Performance and complexity of block turbo decoder circuits

Patrick Adde, Ramesh Pyndiah, Olivier Raoul

Performance and complexity of block turbo decoder circuits

Patrick ADDE, Ramesh PYNDIAH and Olivier RAOUL.
Télécom Bretagne, Technopôle Brest Iroise, BP 832, 29285 BREST Cedex FRANCE.
Tel : (33) 98 00 13 10, Fax : (33) 98 00 13 43
Mail: patrick.adde@enst-bretagne.fr/ ramesh.pyndiah@enst-bretagne.fr/ olivier.raoul@enstbretagne.fr

ABSTRACT

This paper presents the latest results on block turbo codes and also an analysis of the possible implementations of a block turbo decoder circuit. Simulation results show that the SNR (signal to noise ratio) required to achieve a BER (Bit Error Rate) of 10^{-5} with block turbo codes is 2.5±0.2 dB from their Shannon limit for any code rate. We have identified three different solutions for implementing the block turbo decoder circuit. After discussing the advantages and disadvantages of the different solutions, we give the results of the degradation of the performance of the block turbo decoder circuit due to data quantization. Finally, in our conclusion, we discuss how to reduce the complexity of the algorithm for its implementation.

1 INTRODUCTION

The exceptional performance of "turbo codes" was first demonstrated by C. Berrou [1] in 1993. To construct the "turbo code," he used two concatenated recursive convolutional codes with a non-uniform interleaver inserted between the two recursive convolutional codes. Although this concatenated code is relatively powerful in terms of error correction, one must be very careful when decoding concatenated codes. To achieve near optimum decoding performance, C. Berrou proposed an iterative decoding algorithm based on soft decoding of the component codes and a soft decision of the decoded bits [2][3].

The results of convolutional turbo codes [4] encouraged us to search for block turbo codes. To construct the block turbo code, we used the principle of product codes [5] which were invented quite some time ago. Various iterative decoding algorithms have been proposed [6][7] for decoding product codes but the coding gains obtained with these decoding algorithms are well below what one would expect from such powerful codes. The main drawback of these algorithms is that they use algebraic decoders or hard decoders for decoding the component codes. To achieve near optimum decoding of product codes, in 1994 R. Pyndiah proposed a new iterative decoding algorithm [8][9] based on soft decoding of the component codes and a soft decision of the decoded bits. He obtained results similar to convolutional turbo codes and the coding gains were close to the theoretical coding gain expected from product codes when using Maximum Likelihood Sequence Decoding.

The object of this paper is to give more details about this new algorithm. We shall also analyse the complexity of this iterative decoding algorithm from a circuit design point of view.

2 PRINCIPLE AND PROPERTIES OF PRODUCT CODES

The concept of product codes is a simple and relatively efficient method to construct powerful codes (that is having a large minimum Hamming distance δ) using classic linear block codes [5]. Let us consider two systematic linear block codes C^1 having parameters (n_1,k_1,δ_1) and C^2 having parameters (n_2,k_2,δ_2) where n_i, k_i and δ_i $(i=1,2)$ stand for code length, number of information symbols and minimum Hamming distance respectively. The product code $P=C^1\otimes C^2$ is obtained by placing $(k_1.k_2)$ information bits in an array of k_1 rows and k_2 columns, coding the k_1 rows using code C^2 and coding the n_2 columns using code C^1, as illustrated in figure 1. It is shown [5] that all the n_1 rows are code words of C^2 exactly as the n_2 columns are code words of C^1 by construction. Furthermore, the parameters of the resulting product code P are given by $n=n_1.n_2$, $k=k_1.k_2$ and $\delta=\delta_1.\delta_2$ and the code rate R is given by $R=R_1.R_2$ where R_i is the code rate of code C^i. The theoretical results given above clearly show that it is possible to construct powerful product codes using linear block codes. As a general rule, the more powerful a code, the more difficult is the decoder.

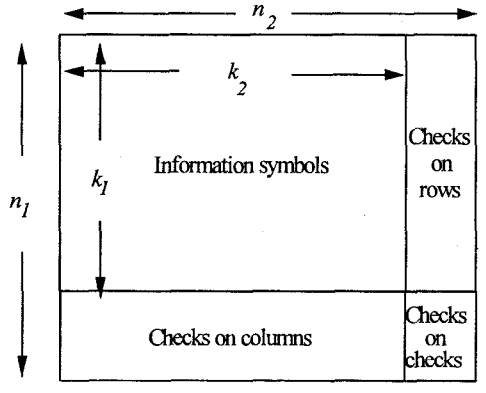

Figure 1: Construction of product code $P = C^1\otimes C^2$.

