New High Performance Deterministic Interleavers for Turbo Codes

Georgian Alexandru LAZAR, Nicoleta Roxana BUZATU, Elena COJOCARIU,
Lucian TRIFINA, Razvan VIERU
Gheorge Asachi University of Iasi
bd.Carol I nr.11, RO-700506 Iasi
lazara@etc.tuiasi.ro

***Abstract*—Turbo codes offer extraordinary performance, especially at low signal to noise ratios, due to a low multiplicity of low weight code words. The interleaver design is critical in order to realize an apparent randomness of the code, thus further enhancing its performance, especially for short block frames. This paper presents four new deterministic interleaver design methods, that lead to highly performing turbo coding systems, namely the block-spread, the block-backtracking and their variations the linearly-spread and linearly-backtracking interleavers. The design methods are explained in depth and the results are compared against some of the most wide-spread turbo code interleavers. Furthermore, the selection method of the generator polynomials used in the simulations is explained.**

***Index Terms*—Channel coding, Concatenated coding, Deterministic algorithms, Error correction coding, Interleaved coding**

I. INTRODUCTION

Turbo codes represent a powerful, yet flexible class of error correcting codes. It has been proven that these codes offer remarkable performance especially over low SNR domains. The low error rate of the turbo coding scheme is achieved by combining two digital IIR (Infinite Impulse Response) filters (convolutional encoders). A non-uniform interleaver scrambles the ordering of the input bits of the second digital filter as shown in Figure 1.

The interleaver creates apparent randomness to the code (it is very unlikely that both component encoders would produce low weight code words, thus the performance is increased), but the turbo encoder itself still retains adequate structure so that the decoding is feasible. The interleaver also provides flexibility in terms of performance and latency. In case that performance is the main concern, then a large size interleaver should be used, whereas if low latency is a mandatory requirement of the application, then an reduced size interleaver can be involved in the communication process.

The main problem of turbo codes is given by their high error floor, which appears at medium to high SNR domains. This drawback is the effect of the low free distance. Its effects can be reduced through the design algorithm of the interleaver.[1]

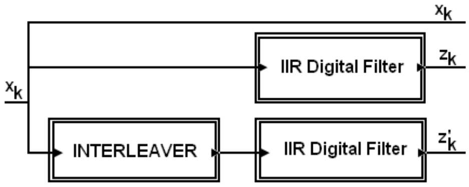

Figure 1. Generic structure of an 1/3 Turbo Encoder.

Digital Object Identifier 10.4316/AECE.2010.02022

From the point of view of how the interleaver is generated, there are two major interleaver design methods: deterministic and random. The deterministic design is based on a certain algorithm that can be reproduced locally, at both the emitter and the receiver sides, thus there is no need to store the value of the interleaver. On the other hand, the random generation algorithm cannot produce the same permutation for both the emitter and receiver. In this case, the interleaver has to be stored locally, which can be a major inconvenient for standards such as UMTS or DVB which require variable frame lengths. That would mean that large memory blocks should be used in order to store not just a single random interleaver, but a whole number of different random interleavers for every frame length used.

The basic role of an interleaver is to construct a long block code from small memory convolutional codes, thus approaching the Shannon capacity limit. Secondly, it spreads out burst errors by decorrelating the inputs of the two component decoders and enabling the use of an iterative suboptimum decoding method. The final role of the interleaver is to break low weight input sequences and hence increase the code free Hamming distance, or reduce the number of codewords with small distances in the code distance spectrum.

II. THE SYSTEM MODEL USED IN THE SIMULATIONS

The turbo encoder used in the simulation is symmetrical and uses two identical convolutional encoders with the feed-forward and feedback polynomials equal in octal to 15 and 13, respectively. In Figure 2 the structure of the encoder is depicted. X_k is the systematic output from the first encoder, Z_k and Z'_k are the parity outputs of the first and second convolutional encoders. The systematic output from the second encoder is punctured. The argumentation behind the selection of this generator polynomial is explained in section VI.

