Modeling and Simulation of a Turbo Encoder and Decoder for Wireless Communication Systems

Sayantan Choudhury

Abstract

This report presents information about modeling and implementation of turbo codes. Turbo coding is a very powerful error correction technique that has made a tremendous impact on channel coding in the last few years. It outperforms all previously known coding schemes by achieving near Shannon limit error correction using simple component codes and large interleavers. The iterative decoding mechanism, recursive systematic encoders and use of interleavers are the characteristic features of turbo codes. The use of turbo codes enhances the data transmission efficiency in digital communications systems. This technique can also be used to provide a robust error correction solution to combat channel fading. The entire turbo coding scheme consists of recursive systematic encoders,interleavers, puncturing and the decoder. This report gives a brief overview of the various components of the turbo coding scheme, analyzes the complexities of the most popular turbo decoding algorithms, describes a suitable model of computation and discusses the various implementation methods of the *maximum a posteriori* (MAP) algorithm.

Keywords

Turbo coding, forward error correction, interleaving, puncturing, iterative decoding, MAP decoding

I. Introduction

Today's world thrives on information exchange. Hence the need of the day is that the information be protected well enough to be transmitted over a noisy environment. This is achieved by adding redundant bits to the information bit streams. If the purpose of adding redundancy is just to detect errors and inform the sender to retransmit the information, it is known as automatic repeat request (ARQ). Forward error correction (FEC) is another way of adding redundancy to the information bit stream so that errors can be detected and corrected thus preventing the need for retransmission. The price paid for adding such redundancy is a faster transmission rate in order to send the same amount of information bits per unit time implying a larger bandwidth requirement. The advantage however, is that the Signal to Noise Ratio (SNR) can be reduced significantly (also referred to as Coding Gain). In wireless systems, one of the most important performance criterion is low power transmission as that can provide a longer battery life and lesser co-channel interference.

From coding theory [1], it is known that by increasing the codeword length or the encoder memory and using "good" codes, one can theoretically approach the limiting channel capacity. However finding such a code and implementing such a decoder in real-time has been an active area of research for a very long time. Turbo codes [2], [3] is a very powerful error correcting technique, which enable reliable communication with Bit Error Rate (BER) close to Shannon limit [4]. Turbo codes are in fact a parallel concatenation of two recursive systematic convolutional codes. The fundamental difference between convolution codes and turbo codes is that while for the former, performance improves by increasing the constraint length, for turbo codes it has a small value which remains pretty much constant. Moreover, it achieves a significant coding gain at lower coding rates. An important factor for achieving this improvement is due to the "soft-input/ soft-output" decoding algorithm to produce soft decisions.

The main motivation behind doing this project is to gain a deeper understanding of turbo codes, learn ways to model it using an appropriate model of computation and to do an optimized implementation of the turbo encoder and decoder. Turbo codes enable reliable communication over power-constrained communication channels at close to Shan-

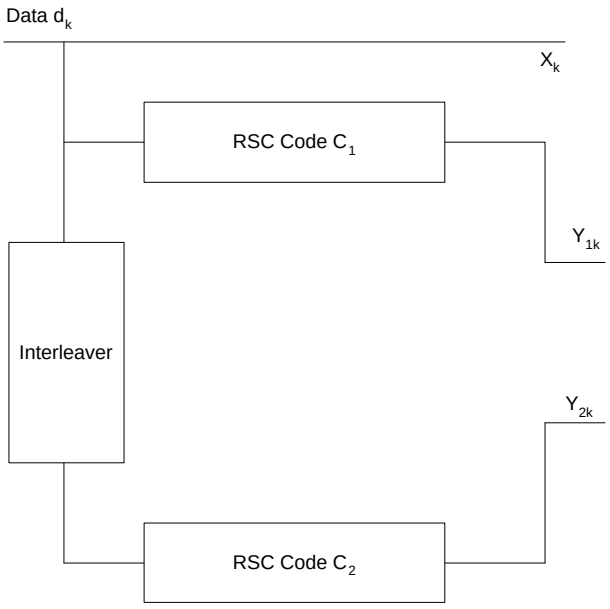

Fig. 1. General structure of a rate 1/3 turbo encoder

non's limit. However, a significant number of iterations are required to produce this result leading to higher latency. Thus efficient implementation of turbo codes in order to meet real-time constraints is an active area of research. The turbo encoder is described in Section II. Interleaver design and puncturing is explained in Section III and IV respectively. Section V gives an overview of the turbo decoder. Modeling and implementation is described in Section VI. Some important applications and standards using turbo codes are described in Section VII.