3 SOFT DECODING AND SOFT DECISION OF BLOCK CODES

If we consider the transmission of block coded binary symbols {-1,+1} using a QPSK modulation over a Gaussian channel, the samples $\mathbf{R} = (r_1, .., r_i, .., r_n)$ at the output of the coherent demodulator, conditionally to a transmitted code word $\mathbf{E} = (e_1, .., e_i, .., e_n)$ can be modeled by the following equation : $\mathbf{R} = \mathbf{E} + \mathbf{N}$ (1) where $\mathbf{N} = (n_1, .., n_i, .., n_n)$ are Additive White Gaussian Noise (AWGN) samples of standard deviation σ. Optimum sequence decoding of the received vector **R** is given by : $\mathbf{D}=\mathbf{C}^i$

if $\Pr\{\mathbf{E}=\mathbf{C}^i/\mathbf{R}\} > \Pr\{\mathbf{E}=\mathbf{C}^j/\mathbf{R}\} \quad \forall j \neq i$ (2)

where $\mathbf{C}^i = (c_1, .., c_l, ., c_n)$ is the i^{th} code word of code C with parameters (n,k,δ) and $\mathbf{D} = (d_1, .., d_l, ., d_n)$ is the decision corresponding to the maximum likelihood transmitted sequence conditionally to **R**. For received samples corrupted by AWGN, decoding rule (2) is simplified into :

$$\mathbf{D} = \mathbf{C}^i \text{ if } \left|\mathbf{R}-\mathbf{C}^i\right|^2 < \left|\mathbf{R}-\mathbf{C}^j\right|^2 \quad \forall j \neq i \tag{3}$$

where: $$\left|\mathbf{R}-\mathbf{C}^i\right|^2 = \sum_{l=1}^{n}(r_l - c_l^i)^2 \tag{4}$$

is defined as the squared Euclidean distance between **R** and $\mathbf{C}^i$. For block codes with a high code rate R, the number of code words 2^k is relatively large and optimum sequence decoding is too complex for implementation. In 1972, Chase proposed an algorithm [10] which approximates optimum sequence decoding of block codes with low computation complexity and a small performance degradation. Instead of reviewing all the code words j=1..2^k, the Chase algorithm searches for the code words at the Hamming distance within a sphere of radius (δ-1) centered on $Y = (y_1,..,y_l,..,y_n)$ where y_l is the sign of r_l ($\pm$1). To further reduce the number of reviewed code words, only the most probable code words within the sphere are selected by using channel information **R**. This search procedure can be decomposed into three steps:

step1: determine the position of the [δ/2] least reliable binary symbols of **Y** using **R**,

step2: form test patterns $\mathbf{T}^Q$ defined as all the n-dimensional binary vectors with a "1" in the [δ/2] least reliable positions and "0" in the other positions, two "1" in the [δ/2] least reliable positions and "0" in the other positions, ...,[δ/2] "1" in the [δ/2] least reliable positions and "0" in the other positions,

step3: decode $\mathbf{Z}^Q = \mathbf{Y} \oplus \mathbf{T}^Q$ using an algebraic decoder and memorize the code words $\mathbf{C}^Q$.

The Chase algorithm described above yields the maximum likelihood sequence **D** for a given soft input **R**. We must now compute the soft decisions associated with the maximum likelihood sequence **D** or the reliability of each component of **D**. This reliability function is given by the Log Likelihood Ratio (LLR) of the decision d_j. After the computations and the simplifications given in [8], the estimated normalized LLR of decision d_j, r'_j is equal to r_j plus w_j which is a function of the code words at the minimum Euclidean distance from **R** and $\{r_l\}$ with $l \neq j$. The term w_j is the additional information given by the decoder to the reliability of the decoded bit. Since w_j is a combination of independent and identically distributed random variables r_l, it is also a random variable with a Gaussian distribution. The approximated normalized LLR of d_j (r'_j) is an estimation of the soft decision of the block decoder. It has the same sign as d_j and its absolute value indicates the reliability of the decision. For each code word $\mathbf{C}^q$ found at step3, we compute its Euclidean distance from **R**:

$$M^Q = \left|\mathbf{R}-\mathbf{C}^Q\right|^2 \tag{5}$$

As in the Chase algorithm we select the code word $\mathbf{C}^d$ at the minimum Euclidean distance from **R**. Then we search for the code word $\mathbf{C}^c$ at a minimum Euclidean distance from **R** such that $c_j^c \neq c_j^d$. If we find such a code word then we compute the soft decision for bit d_j using the relation given below:

$$w_j = \left(\frac{\left(M^c - M^d\right)}{4}\right) c_j^d - r'_j$$

or else we use the relation $w_j = \beta \times c_j^d - r'_j$ (6)

where β is a constant which is a function of the BER and is optimized by simulation. Equations (6) are justified by the fact that w_j has the same sign as the $(\text{LLRN})_j$ but we have not been able to compute the amplitude of reliability since we have not found code word $\mathbf{C}^c$. We now have all the elements to perform the iterative decoding of product codes or block turbo codes which is described in the next section.