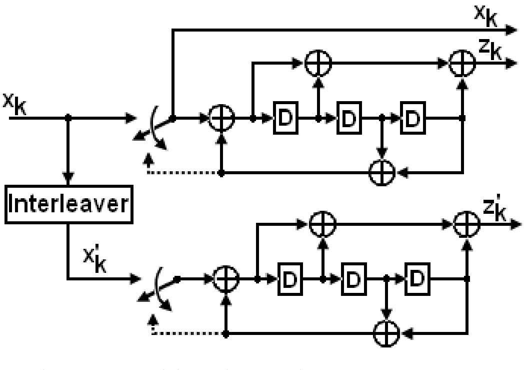

Figure 2. The structure of the turbo encoder.

The turbo encoder used has a post-interleaver trellis termination (flushing), which means that both convolutional encoders are independent from one another reset. This is done by commuting the two switches from the on state (after a number of clock cycles equal to the size of the interleaver) to the off state (for a number of three clock cycles, which is equal to the memory of the constituent convolutional encoders).[2]

In order to increase the coding gain for the fading channel, BICM (Bit Interleaved Coded Modulation) is used.[3] In this situation an random interleaver is deployed between the encoder and the BPSK modulator (Figure 3).

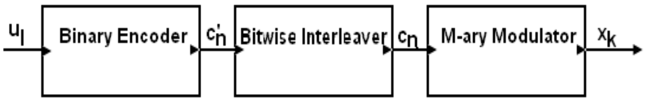

Figure 3. The Bit Interleved Coded Modulation Principle.

Two sets of simulations are run, supposing that the channel is either an Additive White Gaussian Noise (AWGN) channel, or either a Rayleigh Multiplicative Fading (RMF) channel (Figure 4). The matched filter is required in order to translate the receive symbols into a Log Likelihood Ratio (LLR) form.

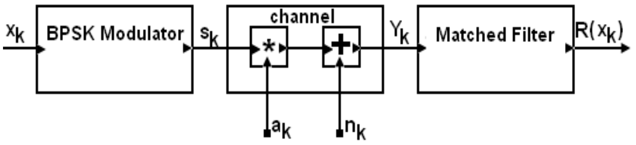

Figure 4. The channel model.

The variable a_k (channel gain) =1 if AWGN, or a Rayleigh random variable if RMF and n_k is the Gaussian noise.

The turbo decoder is based on a SW-SISO (sliding window-soft input soft output) iterative algorithm with two MAP (Maximum A Posteriori) decoders implemented in the log-domain. The LLR of the data bits is calculated by each MAP decoder and the results are passed from one decoder to the other (iterations are performed). The performance of this kind of approach depends on the number of iterations that take place between the two MAP decoders. In the simulation scenarios, the log-map approximation is used, by the means of a look-up table and finally, a hard decision is performed on the value of the LLR. [4]

III. CLASSIC INTERLEAVER TYPES

The block interleaver formats the data frame of length K into a matrix with N rows and M columns, with K=N*M. The data is written row-wise and the reading is performed column-wise. The structure of this interleaver is given in equation (1):

$$\pi(i+j\cdot M+1)=i\cdot N+j+1 \tag{1}$$

where

$$(\forall)i\in\{0,1,\ldots,M-1\} \tag{2}$$

and

$$(\forall)j\in\{0,1,\ldots,N-1\} \tag{3}$$

The random interleaver is constructed by generating a random dither vector of length K. The permutation is given by sorting the dither vector.

The S-Random interleaver is randomly generated and its elements respect a user imposed S spreading factor. The algorithm is initialized by generating an empty vector of size K. The element 'i' of the vector is randomly chosen in order to differ from the last S elements with a value of at least S, thus satisfying the previously mentioned condition. The generation time is reasonable, provided that the spreading value S is less than $S\le\sqrt{K/2}$.[5] The spreading factor S is defined in equation (4):

$$|i-j|\le S\Rightarrow|\pi(i)-\pi(j)|>S;(i;j)\in[0;K-1] \tag{4}$$

IV. THE BLOCK- SPREAD AND LINEARLY -SPREAD INTERLEAVERS

The spreading algorithm is basically a permutation, which ensures that the difference between consecutive numbers forms an arithmetic progression. Thus, the data is scrambled in an irregular fashion. The maximum difference between two consecutive numbers that belong to a vector which has the size K, can be determined by computing the sum of an arithmetic progression and comparing the result to K. If the sum is smaller than K, then the progression is appended with the next term. The largest term of the progression that respects the above mentioned condition represents the maximum difference that can be obtained from the permutation.

