Comparison of Decoding Algorithms for Concatenated Turbo Codes

Drago Žagar, Nenad Falamić and Snježana Rimac-Drlje
University of Osijek
Faculty of Electrical Engineering
Kneza Trpimira 2b , HR-31000 Osijek,
CROATIA

Abstract – This paper deals with a comparison of decoding algorithms for concatenated turbo codes. For this purpose we have implemented an algorithm of the concatenated convolutional turbo code in Matlab© program. The algorithms used for decoding of turbo codes were Log-MAP and SOVA. To perform measurements we have simulated different communications conditions in the implemented model. We have evaluated the impact of the Signal/Noise ratio and the frame length on bit error rate (BER) for different decoders and finally compared the analysed decoding methods with theoretical limits.

Key-Words:- Turbo code, Log-MAP, SOVA, BER, Signal/Noise ratio

1 Introduction

Turbo codes have introduced great improvements in efficiency in comparison to other error correcting codes. Turbo codes work very close to Shannon limits (difference below 0.5 dB) [3].

We can find two basic types of turbo codes: a parallel concatenated convolutional turbo code CRSC and a block turbo code. By the CRSC code the output of the coder is cyclically turned back to the input to repeat a procedure of error checking. By increasing the number of iterations we get better and more accurate results. This technique is also called SISO (Soft Input/ Soft Output). CRSC codes are accepted and adopted for space telemetry, the third generation of mobile phones and satellite TV [2].

The Block turbo code is represented by as non-repetition convolutional decoder. This code has worse characteristics than CRSC, especially by the low code rate, but at higher code rates it is more efficient. The main advantage of block turbo codes over CRSC is an easier implementation, and consequently it was used for broadband satellite transmission. [2].

The concatenated codes consist of two independent codes that are interconnected to build a new code. The concatenated codes could be built in series or in parallel.

Concatenation is a process of assembling two code words $\boldsymbol{c_1}$ and $\boldsymbol{c_2}$ into one common code word consisting of elements of both codes in particular order. If we have code word $\boldsymbol{c_1} = \{000;\ 001;\ 110;\ 111\}$ and $\boldsymbol{c_2} = \{000;\ 100;\ 011;\ 111\}$ then there follows the common code word $\boldsymbol{c\_} = \{000000;\ 001100;\ 110011;\ 111111\}$.

To improve the abilities of code error correction and statistical properties we use an interleaving device. The turbo coders use parallel concatenation, and decoders are based on serial concatenation. The serial concatenated decoders are used because they could share information, while the parallel concatenated decoders work independently [6].

The decoding algorithms based on ML (Maximum Likelihood) find out the most probable information sequence, while the MAP (Maximum Aposteriori Probability) algorithm finds out the most probable information bit in decoding sequence.

A SOVA algorithm (Soft Output Viterbi Algorithm) uses a decoder, which gives a real number as an output. This number roughly represents the probability of a bit error. The SOVA decoder uses finite state machine to find out the information sequence.

2 Turbo coder

The turbo coder consists of two identical Repetition Systematic Convolutional RSC coders with parallel feedback. The coders are separated by an interleaver and we use only one systematic output, while the second one is a permuted systematic output.

The repetition systematic convolutional coder is realised by feedback to the non-repetition convolutional coder. The repetition convolutional coder usually produces code words weightier than the non-repetition one. That results in a decreased number of code words with lower weight, but with better error protection.

Both codes have the same minimal distance, but they differ in Bit Error Rate BER, because it depends on

relations between an input and an output of the respective coder. It is practically proved that by low signal/noise ratio E_b/N_o BER is lower by the repetition code than by the non-repetition one. [3].
The interleaver is used by turbo codes to introduce randomness in the input sequence and to increase the weight of the code words. The interleaver influences directly the distance between the code words, and consequently we could improve BER by avoiding the usage of code words with low weight.
The objective of puncturing by turbo codes is a periodical erasure of particular bits to decrease the system load by coding (the extra bits that have to be processed and transmitted). The puncturing device erases bits with probability λ, while the remaining bits are $(1-\lambda)n$. This concept is especially suitable for convolutional codes and gives the optimal characteristics to adjust the code rate by multiplexing the information into the transmission channel. Two of the most important features by puncturing are: minimal puncturing of the systematic bits and equal puncturing of parity bits of both coders. Puncturing of systematic bits is avoided because they are more important than parity bits and their puncturing causes fading of code characteristics.

