ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 4, April 2014

Simulation and Analysis of Puncturing in Turbo Code using MATLAB

Jitendra Prakash Shreemukh[1], Prof. (Dr.) B.S. Rai[2]

Department of Electronics and Communication Engineering , Madan Mohan Malaviya University of Technology, Gorakhpur, India[1,2]

Abstract: In this paper, we proposed some selection criteria for the puncturing vector to achieve excellent performance in terms of BER and gives useful guideline for design of puncturing vector based on simulation result. In this paper, we have compared the performance of turbo code for different puncturing location for two parity branches. Punctured turbo codes may show poor performance due to uneven error protection of input bits after puncturing. The focus of the work is on understanding and design of Punctured Turbo codes. This includes through investigation of central components that influence Turbo code performance, such as encoders and the interleaver. The investigations are carried out for transmission on additive white Gaussian noise channels. For our simulation size of interleaver selected is 1024*1024 and keeping the number of iteration 6. Simulations are conducted on MATLAB at maximum SNR range of 6 dB. In addition we also simulate the performance of punctured turbo codes with increasing the number of iteration and its effect on the performance is analyzed.

Keywords: Turbo codes, Puncturing, iterative decoding, Error floor, AWGN, MATLAB.

I. INTRODUCTION

This paper concern Turbo codes [1, 2] one of the most powerful types of forward error correcting channel codes. Forward error correcting channel codes are commonly used to improve the energy efficiency of communication systems. On transmitter side, an FEC encoder adds redundancy to the data in form of parity information. Then at the receiver side, a FEC decoder is able to exploit the redundancy in such a way that a maximum number of channel errors can be corrected. Because more channel errors can be tolerated with than without an FEC codes. Coded system can afford to operate with a lower transmit power, transmit over longer distances, avoid more interference, use smaller antennas, and transmit at a higher rate. A binary FEC encoder takes k bits at a time and produces codeword of n bits (n>k). There are 2^n possible sequences of n bits. The ratio k/n is called the code rate and is denoted by r. For every combination of code rate, codeword length, Modulation format, channel type and receiver noise power, there is practical lower limit on amount of energy that must be expended to convey one bit of information. This limit is called as Shannon capacity limit [3]. Each new generation of FEC code would perform incrementally closer to the Shannon capacity than the previous generation, as recently as the early 1990s the gap between the theory and practice for binary modulation was about 3dB. In other words, they required about twice as much energy as the theoretical minimum amount predicted by Shannon capacity limit. A major advancement in coding theory occurred in 1993, when a group of researcher developed Turbo codes. In 1993 Berrou, Glavieux and Thitimajshima [1, 2] proposed a new class of convolution codes called turbo codes whose performance in terms of Bit Error Rate (BER) are close to the Shannon limit. The initial result showed that turbo codes could achieve energy efficiencies within only a half decibel of the Shannon capacity. In now days, Turbo coding techniques are used by NASA for deep space communication, digital video broadcasting, and in UMTS. The basic idea of turbo code is to use two convolutional codes in parallel with some kind of interleaving in between. Fig. 1 shows turbo encoder consisting of two rate-1/2 binary RSC encoders, an interleaver, and a puncturing block. The simplified turbo code block diagram in Figure 1 shows only two branches. In general, one can have multiple turbo encoders with more than two branches. Length of parity bits are same as that of the information sequence and rate of turbo code is 1/3 (one input and three output sequence that is one information bits and two parity bit sequences from two RSC encoder). Turbo codes have three enhancements in the coding area. These are the random interleaver and two recursive systematic convolutional (RSC) encoders of rate 1/2. Convolutional codes can be used to encode a continuous stream of data, but in this case we assumed that data is configured in finite block size – corresponding to the interleaver size. The frame can be terminated i.e. the encoders are forced to a known state after the information block. The termination tail is then appended to the encoded information and used in the decoder. We can regard the turbo code as a large block code. The performance depends on the weight distribution – not only the minimum distance but the number of words with low weight. Therefore we want input pattern giving low weight words from the first encoder to be interleaved to patterns giving words with high weight for second encoder. One of the most interesting features of turbo code is that it is not just a single code. It is combination of two codes that work together to achieve a synergy that would not be possible by merely using one code by itself. Although the two constitute encoders may be different, in practice they are normally identical. The input data stream and parity of the two parallel encoders are then serialized into a single turbo

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 4, April 2014

code word. The interleaver is crucial part of turbo encoder. It is simple device that rearranges the order of the data bits in a prescribed, but irregular manner. Although the same set of data is present at the output of interleaver, the order of these bits has been changed. Thus output of the second encoder will almost surely be different order than the output of first encoder. Turbo code interleaver tries to randomize ordering of data in an irregular manner.

