Iterative Decoding Algorithms for Turbo Product Codes

RODICA STOIAN, LUCIAN ANDREI PERIŞOARĂ
Faculty of Electronics, Telecommunications and Information Technology
"Politehnica" University of Bucharest
Bdul Iuliu Maniu, 1-3, Sector 6
ROMANIA

Abstract: - In this paper we introduce the iterative decoding principle, "the turbo principle", for the bidimensional Turbo Product Codes (TPC's). The constituent codes used for encoding on rows and columns are two concatenated (7,4) Hamming block codes.

Several Soft Input Soft Output (SISO) algorithms can be used for the iterative decoding process. At each iteration, the two decoders decode all rows, then all columns. For particular SISO algorithms, Maximum A Posteriori (MAP) algorithm and Soft Output Viterbi Algorithm (SOVA), the system is simulated and performances, in terms of Bit Error Rate (BER), are evaluated for an AWGN channel with BPSK modulation.

Key-Words: Turbo principle, iterative decoding, extrinsic information, product codes.

1 Introduction

Turbo codes were introduced as binary Error Correcting Codes (ECC's) built up from two Recursive Systematic Convolutional (RSC) codes concatenated in parallel. The turbo decoding algorithm, which processes the data in an iterative way, can achieve very high coding gain, reaching almost the Shannon limit [1]. For the decoding of the component codes are used the Soft Input-Soft Output (SISO) algorithms like Maximum A Posteriori (MAP) algorithm [1][2] or Soft Output Viterbi Algorithm (SOVA) [2].

Turbo Product Codes (TPC's), also known as Block Turbo Codes (BTC's), are based on linear block codes not on convolutional codes. Here, "turbo" refers to the iterative decoding approach and "product" refers to the fact that the TPC parameters are the product of those of its component codes.

Usually, TPC's are built on two or three-dimensional arrays of block codes. While the encoding process is done in a single iteration, the decoding process works with a fixed number of iterations or with a variable number of iterations and with a stop criterion.

The turbo principle, more exactly the turbo iterative decoding algorithm has been successfully applied in several decoding and detection problems as block turbo coding [1][2][3], coded modulation [4], multi-user detection [5], etc.

This paper presents the application of turbo principle to block array codes, using the BPSK modulation, the transmission over an AWGN channel and two SISO algorithms for the decoder.

2 The System Model

2.1 The Product Code Construction

Elias first introduced product codes (or iterated codes) in 1954 [7]. The concept of product codes is very simple and relatively efficient for building very long block codes by using two or more short block codes.

A product code $C = C_1 \otimes C_2$ is defined by the serial concatenation of two block codes $C_1(n_1, k_1, d_1)$ and $C_2(n_2, k_2, d_2)$, where n_i, k_i and d_i (i = 1, 2) denote the codeword length, the number of information bits and the minimum Hamming distance of the code C_i.

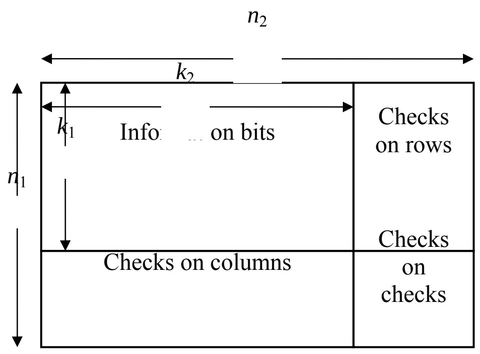

Fig. 1. The construction of the product code.

The construction of the product code is shown in Fig. 1 and can be described by the following steps:

a) the information bits are arranged, line by line, in an array of k_1 rows and k_2 columns;

b) the all k_1 rows are encoded horizontally using the code C_2;

c) the all n_2 columns are then encoded vertically using the code C_1;

d) the bidimensional codewords are transmitted row by row over the transmission channel.

The parameters of the product code $C(n,k,d)$ are :

- the matrix codeword dimension $n = n_1 \cdot n_2$;
- the number of information bits $k = k_1 \cdot k_2$;
- the minimum Hamming distance $d = d_1 \cdot d_2$;
- the code rate R is given by $R = k_1 k_2 / n_1 n_2$.