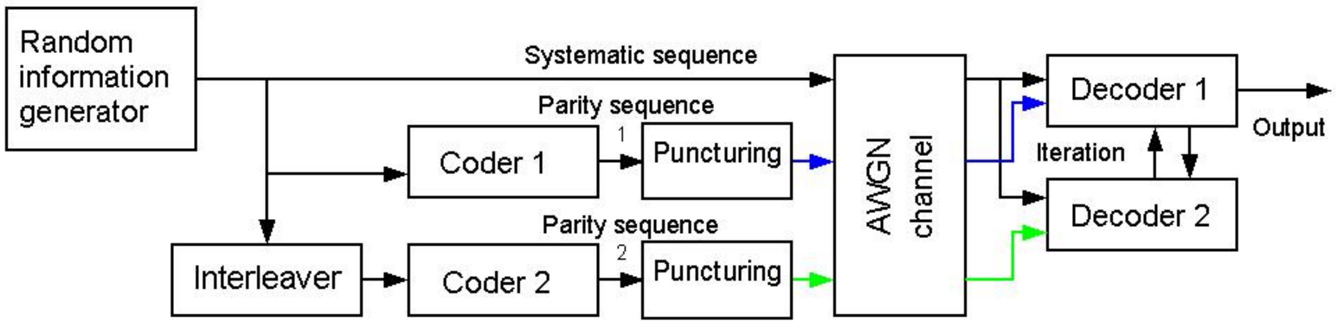

Figure 1. Block diagram of the implemented turbo code

3 Matlab© simulator of turbo code

Figure 1 shows a block scheme of the turbo coder/decoder, realised in Matlab application. An input is randomly generated for all measurements, and an interleaver mixes the input bits of coder 2. Both coders produce parity bits that could be optionally punctured. In an AWGN channel the signal is mixed with a noise distributed by Gaussian, and finally it comes to the input of two decoders. The decoders decode the signal by the chosen decoding algorithm and share the information depending on a desired number of iterations.
The possible options that the program offers are:

- Decoding algorithm – Log-Map or SOVA
- Frame length
- Generator sequence
- Puncturing
- Number of desired iterations for every frame
- Signal/noise ratio (E_b/N_o)

For the measurement purpose we have used two different decoding algorithms: Log-MAP and SOVA with a frame length of 100 and 800 bits. The sequences are punctured at the rate of ½ . The program gives results in frame errors (FER) and bit errors (BER) for each iteration.

3.1 Decoding TC with Log-MAP algorithm

The MAP (Maximum Aposteriori Probability) algorithm is introduced in 1974by Cocke, Jelinik and Raviv Bahl . This algorithm is used for the prediction of the most possible information bit that was sent in code sequence. The Log-Map algorithm is MAP shifted into a logarithmic domain. This algorithm belongs to SISO (Soft – Input, Soft – Output). Even though the MAP algorithm offers a better performance than the Viterbi algorithm, it was not considered till recently, because of the sheer simplicity of the Viterbi algorithm. Also, the differences in error performance between Viterbi and MAP algorithms were not much different at low BER's [2][4].
But recently, the introduction of Turbo codes have brought about an increased interest in this algorithm because of the following reasons:

- its superior performance under low Eb/N_0's and high BER's.

- it is an inherently Soft - Input, Soft - Output algorithm (SISO algorithm) and it very well suited for Iterative Decoding (as it is used in Turbo codes).

3.2 Decoding TC with SOVA algorithm

An ML algorithm (e.g. Viterbi algorithm) is an inherently hard output algorithm and has to be modified to provide soft outputs. Modification resulted in the Soft Output Viterbi Algorithm (generally called SOVA), which is approximately twice as complex as the Viterbi algorithm (but not as complex as the MAP) [1][5].
The SOVA algorithm uses a decoder that gives at its output a real number, which represents the bit error probability. The SOVA decoder uses a finite state machine, by which every binary input is connected with a binary output (i.e. an extra coder's output). In such way we have got the input information and extra bits in the decoder.