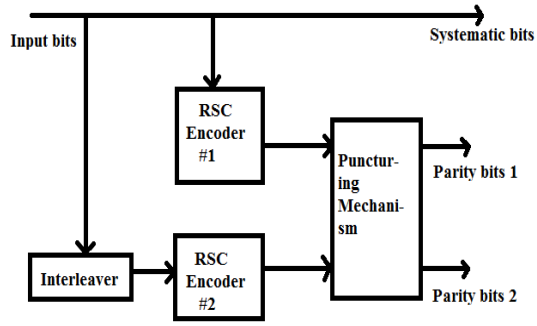

Figure 1. Basic block of turbo encoder

High weight code words are desirable because it means that they are most distinct, and thus decoder will have an easier time distinguishing among them. While a few low weight code words can be tolerated, the relative frequency of their occurrence should be minimized. Since the weight of turbo code word is simply the some of the weight of input and parity output of the two RSC encoders, we can allow one of these parity outputs to have low weight as long as other has high weight using interleaver. Because of the second encoder input has been scrambled by the interleaver, its parity output is quite different from the first encoders. Thus it is possible that one of the two encoders will produce a low weight output. Because of the interleaver probability that both the encoder simultaneously produces a low weight output is extremely small. This improvement is called the interleaver gain. Because of the interleaver gain performance of turbo codes improves. Almost any type of encoder could be used for the two constituent encoders. Turbo codes almost use recursive systematic convolutional (RSC) encoders.

II. PUNCTURED CODES

Puncturing is elimination of some bits of a codeword before of sending out it and replacing zero instead of these bits before of decoding. Puncturing is an effective technique to increase the data rate [4]. Puncturing is a tradeoff between code rate and system performance. Because due to puncturing code rate improves but the same time code performance degrades due to less number of information bits transmitted. Using the technique of puncturing, it is possible to provide different turbo-codes of various rates. The codes obtained after puncturing gives performance very nearer to those obtained with optimal codes (un-punctured codes) of the same rate. In certain application such as satellite communications, link reliability is of prime importance and, consequently, low rate codes are often used. However, bandwidth occupancy is of much greater importance in wireless communications and so high rate codes are preferred. A high rate binary convolutional code can be obtained by periodic elimination, known as puncturing, of particular code bits from the output of a parent low rate convolutional encoder. Extensive analysis on punctured convolutional codes has shown that their performance is always inferior to the performance of their low rate parent codes [5, 6]. Rates higher than 1/3 can be obtained by periodic elimination of specific code bits from the output of a rate-1/3 turbo encoder. Some of puncturing patterns are unsuitable and degrade the performance of turbo code. Unsuitable patterns are those patterns which make impossible convergence of iterative decoding and recovering of message because of inordinate elimination of "1" bits of turbo coded sequence. It is clear that both puncturing pattern and interleaving pattern affect the performance of turbo codes [7, 8]. Puncturing scheme has been applied to turbo codes to increasing the code rate without increasing the complexity of the decoder. Even through some combination of puncturing pattern and interleaving pattern can lead to non-optimal performance of turbo code.