II. Turbo Encoder

The general structure of a turbo encoder is shown in Fig. 1. It consists of two rate half Recursive Systematic Convolutional (RSC) encoders C_1 and C_2. It should be noted that the trellis structure and free distance for the RSC and Non Systematic Convolutional (NSC) codes are the same. The output sequences, however, are not the same for identical input sequences. The N bit data block is first encoded by C_1. The same data block is also interleaved and encoded by C_2.The main purpose of the interleaver is to randomize bursty error patterns so that it can be correctly decoded. It also helps to increase the minimum distance of the turbo code.

The RSC code can be obtained from a NSC by adding a feedback loop and setting one

of the output bits equal to the input bits. In order to increase the transmission efficiency, puncturing can be used. Puncturing is removing certain bits from the output stream according to a fixed pattern given by a puncturing matrix.

III. Interleaver Design

As mentioned earlier, the interleaver is a very important constituent of the turbo encoder. It spreads the bursty error pattern and also increases the free distance. Thus, it allows the decoders to make uncorrelated estimates of the soft output values. The convergence of the iterative decoding algorithm improves as correlation of the estimates decreases.

The simplest interleaver is the "row-column" or "block" interleaver where the elements are written row-wise and read column-wise. The "helical" interleaver writes the data row-wise but reads diagonal-wise. There is also an "odd-even" interleaver, which has been shown to give very good results [5]. In this interleaver the odd positions of the input bits are encoded first. A pseudo-random interleaving of the input sequence follows this and the even positions are now encoded. The output consists of the input sequence and multiplexed sequence of odd and even positioned coded bits. The drawback of this scheme is that some information bits will have two coded bits associated with them while others won't have any. Thus, this causes a non-uniform distribution of coding power across the input bit stream. A solution for this problem is to use an "odd-even" type of interleaver with an odd number of rows and columns as shown in [5].

Another type of interleaver is the "Simile" interleaver [6]. It places an additional restriction: after encoding the sequences of information and interleaved bits, both the encoders must be in the same state. This allows only one sequence of tail bits to cause trellis termination. This is done by dividing the whole block of N information bits into $\nu + 1$ sequences where ν is the memory length of the code. It is seen that the sequences to which the information bits belong and not the order of the individual bits in each sequence, determine the final encoder state. Thus, as long as the interleaver doesn't change the sequence to which the original bits belong, both encoders will end in the same state and only one tail will be required to drive the encoders to all zero state at the same instant.

IV. Puncturing

Puncturing [7] is a technique used to increase the code rate. A rate 1/3 encoder is converted to a rate 1/2 encoder by multiplexing the two coded streams. The multiplexer can choose the odd indexed outputs from the output of the upper RSC encoder and its even indexed outputs from the lower one. In a more complicated system, puncturing tables are used. An important application of puncturing is to provide unequal error protection where relatively unimportant bits or during cleaner channel condition a lower rate coding is used by puncturing the coded bits while for more important bits or noisy channel conditions, higher rate coding can be used.

V. Turbo Decoder

An iterative decoding is proposed in [2], [3] which is basically a modification of the Bahl decoding algorithm [8].The modification is necessary due to the recursive nature of the encoders. The difference in this algorithm from the Viterbi algorithm [9] is that while the former produces hard outputs, this one produces soft outputs. Thus instead of outputting only 0 or 1, the output range is continuous and is a measure of the log-likelihood ratio of every bit estimate. The iterative feedback scheme is shown in Fig. 2.