If the two codes can correct $t_1 = \lfloor (d_1 - 1)/2 \rfloor$, respectively $t_2 = \lfloor (d_2 - 1)/2 \rfloor$ errors, then the product code C is capable of correcting any combination of $t = \lfloor (d_1 d_2 - 1)/2 \rfloor = 2t_1 t_2 + t_1 + t_2$ errors. Thus, we can build very long block codes with large minimum Hamming distance by combining short codes with small minimum Hamming distance.

The parity check matrix $\mathbf{H}$ of a product code C is computed using the parity check matrices $\mathbf{H}_1$ and $\mathbf{H}_2$ of individual systematic codes C_1 and C_2 as:

$$\mathbf{H}^{\mathrm{T}} = \left[\mathbf{I}_{n_2} \otimes \mathbf{H}_1^{\mathrm{T}} \mid \mathbf{H}_2^{\mathrm{T}} \otimes \mathbf{I}_{n_1} \right] \quad (1)$$

where $\mathbf{I}_{n_1}$ and $\mathbf{I}_{n_2}$ are the unit matrices of order n_1 and n_2, respectively.

Given the construction procedure, it is clear that $(n_2\text{-}k_2)$ last columns of the matrix are the control bits of C_2. Also, all $(n_1\text{-}k_1)$ last rows of matrix C are the control bits of C_1. Hence, all the rows of matrix C are the codewords of C_2 and all the columns of matrix C are codewords of C_1.

2.2 The Iterative Decoder

The Bahl, Cocke, Jelinek and Raviv (BCJR) decoding algorithm, used in turbo decoding schemes, is a Soft-Input/Soft-Output algorithm while the Viterbi is a Soft-Input/Hard-Output (SIHO) algorithm [6].

McEliece presented a generalized description for the Viterbi Algorithm (VA), which acts as a unifying concept tying together the Viterbi and BCJR algorithms [7]. According McEliece, the Viterbi and BCJR are the same algorithm, differing only in the definition of the semi-ring operation and both algorithms can be used to produce an SISO decoder.

To further describe turbo decoding in the context of TPC, it is helpful to consider trellis description of linear block codes (see ANNEX 1). In this description, the Viterbi algorithm use a metric with (min) and (+) operations and BCJR algorithm use a metric with (min_log) and (*) operations.

The iterative turbo decoding can be view as a general Viterbi algorithm used in conjunction with MAP or SOVA, with appropriate metric for TPC case and specific applications [2][3].

As indicated by Elias [8], the TPC codes can be decoded by sequentially decoding the rows and columns of C in order to reduce decoding complexity. However, to achieve optimum performance, one must use soft decoding of the component codes using SISO decoders. More over, we can iterate the sequential decoding of C and thus reduce the BER after each iteration as for turbo codes [1].

The iterative decoding process is described in Fig. 2. The decoding is performed iteratively column-wise then row-wise using SISO decoders. The column decoder uses the channel observations $\mathbf{Y}$ and the *a priori* information $\mathbf{L}_c^-$ in the form of log-likelihood ratios to generate the *a posteriori* log-likelihood ratios $\mathbf{L}_c^+$ for all bits received. The *extrinsic* information is defined as $\mathbf{L}_{ec} = \mathbf{L}_c^+ - \mathbf{L}_c^- - L_c\mathbf{Y}$. For the second decoder, $\mathbf{L}_r^- = \mathbf{L}_{ec}$ is used as *a priori* information in conjunction with $\mathbf{Y}$. The decoder generates the *a posteriori* log-likelihood ratios $\mathbf{L}_r^+$ for all bits. The *extrinsic* information is then defined as $\mathbf{L}_{er} = \mathbf{L}_r^+ - \mathbf{L}_r^- - L_c\mathbf{Y}$ and is used as *a priori* information for the columns decoder.

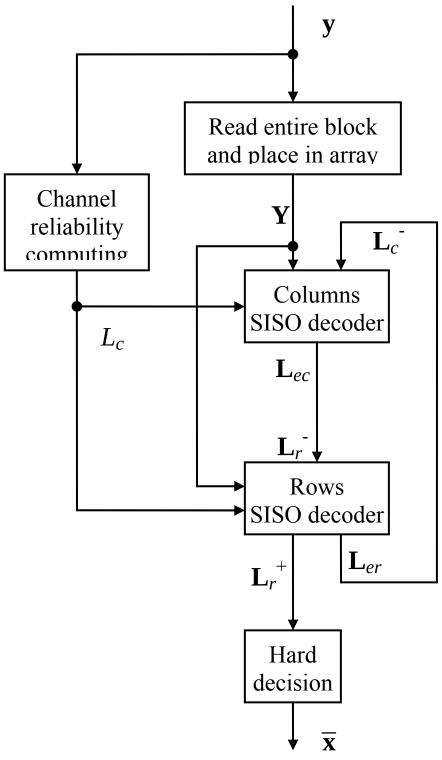

Fig. 2. The iterative decoding principle.