4 The measurements

4.1 Decoding TC with Log-MAP decoder

I. Measurement: Log-MAP, parity bits punctured, frame length 100.

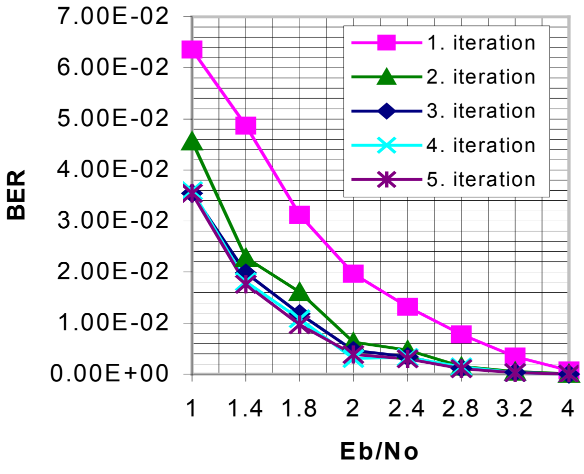

Figure 2. Dependence of BER and Eb/No for each of 5 iterations by Log-MAP algorithm, frame length 100

Figure 2 shows the dependence of BER – Eb/No for a different number of iterations by frame length 100. Obviously after 5 iterations we obtain very similar results and further computation is usually not necessary, especially when the additional processing delay is not desirable.

II. Measurement: Log-MAP, parity bits punctured, frame length 800.

Figure 3 shows the dependence of BER – Eb/No for a different number of iterations by frame length 800. Here we can notice that for the increased frame length, especially by a lower E_b/N_0 ratio, BER is lower.

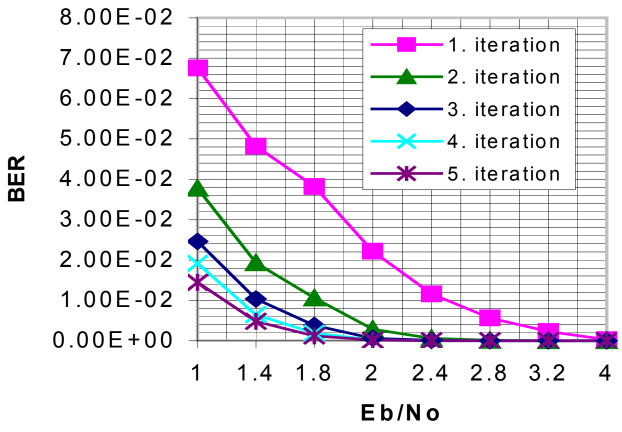

Figure 3. Dependence of BER and Eb/No for each of 5 iterations by Log-MAP algorithm, frame length 800

III. Measurement: Log-MAP, parity bits punctured, 5th iteration, frame length 100 and 800.

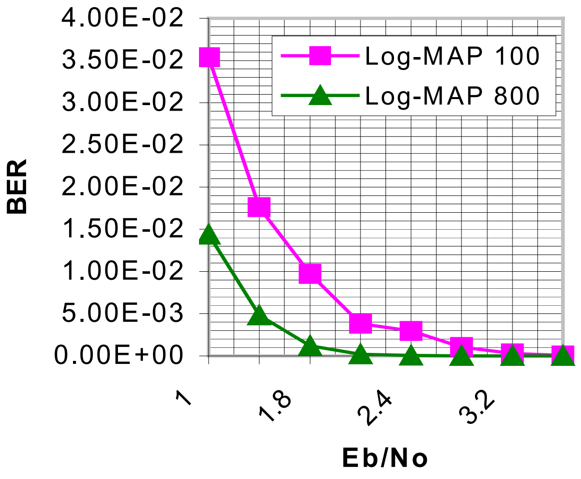

Figure 4. Influence of frame length on BER and Eb/No by Log-MAP algorithm

In Figure 4 we have compared the values of the 5th iteration for a different frame length (100 and 800) for the Log-MAP algorithm. It is obvious that by a lower signal/noise ratio the Log-MAP algorithm gives better results for a longer frame length (for the same number of iterations). However, by increasing the signal/noise ratio this difference becomes negligible.

4.2 Decoding TC with SOVA decoder

IV. Measurement: SOVA, parity bits punctured, frame length 100.

The next measurement is brought out for the SOVA decoder, parity bits were punctured and the frame length was 100 bits. Figure 5 shows the influence of the number of iterations between decoders and the signal/noise ratio on BER.

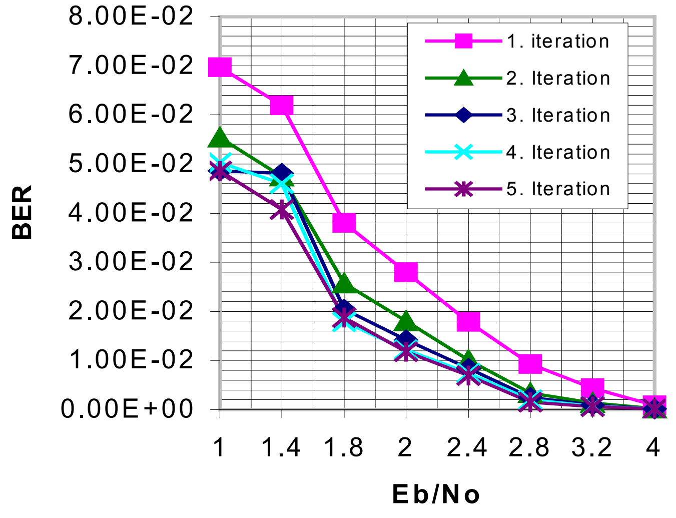

Figure 5. Dependence of BER and Eb/No for each of 5 iterations by SOVA algorithm, frame length 100