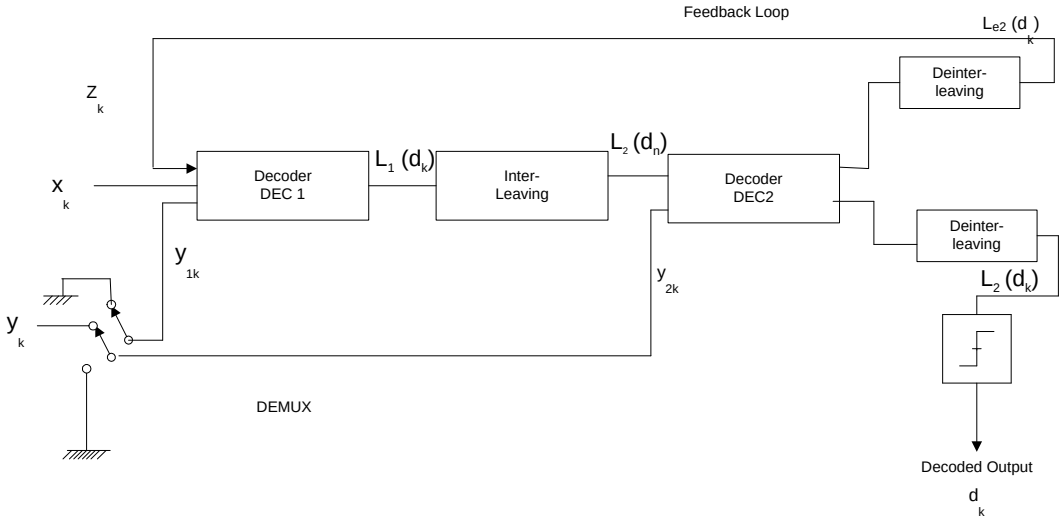

Fig. 2. Turbo Decoder

The input to the decoder x_k and y_k are the punctured encoder outputs X_k and Y_k corrupted by two independent noises with the same variance . The demultiplexer selects

y_{1k} when the transmitted sequence is Y_{1k} and selects y_{2k} when the transmitted sequence is Y_{2k} and sets it to zero for no transmission. The output of DEC1 (known as extrinsic information of the decoder) is used by DEC2 to modify the confidence levels and thus obtain a more accurate estimate of the transmitted message. The purpose of the interleaver is same as before i.e., to de–correlate the error bursts. The output of DEC2 is fed back to DEC1 and the process is repeated several times depending on the BER rate required for the application.

VI. Modeling and implementation

Due to the dataflow intensive nature of the system, a synchronous dataflow graph (SDF) [10] is ideal for modeling the system. The encoder and decoder can be modeled as SDF blocks. The individual components of the encoder/decoder blocks are implemented as SDF actors, each of which can be implemented in hardware or software. I plan to implement them using assembly language for the TMS320C6711 DSK. The block used for puncturing can be modeled using cyclo-static dataflow [11] since it does not behave as a strict SDF actor. The detailed SDF model of the turbo decoder is shown in Fig. 3. It is a homogeneous SDF graph.

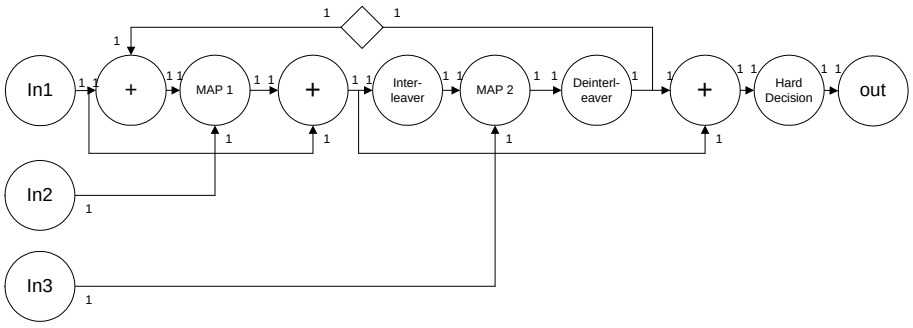

Fig. 3. Turbo Decoder SDF Model

There are two main types of soft decision decoding algorithms which are commonly used. The first one is a modified Viterbi algorithm but it produces soft outputs and hence called, Soft Output Viterbi Algorithm (SOVA) [12]. The second type of algorithm is the *maximum a posteriori* (MAP) algorithm. The detailed derivation of the MAP algorithm can be found in [13]–[15]. The direct implementation of MAP is computationally intensive and hence not feasible for real-time applications. In order to minimize the decoding