After a fixed number of iterations, the hard decision is done for each block of received symbols **y.** $\mathbf{L}_r^+$ is computed at the output of the second decoder and the original information message **u** is estimated based on the sign of the a posteriori values $\mathbf{L}_r^+$, as:

$$\hat{\mathbf{x}} = \operatorname{sign}\{\mathbf{L}_r^+\} \tag{2}$$

It can be easily seen, from Fig. 2, that the iterative principle is applicable to one complete decoding of columns and one complete decoding of rows. Note that all the decoding operations are made on all the bits within that block.

For a low complex implementation, we can use the same SISO decoder for rows and columns decoding if we add a block interleaver at the input of the rows decoder and a deinterleaver at the output. Also, when the constituent codes C_1 and C_2 are identical, the two decoders can be identically.

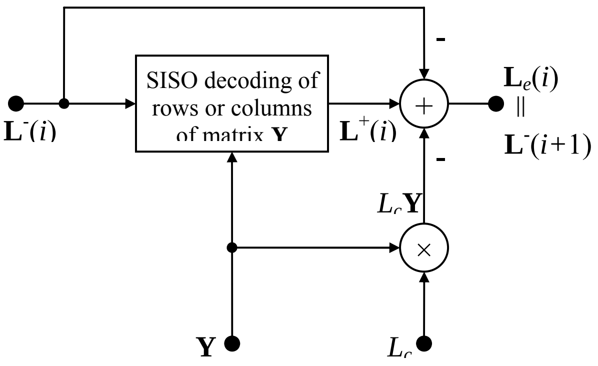

Fig. 3. The elementary block turbo decoder.

In this case, the decoding procedure described above is generalized by cascading elementary decoders illustrated in Fig. 3. The parameter i indicates the current decoding step of the iterative process. For the implementation of SISO decoders, we can use the MAP algorithm or the SOVA algorithm, which are described below.

2.3 The Maximum APosteriori Algorithm

Bahl, Cocke, Jelinek and Raviv proposed the Maximum A Posteriori (MAP) decoding algorithm for convolutional codes in 1974 [6]. The iterative decoder developed by Berrou et al. [1] in 1993 has a greatly increased attention. In their paper [1], the MAP algorithm was modified to minimize the sequence error probability instead of bit error probability for the original MAP algorithm. Because of its increased complexity, the MAP algorithm was simplified and the optimal MAP algorithm called the Log-MAP algorithm was developed.

The decoder operates based on the Logarithm Likelihood Ratio (LLR) for the transmitted bits **x** which is defined as:

$$L(\mathbf{x}) \overset{def}{=} \log\left(\frac{P(\mathbf{x}=+1)}{P(\mathbf{x}=-1)}\right) = \mathbf{L}^- \tag{3}$$

where the sign of the LLR $L(\mathbf{x})$ indicate whether the each bit of **x** is more likely to be +1 or -1 and the magnitude of the LLR gives an indication of the correct values of **x**.

In channel coding theory we are interested in the probability that $\mathbf{x} = \pm 1$, based or conditioned on received sequence **y**. So, we use the conditional LLR:

$$L(\mathbf{x}\,|\,\mathbf{y}) \overset{def}{=} \log\left(\frac{P(\mathbf{x}=+1\,|\,\mathbf{y})}{P(\mathbf{x}=-1\,|\,\mathbf{y})}\right) \overset{We\ noted}{=} \mathbf{L}^+ \tag{4}$$

The conditional probabilities $P(\mathbf{x}=\pm 1\,|\,\mathbf{y})$ are the a posteriori probabilities of the decoded bits **x** and $\mathbf{L}^+$ is the a posteriori information about **x**.