The spreading algorithm that generates a permutation perm of size dim, follows the next steps:

```
1). Initialize the temp vector of size dim with all the elements of the permutation and initialize the size l of the permutation vector perm with zero
2).Compute the maximum possible difference between two consecutive numbers as referred to the size of the permutation
3). While l<dim do
4). For j=1 to the maximum possible difference do
    l:=l+1 and perm(l):=perm(l-1)+j
5).Once the maximum possible difference is achieved rebuild the temp vector with all the unused remaining elements, sorted in an ascending order.
6).The new maximum possible difference is computed as referred to the size k of the temp vector.
7).The algorithm is finished when l reaches the desired length dim.
```

In case the descending spreading algorithm is desired, than the loop from step 4 is done downwards and the remaining unused elements from step 5 are sorted in descending order.

The design algorithm of the block-spread (BS) interleaver aims to increase the dispersion Γ of the spreading algorithm, by defining two indexes which create an apparent randomness in the way the matrix data is read. The indexes are not incrementally built, but rather following a specific pattern, which ensures that the deterministic feature is still retained, albeit the performance is increased.

The algorithm of the block-spread interleaver can be described in the following manner:

```
1).Pre-format the data in a matrix-like
```

```
structure with R rows and C columns, in which the data is written row-wise
  2).Perform the reading from the above matrix column-wise using two index vectors r_column and r_row
  3).The r_column vector is given by applying the spreading algorithm for a permutation with a number of elements equal to C
  4).The r_row vector is given by applying the spreading algorithm for a permutation with a number of elements equal to R
```

The linearly-spread interleaver is designed using the same algorithm, but the data is read row-wise rather than column-wise.

V. THE BLOCK-BACKTRACKING AND LINEARLY-BACKTRACKING INTERLEAVERS

The proposed interleaver design is based on the backtracking algorithm Backtracking is a general algorithm aimed at solving various computational problems by building tree-like structures, for each potential solution candidate. In case the current branch proves not to be a valid solution, then the whole branch is cut off, and a different branch is created starting from the last known potential valid root. The solution candidate vector is valid, if each of its members respects a certain pre-imposed condition.

The proposed backtracking algorithm imposes the condition that the interleaver made up by the solution vector has an user imposed spreading value S. This value can be any ranging from 1 to $\sqrt{K/2}+1$.

The steps of the backtracking algorithm can be synthesized as follows:

```
  1).Initialize the solution vector sol(1)=desired starting position
  2).While i>1 and i<= the desired length do
  3).For j=1 to the desired length do
Incrementally search for valid solution sol(i)=j that complies with the following requirements :
a).j differs with a value of at least S from the last S members of the current solution candidate vector
b).j must be a value that was not previously assigned in the current solutin candidate vector
  4).If the value j is a valid solution than sol(i):=j and i:=i+1
  5).If none of the j values from 1 to the desired length can be a valid solution for sol(i), then i:=i-1 and another solution is searched for j=sol(i+1)+1 to the desired length
  6).The algorithm is finished either when the current solution vector candidate has the desired length, or when the index i becomes equal to 1, case in which the desired starting position cannot produce an interleaver with the desired spread factor S.
```

The pure backtracking algorithm that is described above cannot produce high performance interleavers, because even if the parameter S can be set to a high value, the dispersion Γ is reduced. Furthermore, for greater interleaver lengths, the generation time increases significantly, fact that is intolerable in standards such as the UMTS. For an interleaver of length K, the dispersion Γ is defined in [6]. For an interleaver of length K, the normalized dispersion γ is the dispersion divided by a factor of 0.5*K*(K-1).

In order to increase the Γ parameter and decrease the generation time, the following block-backtracking (BB) interleaver generation algorithm has been implemented:

```
  1).Pre-format the data in a matrix-like structure with R rows and C columns, in which the data is written row-wise
  2).Perform the reading from the matrix above column-wise using two index vectors r_column and r_row
  3).The r_column vector is given by applying the backtracking algorithm starting from a desired position poz, with a spreading value equal to $\sqrt{C/2}+1$ and a total length equal to C
  4).For i=1 to C
  5).The r_row vector is given by applying the backtracking algorithm starting from a position poz equal to r_column(i), with a spreading value equal to $\sqrt{R/2}+1$ and a total length equal to R.
```

The linearly-backtracking interleaver is designed using the same algorithm, but the data is read row-wise rather than column-wise.