III. SOME NOTES on DECODING

In the case of turbo codes, there are two decoders for outputs from both encoders. Both decoders provide estimates of the same set of data bits, albeit in a different order. If all intermediate values in the decoding process are soft values, the decoders can gain greatly from exchanging information, after appropriate reordering of values. Information exchange can be iterated a number of times to enhance performance. At each round, decoders re-evaluate their estimates, using information from the other decoder, and only in the final stage will hard decisions be made, i.e. each bit is assigned the value 1 or 0. Such decoders, although more difficult to implement, are essential in the design of turbo codes. One of the novel attributes of turbo codes is their ability to compose ‘larger codes’ that can be coded with reasonably low complexity. This is achieved by iterative decoding the two constituent codes that together compose a turbo code. The block diagram of an iterative decoder is shown in figure 2. It consist two constituent decoder, one for each constituent code, and the interleaver/ deinterleaving blocks required to convert the sequences between the code spaces. Each decoder processes input blocks of size N, i.e. size of the interleaver. After the first decoder has perform its decoding using the received channel symbols associated with the first code, it passes a block of soft information of length N to the second decoder. Next the second decoder uses the information from the first decoder together with the received channel symbols associated with the second code. Hopefully the second decoder performs better than the first, since it has access to more information. Further, if the first decoder is presented with results from the second decoder it is conceivable that it might improve its performance, compare to first decoding attempt. Thus, in the second decoding round of the first decoder, it uses the same channel information as in the first round, together with the information passed from second decoder. One decoding iteration is completed after one pass of both the first and second constituent decoders. The decoding

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 4, April 2014

performed by one constituent decoder is referred to as a half-iteration. Ideally, the information passed between the constituent decoder should consist of prior knowledge of the probability distribution of each in the information sequence. As such, the decoder inputs used as a priori information should depend only on transmitted information sequence and not on the noise on the other decoder inputs.

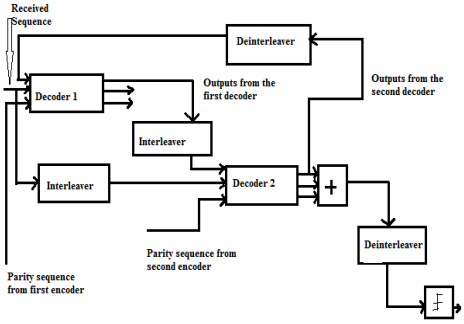

Figure 2. An Iterative turbo code decoder

By using decoders producing a posteriori probabilities so called APP or soft-output decoder. The posteriori probabilities after decoding the first constituent code can be used as a priori input when decoding the second constituent code.

III. SIMULATION RESULTS

In this section MATLAB simulations are shown. The performance of turbo codes is evaluated in terms of BER. The main tool for the performance evaluation of punctured turbo codes is computer simulation. Computer simulation generates reliable probability of error estimates as low as 10^{-6} given by Shannon capacity limit. Computer simulation is useful for rather low SNRs since the error probabilities for larger SNRs are difficult to simulate. All simulations were taken with MATLAB. The number of iterations between the decoders was set at 6. In our simulation, the input data frame size is 1024 bits, i.e. the size of punctured turbo code interleaver. Channel model used in simulation is AWGN channel. The decoder implementation is complex and computationally extensive. It includes processing using a number of loops. The decoder decodes iteratively checking the number of errors after every iteration. If the number of errors is zero for iteration, the code will not execute the next iteration to decrease processing load. The maximum range of SNRs for simulation in dB is 6 and code rate of turbo codes is 1/3.

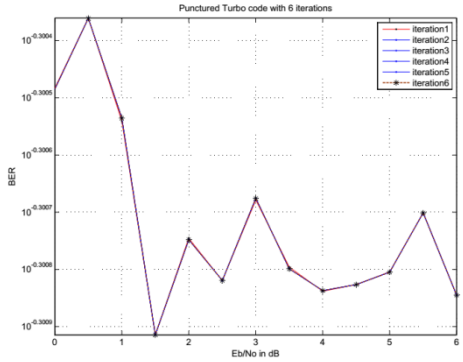

Figure 3(a): BER versus Eb/No in dB for turbo code under puncturing arrangement [010001] for both parity branches