4 ITERATED DECODING OF PRODUCT CODES

On receiving the matrix **[R]** corresponding to a transmitted code word **[E]** of the product code, the first decoder performs the soft decoding of the rows (or columns) of the matrix, estimates the normalized LLR **[R']** as described in section 3 and gives as output **[W(1)]**. Then the next decoder performs the same operations on the columns (or rows) using as input:

$$[\mathbf{R}(1)] = [\mathbf{R}] + \alpha(1)[\mathbf{W}(1)] \tag{7}$$

where the coefficient $\alpha(k)$ is used to reduce the influence of $[\mathbf{W}(k)]$ in the first iterations when the BER is relatively high and thus when $[\mathbf{W}(k)]$ is not absolutely reliable. The decoding procedure described above is then iterated by cascading elementary decoders illustrated in figure 2. A close analysis of the iterative decoder shows that the parameters $\alpha(k)$ and $\beta(k)$ are not independent and that the optimum value for these parameters also depends on the product code used.

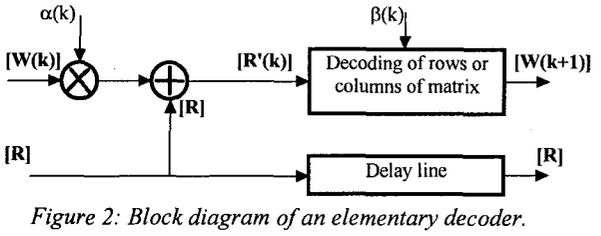

Figure 2: Block diagram of an elementary decoder.

This requires an optimization of the parameters for each product code. Concerning the set of test patterns used in the decoding algorithm, it is clear that the performance of the turbo decoder depends on the number of test patterns. We shall now give some further results concerning block turbo codes. We evaluated the performance of different block turbo codes based on extended BCH codes when using a QPSK modulation on a Gaussian channel. The BER is obtained using Monte Carlo simulations for different SNRs (signal to noise ratio (E_b/N_0)). We used identical BCH codes for coding the rows and columns of the product codes, that is C^1=C^2. The simulation results show that after iteration 4, the amelioration of the coding gain becomes negligible because of the steep slope of the BER curve. To evaluate the performance of the turbo decoder, we compared the coding gain at iteration 4 of the different

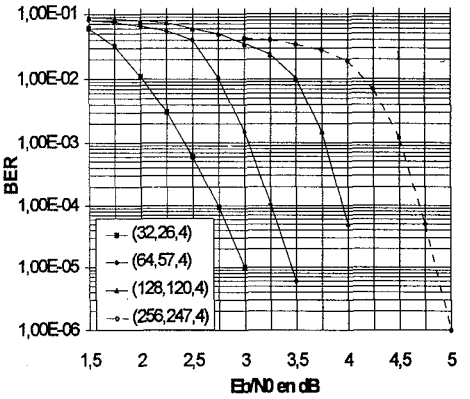

Figure 3: Bit Error Rate as a function of SNR (E_b/N_0) after four iterations when using a QPSK modulation.

turbo codes with an upper bound of the asymptotic coding gain G_a in the case of maximum likelihood sequence decoding and the SNR required for a given BER (10^{-5} for example) at iteration 4 with Shannon's theoretical minimum SNR required to achieve an error free transmission over a Gaussian channel. The result of our comparison is summarized in the table below,

Block turbo code	R	δ	SNR at 10^{-5}	Gain at 10^{-5}	ΔG (dB)	ΔS (dB)
(32,21,6)	0.43	36	2.3	7.0	4.9	2.6
(64,57,4)	0.79	16	3.4	6.0	5.0	2.4
(64,51,6)	0.64	36	2.7	6.6	7.0	2.3
(128,120,4)	0.88	16	4.0	5.4	6.0	2.7
(128,113,6)	0.78	36	3.3	6.1	8.4	2.3

Figure 4: Table comparing the performance of the block turbo codes with the theoretical optimum performance.

where ΔG is the difference between $(G_a)_{max}$ and the coding gain at a BER of 10^{-5} at iteration 4 and ΔS the difference between the SNR at a BER of 10^{-5} at iteration 4 and Shannon's theoretical limit. From the table in fig. 4 we observe that the block turbo codes are 2.5 ± 0.2 dB from their Shannon limit. ΔG increases with the code rate R and the minimum Hamming distance δ of the block turbo codes. We believe that this is due to the fact that the $(G_a)_{max}$ increases with R and δ and thus the asymptotic coding gain of the turbo code is reached at a lower BER. This explanation is supported by the slope of the BER curves which increases with R and δ.