We can see that for lower values of Eb/No the impact on BER is much greater than for higher ones. It is also noticeable that the algorithm converges very fast, and further iterations are usually not necessary.

V. Measurement: SOVA, parity bits punctured, frame length 800.

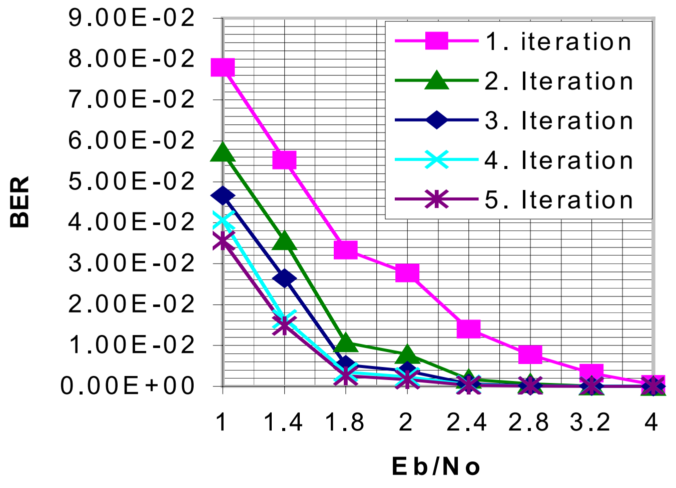

Figure 6. Dependence of BER and Eb/No for each of 5 iterations by SOVA algorithm, frame length 800

Figure 6 shows the diminution of BER by increasing Eb/No for the frame length 800. We can see that BER is minimal already by 2.4 dB for the 2nd and further iterations, while by frame length 100 it is minimal just beyond 3.2 dB.

VI. Measurement: SOVA, parity bits punctured, 5th iteration, frame length 100 and 800.

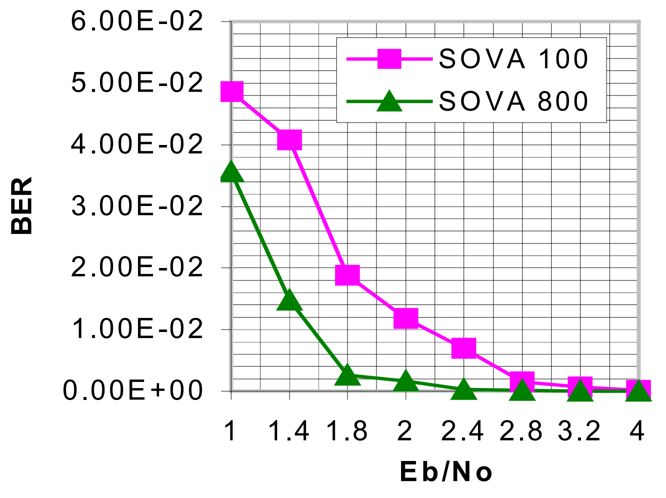

Figure 7. Influence of frame length on BER and Eb/No by SOVA algorithm

Figure 7 shows the influence of the frame length on BER. We can see that better results are achieved by a a longer frame length, similarly to by the Log-MAP algorithm. The curves are merged and the difference is negligible by higher Eb/No beyond 3.2 dB.

4.3 Comparison of Log-MAP and SOVA

Figure 8 shows the comparison of the performed algorithms for different frame lengths. We can see that the Log-MAP algorithm is dominant by both shorter and longer frames. The difference is more visible for smaller signal/noise ratios that are usual in practice.