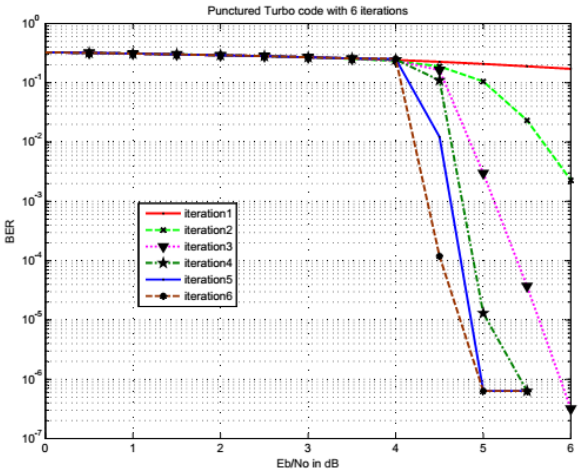

Figure 3(b): BER versus Eb/No in dB for turbo code under puncturing vector arrangement [011100] for parity bits

Figure 3(a) shows the bit-error rate versus signal-noise ratio plots for interleaver size 1024*1024, R= 1/3 and number of iteration=6 with puncturing vector [010001]. Figure 3(b) shows a similar graph but was simulated for interleaver size of 1024*1024 and puncturing pattern is [011100] with maximum range of SNR is 6 dB.

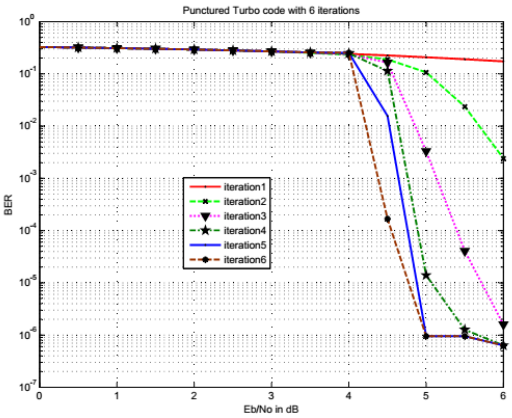

Figure 3(c): BER versus Eb/No in dB for turbo code under puncturing arrangement [101010] for parity branches

In Figure 3(c) and Figure 3(d) the performance of turbo code is compared at interleaver size of 1024*1024 and code rate 1/3 with puncturing vector [101010] and [010101] respectively. Puncturing can be applied to the systematic bits as well as to the parity bits or both. But puncturing the information bits causes serious degradation in system performance [10]. Hence it is recommended to avoid puncturing information bits and keep puncturing the parity bits. In our simulation, we use puncturing vector of period 6. In figure 3(c) and 3(d) puncturing pattern is equally distributed between two parity branches and also maximally scattered within each branch. In figure 3(b) puncturing vector is equally distributed but not well scattered. The puncturing vector used in figure 3(a) is an extreme case of puncturing in which puncturing is neither equally distributed nor well scattered. The puncturing vector used in figure 3(a) for simulation delete maximum number of bits from the two parity branches among all the combination used in our simulation. Among all the simulation result, figure 3(c) and figure 3(d) shows the

ISSN (Online) : 2278-1021
ISSN (Print) : 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 3, Issue 4, April 2014

best performance. Performance of puncturing vector used in figure 3(b) is slightly inferior with number of iteration as compared to figure 3(c) and 3(d). Figure 3(a) shows much worse performance among all. The punctured vector used for simulation result in figure 3(c) and 3(d) are one cyclic shift of each other, and their performance are close enough to each other for few iteration and hence we can conclude that cyclic shifting the puncturing vector have essentially the approximately same performance. In our simulation analysis we observed that when number of iteration increases the performance improves but after some iteration there is no significant improvement in BER.

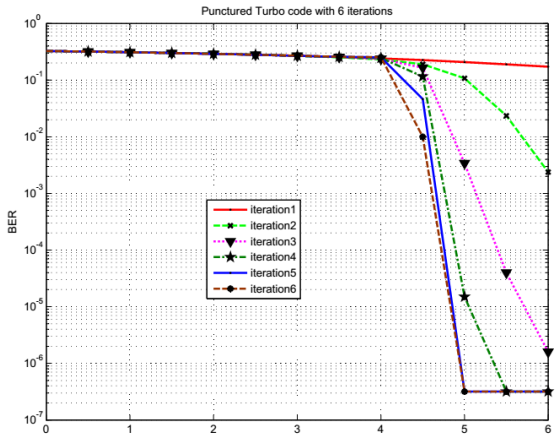

Figure 3(d): BER versus Eb/No in dB for turbo code under puncturing arrangement [010101] for parity bits