5 INTEGRATION OF A BLOCK TURBO DECODER CIRCUIT

5.1 - General architecture.

A first analysis of the block turbo decoder algorithm shows that there are two different structures which can be adopted for hardware implementation [11]. The first solution is a modular structure where a module integrates the elementary decoder illustrated in fig. 2. These modules are then pipelined to realize the block turbo decoder-1. This solution which is similar to the one used for convolutional turbo codes [1] is very attractive from a practical point of view. However, each additional module introduces a delay given by the time required to fill the matrix **[R]** and to decode the rows (or columns) of matrix **[R]**. Sequential decoding is well adapted to the block turbo decoder algorithm since most of the procedures in the algorithm are easily implementable sequentially, for example: algebraic decoding, identification of the positions of the least reliable bits, reading and writing of the data in matrix **[R]**. In this solution, the time required to decode one bit in the elementary decoder must not exceed the inverse of the bit rate. This bit rate must be greater than 1 Mbit/s in order to keep the delay introduced by the block turbo decoder negligible.

In the second solution, several iterations are implemented in the same circuit. This reduces the decoding delay which is less than or equal to twice the time required to fill matrix **[R]** whatever the number of iterations. However, the time required to decode one bit in the elementary decoder must not exceed the bit duration divided by twice the number of iterations. The number of iterations performed by the block turbo decoder is thus limited by the bit rate and the time required to decode one bit. In this solution there are two alternatives for processing matrix **[R]** which are: sequential processing (block turbo decoder-2) or block processing of rows and columns (block turbo decoder-3). In sequential processing, the time required to decode one bit is relatively high, which limits the number of iterations in the circuit. In block processing, the processor handles blocks (rows or columns) instead of bits. This considerably reduces the time required to decode one bit. We can thus integrate more iterations in a circuit with a block processor than with a sequential processor. However block processing requires a specific memory architecture where the rows or columns of data **[R]** and **[W]** can be read and written blockwise.

5.2 - Memory block

Block turbo decoders in general require a large amount of memory to store data **[R]** and **[W]** which represents an important fraction of the surface occupied by the circuit. If we consider a q-bit quantization of the data, we need a $q(n_1 x n_2)$ memory per matrix. The number of matrices for every block turbo decoder shows a clear advantage for the second solution when designing a 4-iteration turbo decoder since block turbo decoder-2 and block turbo decoder-3 require only a $4q(n_1 x n_2)$ memory while block turbo decoder-1 requires a $32q(n_1 x n_2)$ memory. When considering the silicon technologies commercially available today, we must limit the block turbo codes to those with $(n_1 x n_2) \leq (128)^2$. Otherwise, external memories are required.

The size of the memory required also depends on the number of bits q used for the data quantization. We simulated the performance of the block turbo decoder when using 3, 4 and 5 bits for the linear quantization of the data **[R]** and **[W]**. The transmission of product code BCH(64,57,4)⊗BCH(64,57,4) over a Gaussian channel was used with QPSK signalling as in section 4. We observed a maximun degradation of 0.1dB at iteration 4 when using a 4 bit quantization which is a good compromise between complexity and performance.

With 0.8µm CMOS standard cell technology, the surface necessary for a 4(64x64) memory is about 4.3mm^2. Therefore, the size of the sequential memory can become very large according to the architecture used and the parameter n of the BCH code.

5.3 - Processing unit

To evaluate the surface, we considered a turbo decoder for the product code given by $C^1=C^2$=BCH(64,57,4). The reference library is a 0.8µm CMOS standard cell; the data **[R]** and **[W]** are coded using q=4bits (one bit for the sign and three for its reliability); sixteen test vectors are used.

We can decompose the algorithm functionally in the following way:

-read **[R]** and **[W]**, compute syndrome and initial parity,
-search for the five least reliable positions,
-compute the syndromes of test vectors,
-decode algebraically the test vectors,
-compute the square Euclidean distances,
-search for the minimal square Euclidean distance,
-compute the reliability of every symbol.