Also, it is used the conditional LLR $L(\mathbf{y}\,|\,\mathbf{x})$ based on the probability that the receiver's output would be **y** when the transmitted bits **x** were either +1 or -1:

$$L(\mathbf{y}\,|\,\mathbf{x}) \overset{def}{=} \log\left(\frac{P(\mathbf{y}\,|\,\mathbf{x}=+1)}{P(\mathbf{y}\,|\,\mathbf{x}=-1)}\right) \tag{5}$$

According [9], for AWGN fading channel using BPSK modulation we can write:

$$P(\mathbf{y}\,|\,\mathbf{x}=\pm 1) = \frac{1}{\sqrt{\pi N_0}} \exp\left[-\frac{E_b}{N_0}(\mathbf{y} \mp a)^2\right], \tag{6}$$

where E_b is the transmitted energy per bit, a is the fading amplitude and $N_0/2$ is the noise variance.

We can rewrite the equation (5) as following:

$$L(\mathbf{y}\,|\,\mathbf{x}) = -\frac{E_b}{N_0}\left[(\mathbf{y}-a)^2 - (\mathbf{y}+a)^2\right] = 4a\frac{E_b}{N_0}\mathbf{y} \overset{Noted}{=} L_c\mathbf{y} \tag{7}$$

where $L_c = 4aE_b/N_0$ is defined as the *channel reliability* value. For non fading AWGN channels a=1 and $L_c = 4E_b/N_0$.

In [1], [9] the extrinsic information is defined as:

$$\begin{aligned} \mathbf{L}^e &= \log\left(\frac{P(\mathbf{x}=+1\,|\,\mathbf{y})}{P(\mathbf{x}=-1\,|\,\mathbf{y})}\right) - \log\left(\frac{P(\mathbf{x}=+1)}{P(\mathbf{x}=-1)}\right) \\ &\quad - \log\left(\frac{P(\mathbf{y}\,|\,\mathbf{x}=+1)}{P(\mathbf{y}\,|\,\mathbf{x}=-1)}\right) = \mathbf{L}^+ - \mathbf{L}^- - L_c\mathbf{y} \end{aligned} \tag{8}$$

In the iterative decoding procedure the extrinsic information $\mathbf{L}^e$ becomes the a priori information $\mathbf{L}^-$ for the next decoder. If $\mathbf{L}^-$ is a large (or small) positive number, then it would be difficult (or easier) to change the estimated symbol decision from +1 to -1 between to consecutive decoding stages [10].

The term $L_c\mathbf{y}$ is the soft output of the channel for the information symbol **x**. For high SNR, the channel reliability value L_c will be high and this information symbol will have a large influence on $\mathbf{L}^+$. Conversely, for low SNR, the L_c is low and it's influence on $\mathbf{L}^+$ is insignificant.

2.4 The Soft Output Viterbi Algorithm

In practical systems, we quantize the received channel symbols with one (hard decision) or a few bits of precision (soft decision) in order to improve the performances of the Viterbi decoder. For *m*-bit quantization, one quantization bit is devoted to the sign of the decision and *m*-1 bits are devoted to the signal's magnitude. The larger the magnitude, the more confidence that the sign bit is correct. Decoders that exploit soft decisions can reduce S/N ratio requirements by approximately 2 dB over those that use hard decisions alone [11].

The Viterbi algorithm finds the trellis path or state sequence **s** so that the a posteriori probability $p(\mathbf{s}|\mathbf{y})$ is maximized. Accordingly to the Bayes rule, we can equivalently maximize:

$$p(\mathbf{s}_j,\mathbf{y}_j) = p(\mathbf{s}_{j-1},\mathbf{y}_{j-1})p(u_j)p(y_j \mid s',s)\,, \qquad (9)$$

where $\mathbf{s}_j = (s_1, s_2, ..., s_j)$, $\mathbf{y}_j = (y_1, y_2, ..., y_j)$, $s' = s_{j-1}$ and $s = s_j$. The u_j is the source symbol for the state transition $s' \rightarrow s$ of trellis path s_j. The path metric $M_j(\mathbf{s}_j)$ associated with the trellis path $\mathbf{s}_j$ is defined as:

$$M_j(\mathbf{s}_j) = \log\left(p(\mathbf{s}_j,\mathbf{y}_j)\right). \qquad (10)$$

Obviously,

$$p(\mathbf{s}_j,\mathbf{y}_j) = \exp\left(M_j(\mathbf{s}_j)\right). \qquad (11)$$

Substituting (9) into (10) gives:

$$M_j(\mathbf{s}_j) = M_{j-1}(\mathbf{s}_{j-1}) + \log\left(p(u_j)\right) + \log\left(p(y_j \mid s',s)\right) \quad (12)$$

where $\log\left(p(u_j)\right)$ is the a priori information of the source symbol u_j and $\log\left(p(y_j \mid s',s)\right)$ is the branch metric for the state transition s'→s given the received signal y_j. At time j, for each state s, the path metrics for all possible paths terminating at state *s* are calculated.