IV. CONCLUSION

In this paper performance of punctured turbo code has been compared in terms of BER by varying puncturing pattern for two parity branches. Performance of turbo code improves by puncturing when puncturing vector is equally distributed and maximally scattered. For this case we achieved excellent performance of turbo code. Performance of turbo codes decreases when puncturing vector is not equally distributed or not maximally scattered. For this case performance of turbo code shows some degradation. We also conclude that the one cyclic shift of puncturing vector is not going to affect performance much poorer. Our investigation leads to provide a set of useful guideline for constructing efficient puncturing vector to achieve acceptable performance. The turbo code performance improves with the increase in the number of iterations. However, the rate of improvement decreases. Hence, after a specific number of iterations, the BER stays constant and does not decrease any further. Thus, the number of iterations should be kept such as to avoid extra computations. Thus, upper bounds need to be specified for the number of iterations. We believe that the guideline for puncturing vector achieved in this simulation work will be useful for further research in field of turbo coding.

REFERENCE

[1] C. Berrou, A. Glavieux, P. Thitimajshima, "Near Shannon limit error correcting coding and decoding: Turbo-codes", Proc.ICC'93, Geneva,Switzerland, May 1993, pp. 1064- 1070.

[2] C. Berrou, A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo-Codes", IEEE Transactions on Communications, Vol. 44, No. 10, October 1996.

[3] C. E. Shannon, "A Mathematical theory of Communication," Bell System Technical Journal 27 (July, Oct 1948): 379-423, 623-56.

[4] J.Hagenauer, E. Offer, L. Papke, "Iterative Decoding of Binary Block and Convolutional Codes", IEEE Trans. Inform.Theory, vol. 42, no. 2, pp. 429-445, March 1996.

[5] J.Hagenauer, "Rate compatible punctured convolutional codes and their applications,"IEEE Trans. Commun., vol. 36, no. 4, pp. 389–400, Apr.1988.

[6] D. Haccoun and G. B´ egin, "High-rate punctured convolutional codes for Viterbi and sequential decoding," IEEE Trans. Commun., vol. 37, no. 11, pp. 1113–1125, Nov. 1989.

[7] Y. Kim, J. Cho, W. Oh and K. Cheun, "Improving the performance of turbo codes by repetition and puncturing," Project Report, Division of Electrical and Computer Engineering, Pohang University of Science and Technology.

[8] D. Divsalar, S. Dolinar, and F. Pollara, "Transfer Function Bounds on the Performance of Turbo Codes," TDA Progress Report 42-122, Aug. 1995, Communications Systems and Research Section, R. J. McEliece California Institute of Technology, pp. 44-55.

[9] Narushan Pillay and HongJun Xu, " Dual-Repeated- Punctured Turbo Codes on AWGN channels" in IEEE AFRICON 2009.

[10] BERROU, C., and GLAVIEUX, A,: 'Near optimum-error correcting coding and decoding: turbo-codes', IEEE Trms. Cornrnun., 1996, 44,pp. 1261-1271

BIOGRAPHY

Jitendra Prakash Shreemukh is M. Tech (Communication Engineering) student at Madan Mohan Malaviya University of Technology, Gorakhpur, India. He did his B. Tech (ECE) from IIMT college of Engineering, Greater noida, UP. He has published 4 papers in International/National Conference/ Journals. He has received third best paper award in National Conference. His research interests include wireless communication and Coding Theory.

Dr. B. S. Rai. is a professor in department of electronics and communication engineering at Madan Mohan Malaviya University of Technology, Gorakhpur, India. He holds a B. Tech and M. Tech from University of Allahabad, Allahabad and Ph. D from D.D.U. University, Gorakhpur. He has teaching experience of about 36 years. He has published about 50 papers in International/National Journals/ Conference. He is currently a member of IEEE, IE, IETE, ISTE. His research interest includes Signal processing, Embedded System, Coding Techniques and Communication System.