TABLE I

Decoder complexity comparison

Operation	MAP	Log-MAP	Max-Log-MAP	SOVA
add.	$2 \times 2^k \times 2^\nu + 6$	$6 \times 2^k \times 2^\nu + 6$	$4 \times 2^k \times 2^\nu + 8$	$2 \times 2^k \times 2^\nu + 9$
multipl.	$5 \times 2^k \times 2^\nu + 8$	$2^k \times 2^\nu + 6$	$2 \times 2^k \times 2^\nu$	$2^k \times 2^\nu$
max ops		$4 \times 2^\nu - 2$	$4 \times 2^\nu - 2$	$2 \times 2^\nu - 2$
table look-ups		$4 \times 2^\nu - 2$		
exp.	$2 \times 2^k \times 2^\nu$			

complexity, the logarithms of the state metrics are taken. This converts the multiplication operation to additions (Log-MAP algorithm). The problem with the Log-MAP algorithm is that now we have logarithms of sum of exponentials. This can be simplified using the Jacobian logarithm [16], [17],

$$\log(e^{L1} + e^{L2}) = max(L_1, L_2) + \log(1 + e^{|L_1 - L_2|}) \tag{1}$$

Most implementations compute the maximum term and ignore the correction factor (Max-Log-MAP algorithm). As suggested in [17], I will use a small lookup table for the correction factors.

The complexity analysis [12] for a (n, k) convolutional code with memory order ν is shown in Table I. It is evident that the Log-MAP is approximately 3 times more complex than SOVA while the Max-Log-MAP is about twice as complex as SOVA. However, the performance of MAP is about 0.5 dB better than SOVA at lower SNR and high BERs [16]. This is very important for turbo codes since the output BERs from the first stage of iterative decoding is quite high and any improvement at this stage, leads to significant overall performance improvements. A detailed comparison of the above algorithms in terms of performance, throughput, complexity and power consumption can be obtained in [18]. Another limitation of the MAP algorithm is the large memory requirement for the state metrics. Due to the limited on-chip memory in an embedded processor, I will do a sliding window implementation [19] with an overlap of $4 - 6$ times the constraint length.

VII. Applications

Applications of turbo codes include deep space communications, coding for ATM (Asynchronous Transfer Mode) and wireless applications, fading channels, digital direct broadcast satellite services, CDMA (Code Division Multiple Access), channel equalization, combined carrier estimation and decoding, wireless LAN, digital TV, cable modem, and DSL (Digital Subscriber Line) systems. Turbo codes have replaced convolutional codes in CDMA2000(3rd generation of IS-95 system) [20]. They have also been adopted in 3GPP (3rd Generation Partnership Project) WCDMA (Wideband CDMA) and DVB-RCS (Digital Video Broadcast – Return Channel Satellite) systems. NASA's next generation deep space transponder will use turbo codes [21]. Fig 4 shows that at BER = 10^{-5} the turbo

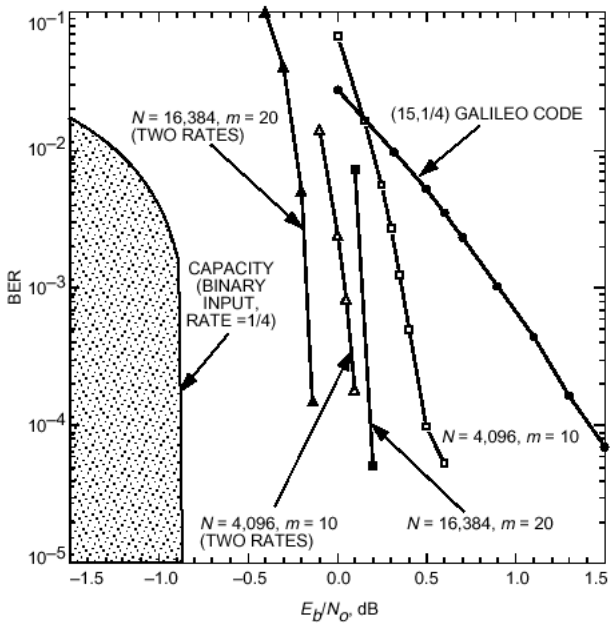

Fig. 4. Turbo code performance with code rate r = 1/4. Figure from [22].

code is better than the (15,1/4) convolutional code, developed at the Jet Propulsion Laboratory (JPL) for the Galileo mission, by 0.25 dB [22]. A more detailed list of applications of turbo codes and standards using them can be found in [23].