3 Performance Evaluation

To simulate the application of iterative decoding principle, we applies the described algorithms to TPC ensemble which use two identical systematic Hamming block codes $H_1(7,4,3)$, $H_2(7,4,3)$ concatenated in a serial way.

The product code is $H_1(7,4,3) \otimes H_2(7,4,3) = H(49,16,9)$ and the output sequence of TPC is BPSK modulated and transmitted over an AWGN channel, with fading amplitude $a = 1$ [11].

The most important characteristic of iterative principle is the dependence of BER(E_b/N_0) of the number of decoding iterations. Bit Error Rate is computed over 10^5 blocks, each block of dimension 49 bits.

- For Maximum A Posteriori algorithm we obtain the curves plotted in Fig. 4. For each additional iteration, we obtain a reduction of BER. We observe that for an E_b/N_0 of 3dB the BER is equal to 0.0633 for one iteration, 0.0105 at iteration 2, 0.00054 at iteration 3 and 0.000043 at iteration 5.

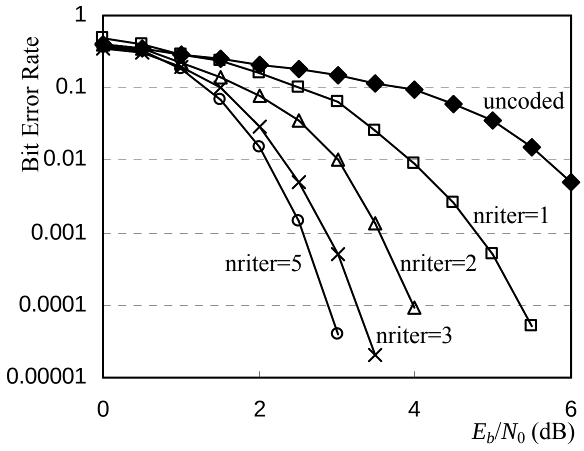

Fig. 4. BER(E_b/N_0) performance for MAP algorithm.

- The performances of the SOVA algorithm are illustrated in Fig. 5. In this case, for the same E_b/N_0 of 3dB the BER is greater, 0.0867 for one iteration, 0.0273 at iteration 2, 0.0016 at iteration 3 and 0.0007 at iteration 5. From Fig. 4 and Fig. 5 we observe that the MAP algorithm gives better results, in terms of BER, compared with SOVA algorithm.

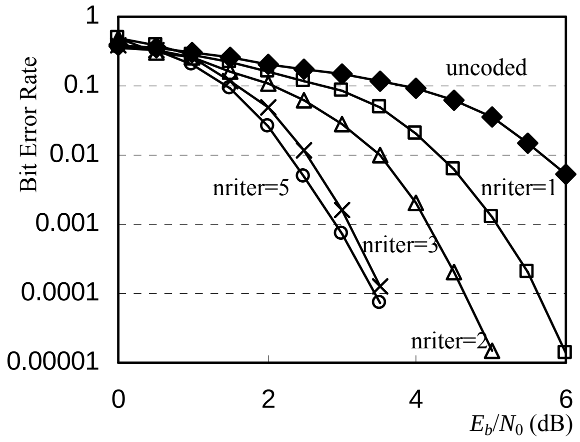

Fig. 5. BER(E_b/N_0) performance for SOVA algorithm.

4 Conclusions

In this paper, two iterative SISO decoding algorithms for TPC have been presented. It has been proved that the two-bit soft decision decoding for

TPC (SISO algorithms) can pick up 2 dB of additional coding gain compared with SIHO variant [11][12].

For these results, the complexity is low and TPC systems starts to be available as standard products. Of major interest are the combination of the TPC coding with modulation and the development of specific SISO algorithms, combined with helical data scrambling to improve burst error performance.

ANNEX 1. TRELLIS DESCRIPTION OF BLOCK CODES:

For a Hamming code with control matrix $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_n]$, where $\mathbf{h}_i$ is the i^{th} column of $\mathbf{H}$, any codeword $\mathbf{c}_i = (c_{i1}, c_{i2}, ..., c_{in})$, $\mathbf{c}_i \in \mathscr{C}(n,k)$, $i = \overline{1,n}$, must satisfy the condition:

$$c_{i1}\mathbf{h}_1 + c_{i2}\mathbf{h}_2 + ... + c_{in}\mathbf{h}_n = \mathbf{0}, \qquad (13)$$

where $c_{ij} \in F_2$ and $\mathbf{h}_j \in F_2^{n-k}$.