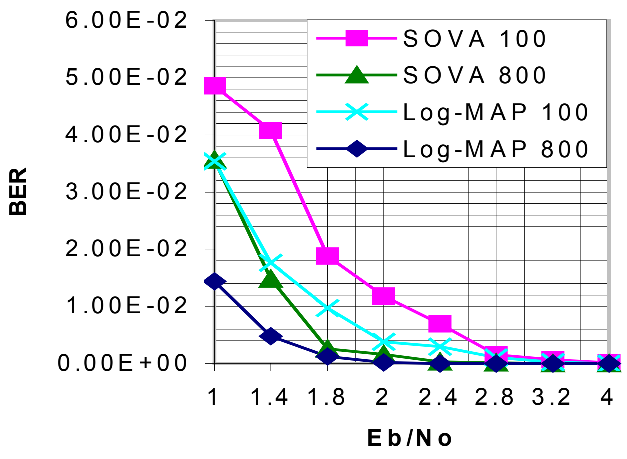

Figure 8. Comparison of Log-MAP and SOVA algorithm by frame length 100 and 800

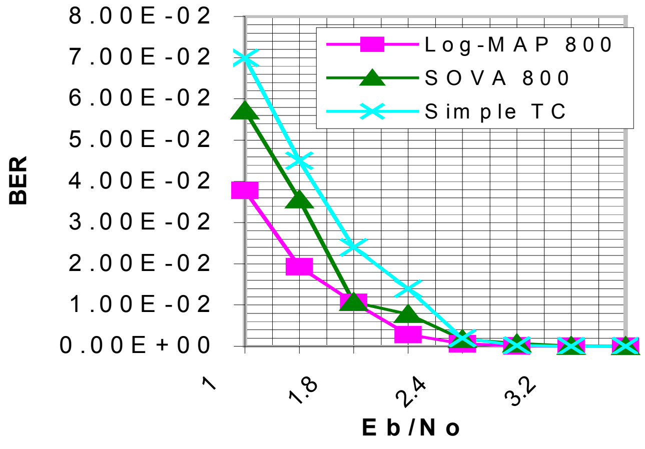

Figure 9. Comparison of Log-MAP and SOVA algorithm with simple theoretical TC

Figure 9 shows the results of the Log-MAP and SOVA decoder for the 2nd iteration, frame length 800, and the 2nd iteration of the simple Turbo decoder. We can see that the best results were obtained for the Log-MAP decoder and despite a relatively complex algorithm it is the best solution for most situations.

5 Conclusion

Due to huge volume of digital data presen practically everywhere it is necessary to develop appropriate algorithms for their transmission. However, data processing by a sender and a receiver is becoming efficient, and the transmission system must be able to transmit such amount of information by the allowed low BER. The turbo code fits very well into this scenario because of its simplicity and efficiency. Turbo codes have introduced great improvements in efficiency compared to other error correcting codes.

This paper presents the comparison of two methods for turbo code decoding: Log-Map and SOVA algorithm. The MAP (Maximum Aposteriori Probability) algorithm is used for prediction of the most possible information bit sent in code sequence. The Log-Map algorithm is MAP shifted into a logarithmic domain. The Soft Output Viterbi Algorithm, SOVA, uses a decoder that gives at its output a real number, which represents the bit error probability.

The results obtained from the implemented model of turbo code in Matlab program show that a better BER, for both decoders, could be obtained by sending the information in longer frames. If we compare the Log-MAP and SOVA algorithm we can conclude that Log-MAP dominated in all relevant parameters. It has lower BERs for all same signal/noise ratios (E_b/N_o).

References:

[1] D. Garrett, M. Stan, Low power architecture of the soft-output Viterbi algorithm, *1998 International Symposium on Low Power Electronics and Design*, 1998, pp. 262-267.

[2] S. S. Pietrobon, Implementation and performance of a Turbo/MAP decoder, *International Journal of Satellite Communications,* vol. 16. pp. 23-46, 1998.

[3] Berrou, C., Glavieux, A. and Thitimajshima, P., Near Shannon limit error-correcting coding and decoding: turbo codes, *ICC 1993,* Geneva, Switzerland, pp. 1064-1070, May 1993.

[4] S. Benedetto, G. Montorsi, Iterative decoding of serially concatenated convolutional codes, *IEE Electron. Lett.,* vol. 32, pp. 1186-1188, June 1996.

[5] J. Hagenauer and P. Hoeher, A Viterbi Algorithm with Soft-Decision Outputs and its Applications, In *Proc. Globecom `89,* Dallas, USA, pp. 1680-1686, 1989.

[6] B.J. Frey and F.R.Kschischang , Probability Propagation and Iterative Decoding, *IEEE Transactions on Communication, Control and Computing 1996,* Champaign-Urbana, Illinois.

[7] L.Bahl, J.Jelinek, J.Raviv, and F.Raviv, Optimal Decoding of Linear Codes for minimising symbol error rate, *IEEE Transactions on Information Theory,* vol. IT-20, pp.284-287, March 1974.