References

[1] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Wiley-Interscience, August 1991.

[2] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding And Decoding: Turbo-Codes," in *ICC 1993*, (Geneva, Switzerland), pp. 1064–1070, May 1993.

[3] C. Berrou and A. Glavieux, "Near Optimum Error Correcting Coding And Decoding: Turbo-Codes," *IEEE Trans. on Communications*, vol. 44, pp. 1261–1271, Oct. 1996.

[4] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp. 379–423 and 623–656, July and October 1948.

[5] S. A. Barbulescu and S. S. Pietrobon, "Interleaver Design for Turbo Codes," *Electronic Letters*, vol. 30, pp. 2107–2108, Dec 1994.

[6] S. A. Barbulescu and S. S. Pietrobon, "Terminating the Trellis of Turbo Codes in the Same State," *Electronic Letters*, vol. 31, pp. 22–23, Jan 1995.

[7] J. B. Cain, G. C. Clark, Jr., and J. M. Geist, "Punctured Convolutional Codes of Rate $(n-1)/n$ and Simplified Maximum Likelihood Decoding," *IEEE Trans. on Information Theory*, vol. IT-25, pp. 97–100, Jan 1979.

[8] L. Bahl, J. Jelinek, J. Raviv, and F. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Trans. on Information Theory*, vol. IT-20, pp. 284–287, Feb 1974.

[9] G. D. Forney, "The Viterbi Algorithm," *Proc. of the IEEE*, vol. 61, pp. 268–278, Mar 1973.

[10] E. Lee and D. Messerschmitt, "Synchronous Data Flow," *Proc. of the IEEE*, vol. 75, pp. 1235–1245, Sept 1987.

[11] S. A. Edwards, *Languages for Digital Embedded Systems* , ch. 12. Kluwer Academic Publishers, July 2000.

[12] B. Vucetic, J. Yuan, and J. Yuan, *Turbo Codes - Principles and Applications*. Kluwer Academic Publishers, July 2000.

[13] J. Hagenauer, E. Offer, and L. Papke, "Iterative Decoding of Binary Block and Convolutional Codes," *IEEE Trans. on Information Theory*, vol. 42, pp. 429–445, March 1996.

[14] S. A. Barbulescu, *Iterative Decoding of Turbo Codes and Other Concatenated Codes*. PhD thesis, University of South Australia, Feb 1996.

[15] S. Pietrobon and S. A. Barbulescu, "A Simplification of the Modified Bahl Decoding Algorithm for Systematic Convolutional Codes," in *Int. Symp. on Information Theory and its Applications*, (Sydney, Australia), pp. 1073–1077, Nov 1994.

[16] S. S. Pietrobon, "Implementation and Performance of A Turbo/MAP Decoder," *Int. Journal of Satellite Communications*, vol. 16, pp. 23–46, 1998.

[17] P. Robertson, P. Hoeher, and E. Villebrun, "Optimal and Sub-Optimal Maximum A Posteriori Algorithms Suitable for Turbo Decoding," *Europ. Trans. on Telecommun*, vol. 8, pp. 119–125, Mar-Apr 1997.

[18] C. Edwards, C. Steilzvied, L. Deutsch, and L. Swanson, "Comparison of Different Turbo Decoder Realizations for IMT-2000," *GLOBECOM '99*, vol. 5, pp. 2704–2708, Dec 1999.

[19] S. A. Barbulescu, "Sliding Window and Interleaver Design," *Electronic Letters*, vol. 37, pp. 1299–1300, Oct 2001.

[20] "Physical Layer Standard for CDMA2000 Spread Spectrum Systems." http://www.3gpp2.org/Public_html/specs/C.S0002-0_v1.0.pdf.

[21] C. Edwards, C. Steilzvied, L. Deutsch, and L. Swanson, "NASA's Deep-Space Telecommunications Road MAP," *TMO Progress Report 42-136*, pp. 1–20, Feb 1999.

[22] D. Divsalar and F. Pollara, "Turbo Codes for Deep-Space Communications," *TDA Progress Report 42-120*, pp. 29–39, Feb 1995.

[23] S.A. Barbulescu and J.A. Torres and F. Hirzel and V. Demjaneko, "Turbo Codes 2000." http://www.vocal.com/white_paper/CF-036.pdf, Jan 2001.