 0dB to 2.5dB. Fig. 2. represent the correction term and its approximation. The BER performance for the HLM algorithm including Log-MAP, Max-Log-MAP, Linear Log-MAP and Multistep Log-MAP are presented in Fig. 3. The HLM algorithm is shown to have the closest performance to the exact Log-MAP solution.BER is greatly reduced.The design parameters are shown in Table 1. Time taken for each iteration is 0.3988 sec,which is quite small considering the large number of information bits shown in Table 2.

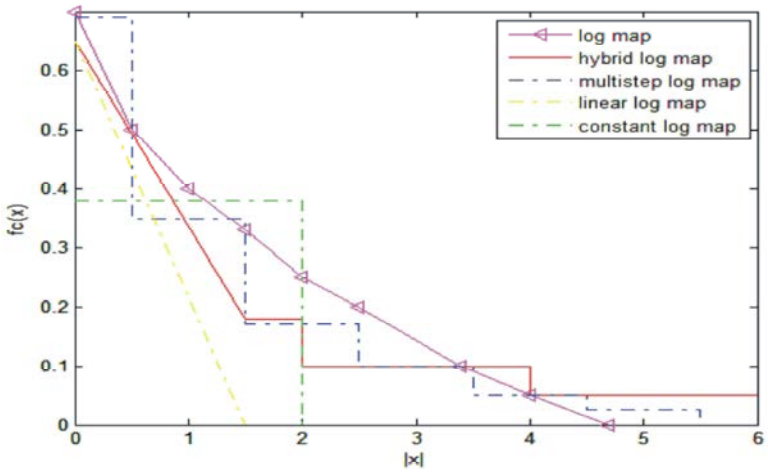

Fig. 2: Approximation Correction function

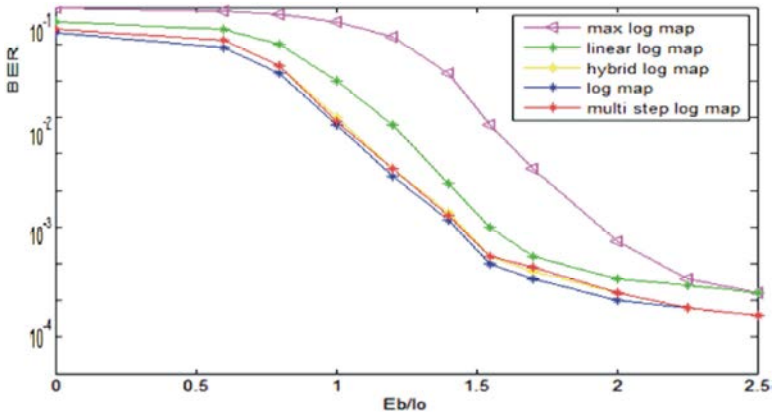

Fig. 3: BER Performance of Turbo Decoder

Table 1: Turbo Decoder Parameters

Generator matrix	[13 11]
Constraint length	3
Code Rate	1/3
Decoder Algorithm	Hybrid Log-MAP
Frame Size, N	1024
QPP function	f(x)-(31*x+64*x□2)%N
Number of iteration	1

Table 2: Comparison of parallel turbo decoder

Parameters	This work	[1]	[9]
Max.no.iteration	1	4	5
Max.block size	1024	1000	1024
speed	0.3988	3.24	9.652
Algorithm	Hybrid Log-BCJR	BCJR	Max-Log-BCJR

CONCLUSION

In this paper, a new suboptimal hybrid Log- MAP algorithm for decoding turbo code is proposed. It approximates the exact Log-MAP accurately and the Performance of the hybrid Log- MAP algorithm is shown to have the closest performance to the exact Log-MAP solution. This design was implemented using Matlab. High frame size will get better performance in turbo code system. From the simulation results for BER the performance of Hybrid Log-MAP BCJR iterative decoder is much better than the decoding using various algorithm. This algorithm achieves nearly identical performance to the Log-MAP algorithm. In addition, we also show that the Hybrid algorithm outperforms existing Log-MAP based algorithm. This paper proposes the adoption of an advanced interleaver, Quadratic permutation polynomial (QPP) interleaver which reduces delay in decoding process. Hence Turbo code is better in terms of performance, low latency, computational complexity than other codes.

REFERENCES

1. Alain Glavieux, Claude Berrou and Punya Thitimajshima, 1996. ”Near Shannon limit error-correcting coding and decoding: turbo codes” IEEE Transactions on Communications, 44: 1261-1271.
2. Cheng-chi Wong, Chien-Ching Lin Ming-Wei Lai and Hsie-Chia Chang, 2010. ”Turbo Decoder using contention free interleaver and parallel architecture”, IEEE Journal of Solid State Circuits.
3. Cheng, Chi Wong and Hsie Chiachang, 2010. ”Reconfigurable Turbo decoder with parallel architecture for 3GPP LTE system”, IEEE Transaction on Circuits and Systems.
4. Christian Benkeser, Christoph Studer, Sandro Belfat, Quiting Huang, 2011. ”Design and Implementation of a parallel Turbo Decoder ASIC for 3GPP-LTE”, IEEE Journal of Solid State Circuits.
5. Cheng-Chi, Wong and Hsie-Chia Cheng, 2011. ”High-efficiency Processing Schedule for parallel turbo Decoders using QPP Interleaver”, IEEE Transaction on Circuits and Systems.
6. Chakrabarti, I. and S.M. Karim, 2012. ”High throughput Turbo Decoder using pipelined parallel architecture and collision-free interleaver”, Communications, IET, 6(11): 1416.
7. Jienam Chen, 2013. “High Throughput Stochastic Log-MAP Turbo-Decoder Based on Low bits Computation”, IEEE Signal Processing Letters, 20(11).
8. Rahul, Shrestha and Roy P. Paily, 2014. ”High Throughput Turbo Decoder with Parallel Architecture for LTE wireless communication standard”, IEEE Transactions on Circuits and Systems.

9. Arun, C. and Aso M. Raymond, 211. ”Design and VLSI of a High throughput turbo decoder”, International Journal of computer Applications, 22(3).
10. Gross, W.J. and P.G. Gulak, 1998. ”Simplified MAP algorithm suitable for implementation of turbo decoders”, IEEE Electronics Letters, 34(16).
11. Laila, Sabeti and Shahram Talakoub, 2005. ”A Linear Log- MAP Algorithm for turbo decoding and turbo Equalization”, IEEE 2005.
12. Hao Wang, Hongwen Yang and Dacheng Yang, 2006. “Improved Log- MAP Decoding Algorithm for Turbo - like Codes”, IEEE Communications Letters, 10(3).