We evaluated the surface necessary for a sequential processing structure and a block processing structure. For the sequential processing structure, the surface is about 22mm^2 for the processing unit (number of gates: 43,000). If we add the memory surface, we have about 40mm^2 for a half-iteration. In the second structure, this surface reaches 75mm^2 (number of gates: 102,000). These surfaces are greater than those obtained with convolutional codes (the first CAS5093 circuit [12] has a surface of 64mm^2 with the same technology). We can therefore attempt to reduce this surface by:

-using a BCH(32,26,4) code or even a BCH(16,11,4) code, but the code rate R is smaller (thus, the memory size is divided by 4 or even 16),
-changing the CMOS technology: 0.6µm or 0.35µm,
-reducing the number of test vectors: 8 or 4,
-trying to simplify the weighting method which uses more than 50% of processing unit space.

6 Conclusion

In this paper we have presented a block turbo code which is the equivalent of convolutional turbo codes [1]. The block turbo codes (product codes) are decoded using an iterative decoding algorithm (block turbo decoder) [8] based on soft decoding and a soft decision of the component codes. We have shown the performance of block turbo codes with different code lengths and different minimum Hamming distance δ. We have observed that ΔS, which is the difference between the Shannon limit and the SNR at iteration-4 to achieve a BER of 10^{-5}, is practically independent of the code rate R and δ (ΔS = 2.5±0.2 dB). Then, we identified three possible solutions for the implementation of the block decoder circuit. The surface estimated seems too large for codes with $n \geq 128$. At present, a prototype is being developed at Télécom Bretagne. It will validate the new concepts introduced by the block turbo-decoders. This breadboard will contain eight 10,000 gate FPGA circuits (one per half-iteration). The characteristics of this prototype are:

-a product code with $C^1=C^2$=BCH(32,26,4),
-sequential decoding,
-weighting algorithm with 8 test vectors,
-programmable inputs for the α and β parameters.

We hope that first results will be available before the lecture concerning this paper.

Acknowledgment

This work has been financially supported by the Centre National d'Etudes des Télécommunications (CNET) and the Centre Commun d'Etudes de Télédiffusion et Télécommunications (CCETT). The authors would like to thank P. Combelles, D. Calonnec, C. Berrou and A. Glavieux for their helpful comments and discussions.

References

[1] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes (1)," *IEEE Int. Conf. on Comm.ICC' 93*, vol 2/3, May 1993, pp. 1064-1071.

[2] C. Berrou, P. Adde, E. Angui and S. Faudeil, "A low complexity Soft-output Viterbi decoder," *IEEE Int. Conf. on Comm. ICC' 93*, vol 2/3, May 1993, pp. 737-740.

[3] P. Adde, E. Angui, S. Faudeil et C. Berrou, "Architecture d'un décodeur de Viterbi à décision pondérée en sortie utilisant la méthode de mémorisation de chemin REA," *GRETSI'93*, Juan-Les-Pins, sept. 1993, pp 1123-1126.

[4] C. Berrou, A. Glavieux and R. Pyndiah, "Turbo-code: principe et applications," *GRETSI'95*, Juan-Les-Pins, Sept. 1995, pp 1369-1376.

[5] F.J. Macwilliams and N.J.A. Sloane, "The theory of error correcting codes," *North-Holland publishing company*, 1978, pp. 567-580.

[6] S. M. Reddy, "On decoding iterated codes," *IEEE Trans. Inform. Theory*, vol IT-16, Sept 1970, pp. 624-627.

[7] S. M. Reddy and J. P. Robinson, "Random error and burst correction by iterated codes," *IEEE Trans. Inform. Theory*, vol IT-18, Jan. 1972, pp. 182-185.

[8] R. Pyndiah, A.Glavieux, A. Picart and S. Jacq, "Near optimum decoding of products codes," in proc. of *IEEE GLOBECOM* '94 *Conference*, vol. 1/3, Nov.-Dec. 1994, pp. 339-343.

[9] S. Jacq, R. Pyndiah and A. Picard, "Algorithme Turbo: un nouveau procédé de décodage pour les codes produits," *GRETSI'95*, Juan-Les-Pins, Sept. 1995, pp 529-532.

[10] D. Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. Inform. Theory*, vol IT-18, Jan. 1972, pp. 170-182.

[11] O. Raoul, P. Adde and R. Pyndiah, "Architecture et conception d'un circuit turbo-décodeur de codes produits," *GRETSI'95*, Juan-Les-Pins, Sept. 1995, pp 981-984.

[12] C. Berrou and P. Adde, "Développement d'un circuit intégré turbo codeur/décodeur CAS5093, "*Internal report Télécom Bretagne*, April 1994.