For any codeword affected by errors the value of the syndrome is:

$$\mathbf{s}_n = \sum_{i=1}^{n} y_i \mathbf{h}_i \qquad (14)$$

where y_i are the components of received vector $\mathbf{y}$.

The BCJR trellis construction for linear block codes is based on recursive computation of the syndrome [6]:

$$\mathbf{s}_i = \mathbf{s}_{i-1} + y_i \mathbf{h}_i, \ \ \mathbf{s}_0 = 0, \qquad (15)$$

which determine the *unconstrained trellis*. Because for linear block codes the initial and final states must be 0, the branches of the unconstrained trellis, which not start from the state 0 and not end at the state 0, are removed. Fig. 6 shows the BCJR trellis for the systematic Hamming block code H(7,4,3).

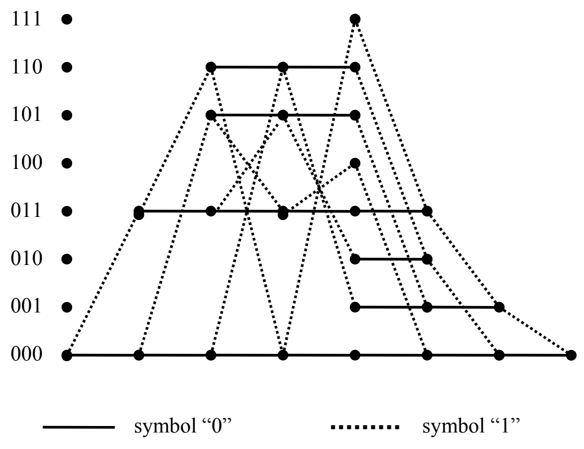

Fig. 6. The BCJR trellis for the systematic H(7,4,3).

REFERENCES:

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

[2] Rodica Stoian, L.A. Perişoară, M. Crângeanu, "A comparison of MAP and SOVA decoding algorithms for Turbo Codes", *in Proc. of the 35th Symp. of ACTTM*, Bucharest, Romania, Mar. 2004, (CD).

[3] J. Hagenauer, E. Offer, L. Papke, "Iterative decoding of binary block and convolutional codes", *IEEE Trans. on Inf. Theory*, vol. 42, March 1996, pp. 429-445.

[4] S. Benedetto, D. Divsalar, G. Montorsi, F. Pollara, "Parallel concatenated trellis coded modulation", *in Proc. IEEE Int. Conf. on Comm.*, vol. 2, Dallas, Jun 1996, pp. 974-978.

[5] A. Stefanov, T. M. Duman, "Turbo-coded modulation for wireless communications with antenna diversity", *in Proc. IEEE Vehicular Technology Conf.*, Amsterdam, Netherlands, Sept. 1999, pp. 1565-1569.

[6] L. R. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate", *IEEE Trans. on Inf. Theory*, vol. 20, 1974, pp. 284-287.

[7] R. J. McEliece, "On the BCJR trellis for linear block codes", *IEEE Trans. on Inf. Theory*, vol. IT-42(4), 1996, pp. 1072-1092.

[8] P. Elias, "Error-free coding", *IRE Trans. on Inf. Theory*, vol. IT-4, Sept. 1954, pp. 29-37.

[9] C. Berrou, A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo Codes", *IEEE Trans. on Comm.*, vol. 44, no. 10, Oct. 1996, pp.1261-1271.

[10] R. Stoian, L.A. Perişoară, "The reliability of turbo decoders over Gaussian channels", *in Proc. of "Communications 2004"*, Technical Military Academy, Bucharest, Romania, June 2004, pp. 545-550.

[11] J. Hagenauer and P. Hoeher, "A Viterbi Algorithm with Soft Decision Outputs and Its Applications", *in Proc. of GLOBECOM 1989*, Dallas, Texas, Nov. 1989, pp. 1680-1686.

[12] R. M. Pyndiah, "Near-optimum decoding of Product Codes: Block Turbo Codes", *IEEE Trans. on Comm.*, vol. 46, no. 8, Aug. 1998, pp. 1003-1010.