VI. THE CODE GENERATOR POLYNOMIAL SELECTION

The most important parameters as computed for the selected interleavers are depicted in Tables I-X. From this tables it can be concluded that the block- spread (BS) and block- backtracking (BB) interleavers have a D-spreading parameter and corner of merit (C_m), both defined in [7] larger than that of the S-Random interleaver. The corner of merit plays a critical role especially for turbo codes in which the post-interleaver termination strategy is applied. In this situation C_m has to be maximized, in order to avoid edge effects (low-weight codewords can be generated if a weight-one input sequence with the non-zero bit near the end of one constituent code, maps to a near-end position in the other constituent code, because both encoders are terminated but the tails are not interleaved). The generation time T_{gen} is expressed in ms.

As far as the S-Random interleaver is concerned, the two new interleaver designs prove to have a much faster generation time and above all, their design is purely deterministic, so there is no need to store their values in memory. This feature is essential in standards with variable frame lengths.

Not only the generation time of the two proposed interleavers is significantly faster than the generation time of the S-Random interleaver, but also the latency introduced in the system is lower, thus this feature enables these kinds of interleavers to be used in various communication standards that require low memory usage and latency. The latency of an inteleaver (L) is defined in [8], while the S_{new} spreading factor is defined in [9].

TABLE I. PARAMETERS OF THE RANDOM INTERLEAVER FOR L1=100

S	S_{new}	D	γ	C_m	L	T_{gen}(ms)
1	3	3	0.81	24	178	0.2

TABLE II. PARAMETERS OF THE BLOCK INTERLEAVER FOR L1=100

S	S_{new}	D	γ	C_m	L	T_{gen}(ms)
8	11	2	0.03	2	162	0.6

TABLE III. PARAMETERS OF THE S-RANDOM INTERLEAVER FOR L1=100

S	S_{new}	D	γ	C_m	L	T_{gen}(ms)
6	8	5	0.79	6	184	362.7

TABLE IV. PARAMETERS OF THE BB INTERLEAVER FOR L1=100

S	S_{new}	D	γ	C_m	L	T_{gen}(ms)
2	5	5	0.52	11	163	5.1

TABLE V. PARAMETERS OF THE BS INTERLEAVER FOR L1=100

S	S_{new}	D	γ	C_m	L	T_{gen}(ms)
5	9	9	0.51	9	171	3.2

TABLE VI. PARAMETERS OF THE RANDOM INTERLEAVER FOR L2=225

S	S_{new}	D	γ	C_m	L	T_{gen}(ms)
0	2	2	0.81	30	413	0.7

TABLE VII. PARAMETERS OF THE BLOCK INTERLEAVER FOR L2=225

S	S_{new}	D	γ	C_m	L	T_{gen}(ms)
13	16	2	0.01	0	392	0.7

TABLE VIII. PARAMETERS OF THE S-RANDOM INTERLEAVER FOR L2=225

S	S_{new}	D	γ	C_m	L	T_{gen}(ms)
10	12	8	0.78	7	424	1690.1

TABLE IX. PARAMETERS OF THE BB INTERLEAVER FOR L2=225

S	S_{new}	D	γ	C_m	L	T_{gen}(ms)
3	8	8	0.49	21	397	16.4

TABLE X. PARAMETERS OF THE BS INTERLEAVER FOR L2=225

S	S_{new}	D	γ	C_m	L	T_{gen}(ms)
9	13	13	0.51	14	406	6.2

All the parameters illustrated in the Tables I-X are not code dependent. It is expected that an interleaver with better parameters leads to lower bit and frame error rates, no matter the component convolutional code used. The performance of the turbo encoder is both code and interleaver dependent. Not only the design and length of the interleaver influence the performance, but also the memories and the structure of the component convolutional encoders play a major part.

The composite influence of both the constituent encoders and the interleaver can be weighted using a distance spectrum evaluation. This performance evaluation can be made for turbo codes of the same memory and interleaver length. In practice, memory three component convolutional codes are used. This is the case for standards such as UMTS, CDMA2000 and LTE.

The search for the best memory three generator polynomials is made using two block- spread interleavers of lengths L_1=100 and L_2=225. The chosen generator polynomial for the turbo encoder is that which leads to the best first spectral line (highest free distance d_{free} and lowest multiplicities n_{free} and total information weight w_{free})[11].

The best generator polynomials in octal form, with the most significant bit in the left side are depicted in Tables XI and XII, for the two considered block- spread interleaver lengths. From these tables it can be deduced that the best generator polynomial is 15/13.

TABLE XI. THE BEST M=3 GENERATOR POLYNOMIALS FOR BS OF L1=100

Rank	G_8	d_{free}	n_{free}	w_{free}
1	15/13	17	5	13
2	11/15	15	4	12
3	11/13	14	1	1

TABLE XII. THE BEST M=3 GENERATOR POLYNOMIALS FOR BS OF L2=225

Rank	G_8	d_{free}	n_{free}	w_{free}
1	15/13	21	4	10
2	11/15	20	1	1
3	11/13	18	1	1

VII. SIMULATIONS AND RESULTS

The simulations were run for two interleaver lengths (L1=100 and L2=225), both under AWGN and RMF situations. The generator polynomial for the component covolutional encoders was 15/13, the number of decoder iterations was 12, the modulation used was BPSK with BICM and the decoding algorithm was log-MAP. The BER or FER curves are shown in Figures 5-6 (for L1=100) and Figures 7-8 (for L2=225), respectively. Additionally, Tables XIII and XIV depict the first spectral line as computed for the selected interleavers. The two proposed interleaver designs surpass the interleavers that are compared against. This fact can be explained through their increased d_{free}, decreased N_{free} and w_{free}. Furthermore, the S_{new} and D spreading factors are greater for the two proposed interleavers. In the context of a post-interleaver termination, the corner of merit C_m contributes to the performance enhancement obtained by the BS and BB permutations.

TABLE XIII. THE FIRST SPECTRAL LINE FOR L1=100 AND G_8=15/13

Interleaver	d_{free}	n_{free}	w_{free}
Random	12	1	2
Block	17	81	241
S-Random	14	4	8
BB	16	2	4
BS	17	5	13

TABLE XIV. THE FIRST SPECTRAL LINE FOR L2=225 AND G_8=15/13

Interleaver	d_{free}	n_{free}	w_{free}
Random	13	1	3
Block	19	2	2
S-Random	16	5	6
BB	19	1	1
BS	21	4	10

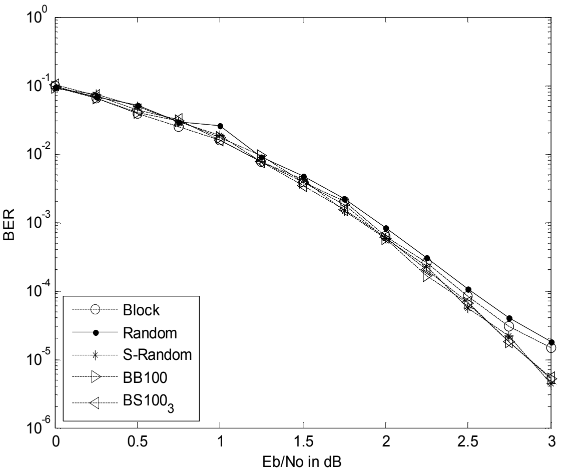

Figure 5. BER for L1=100 and AWGN.

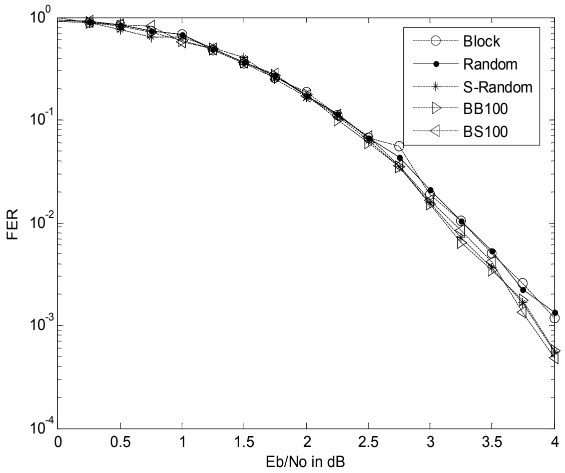

Figure 6. FER for L1=100 and Rayleigh Fading.

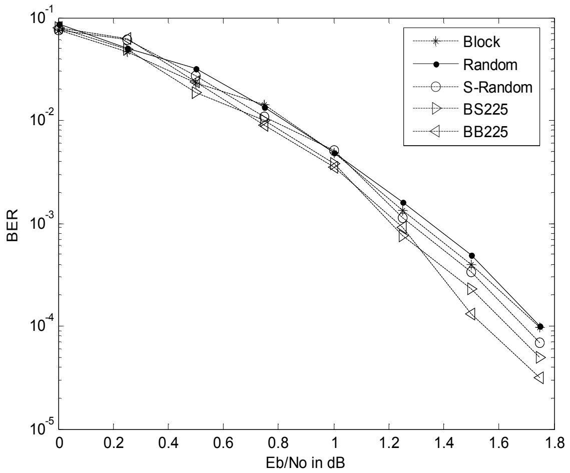

Figure 7. BER for L2=225 and AWGN.

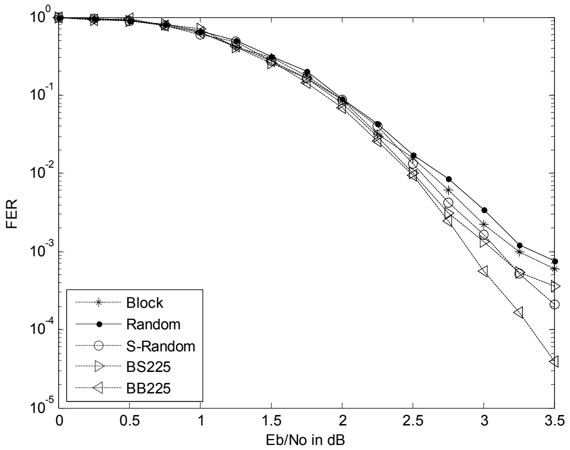

Figure 8. FER for L2=225 and Rayleigh Fading.

VIII. CONCLUSIONS AND FUTURE WORK

This paper presents two new interleaver designs, the block-backtracking (BB) and the block-spread (BS) techniques. Their performances are compared against the block, random and S-random interleavers, in case of both AWGN and RMF channels, for two different short length frame sizes. The selection of the generator polynomials for the turbo encoder used in the simulations is contended. Future work should address to the study of these interleaver behaviors for longer frame sizes and for generator polynomials of larger memories.

REFERENCES

[1] A.H.S Mohammadi, W. Zhuang- "Variance of the Turbo Code Performance Bound over the Interleavers" IEEE Transactions on Information Theory vol. 48, no. 7, july 2002, pp. 2078-2086

[2] M. C. Valenti and J. Sun-"The UMTS Turbo Code and an Efficient Decoder Implementation Suitable for Software-Defined Radios" International Journal of Wireless Information Networks vol.8, no 4, october 2001, pp. 203-215

[3] X.Zou, M.Wang, G.Feng –"A New Interleaver Design for Iteratively Decoded Bit-Interleaved Coded Modulation" International Journal of Soft Computing 3 (5), Medwell Journals 2008, pp. 338-343

[4] P. Robertson-"Illuminating the Structure of Code and Decoder of parallel concatenated recursive systematic (turbo) codes" IEEE International Conference on Communications (ICC) 1994, pp 1298-1303

[5] B.G. Lee, S. J. Bae, C. K. Jeong, E. K. Joo- "Performance Analysis of Swap Interleaver for Turbo Codes" Electronics Letters, vol 35, no 32, october 1999, pp. 1939-1940

[6] C. Avenacio-Leon – "Analysis of the Dispersion and Spreading Properties of Interleavers for Turbo Codes " Computing Research Conference CRC 2004

[7] O. Takeshita- "Permutation Polynomial Interleavers: An Algebraic-Geometric Perspective" IEEE Transactions on Information Theory, vol 53, no 6, june 2007

[8] K. Andrews, C. Heegard, D. Kozen – "A Theory of Interlavers" Cornell University Press 1997

[9] S. Crozier- "New High-Spread High-Distance Interleavers for Turbo-Codes", 20th Biennial Symposium on Communications, Kingston, pp 3-7 May 2000

[10] L. Perez, J. Seghers, D. Costello –"A Distance Spectrum Interpretation of Turbo Codes" IEEE Transactions of Information Theory, special issue on coding and complexity 1996, pp 1698-1709

[11] R. Garello, F. Chiaraluce, P. Pierleoni, M. Scaloni, S. Benedetto –"On the Error Floor and Free Distance of Turbo Codes" IEEE International Conference on Communications 2001, pp.45-69