Improving short frame turbo decoding towards ML decoding

Arnaud Guéguen and Damien Castelain

Mitsubishi Electric ITE, Telecommunication Laboratory
Immeuble Germanium, 80 avenue des Buttes de Coësmes, 35700 Rennes, France.
E-mail: gueguen@tcl.ite.mee.com, castelain@tcl.ite.mee.com

Abstract - **Turbo codes are becoming a widespread and mature coding scheme, as they are included in the standards of the third generation of mobile communication systems [1]. Whereas they were originally shown to perform very well from long to medium sizes of block [2], they are now also intended to be used for very small blocks. However graph theory predicts that in the case of short concatenated codes, turbo decoding may be suboptimal because of the presence of small loops in the network representation of the code [6,7,8,9]. The aim of this paper is first to quantify the suboptimality of turbo decoding in the case of short turbo codes, and then to propose a novative and simple scheme to partially overcome the performance loss. The Union Bound using the measured truncated weight distribution of the turbo code is recommended as the appropriate tool to quantify the suboptimality of iterative decoding compared to optimal Maximum Likelihood (ML) decoding.**

Keywords: **Turbo codes, Iterative decoding, Suboptimality of turbo decoding, ML bound, Improved decoding**

I. INTRODUCTION

Turbo codes were shown to perform very close to the theoretical limit of the channel capacity for long to medium sizes of blocks [2] thanks to their good weight distribution – i.e. rather large free distance and low multiplicity of low weight codewords – and to the ability of iterative decoding to perform near optimum decoding in the sense of Maximum Likelihood (ML).

Since, it has been shown that the turbo decoding algorithm is actually an instance of belief propagation on a loopy network [6,7,8] which appears to be a well known issue in graph theory, and it is admitted that it produces optimal performance, i.e. true a posteriori probabilities, provided that the network loops are long enough [8]. This is the case most of the time when the interleaver is large enough, which certainly explains the very good performance of the decoding algorithm for long turbo codes. However this becomes less obvious concerning short turbo codes.

In the first part of this article we introduce a tool to measure the suboptimality of the iterative decoding applied to a particular turbo code with a specific interleaver. In the second part we briefly recall in what cases turbo decoding may be suboptimal and we argue with some typical examples. Then we introduce and evaluate a simple post processing scheme that helps the turbo decoder converge closer to ML decoding. All results are presented on AWGN channel.

II. ML BOUND FOR A PARTICULAR TURBO CODE

Beyond their theoretical interest, error bounds enable to predict the performance of a code at high Signal to Noise Ratios (SNR), at Bit Error Rates (BER) that cannot be reached through simulations, and also to account for the efficiency of the decoding scheme. In the case of turbo codes the error bounds should also help predicting the so-called error floor that shows up at low BER. However most of the existing bounds for turbo codes have been derived considering a statistical interleaver, also called uniform interleaver [3,4,5], and they only provide the performance of a turbo code after averaging on all possible interleavers. Although it is theoretically interesting, such statistical approach is not well suited here, as we want to analyze the performance with one specific interleaver. In the rest of the article the bound we choose to use is the ML Union Bound as it is the most straightforward. Considering this bound, we show that the average weight distribution obtained with the uniform interleaver does not provide sufficiently accurate results and that it is necessary to use the exact weight distribution of the code.

Considering a Parallel Concatenated Convolutional Code (PCCC) obtained by concatenation of two Recursive Systematic Convolutional codes (RSC) denoted C_1 and C_2, separated by an interleaver of length N (see Figure 1), the turbo code transfer function using the uniform interleaver concept is given by equation (1) :

$$A^{C_p}(W,Z)=\sum_{w=1}^{N} W^w \frac{A_w^{C_1}(Z)\cdot A_w^{C_2}(Z)}{\binom{N}{w}} \triangleq \sum_{w,j} A_{w,j} W^w Z^j \qquad (1)$$

where $A_{w,j}$ is the number of codewords with information weight w and redundancy weight j, and $A_w^{C1}(Z)$ and $A_w^{C2}(Z)$ are the conditional weight enumerating polynomials of the two elementary codes as defined in [3]. The obtained weight distribution is used in the ML Union Bound equation for the BER :

$$P_b(e) \approx \frac{1}{2}\sum_m D_m \operatorname{erfc}\left(\sqrt{m\frac{R_c E_b}{N_0}}\right) \qquad (2)$$

where D_m is the normalized weight distribution given by:

$$D_m \stackrel{\Delta}{=} \sum_{j+w=m} \frac{w}{N} A_{wj} \qquad (3)$$

R_c is the code rate and E_b/N_0 is the SNR per information bit.

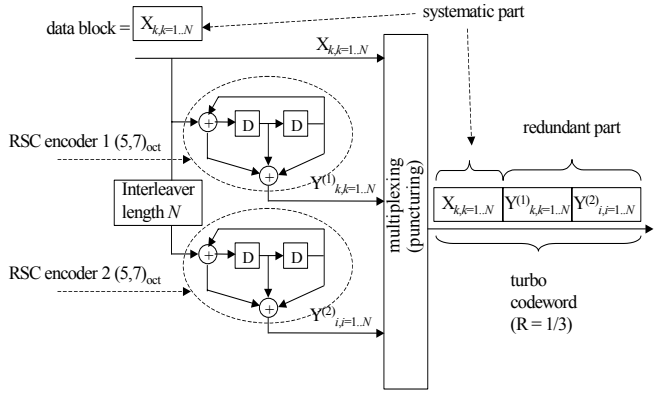

Figure 1. Parallel Concatenated Convolutional Code (PCCC) made of two $(5,7)_{oct}$ RSC constituent codes

As an example, we consider the PCCC made of two identical $(5,7)_{oct}$ RSC codes with interleaver length N=80 represented on Figure 1. The two trellises are terminated independently with tail bits. The average distribution D_m and the associated ML bound obtained with the uniform interleaver, are represented respectively on Figures 2 and 3. The saturation of the BER curve at low SNR is characteristic of the union bound and comes from the high multiplicity of higher weight codewords ($m > d_{min}$). More sophisticated bounds, like the one presented in [5] for instance, don't have this drawback. However in this study we are only interested in the lower part of the bound corresponding to high SNR, in the so-called "error floor" region. Indeed at lower SNR no bound is accurate enough to measure the optimality of the decoding process. Besides, the weight distribution of higher weight codewords and the performance of the turbo code at low SNR are much less sensitive to the interleaver.

The simulated performance of this PCCC using an optimized interleaver of length $N = 80$ is plotted together with the average union bound on Figure 3. The interleaver is designed to maximize the free distance of the code and the cycles length (see section III). Each elementary decoder uses a LogAPP and 20 decoding iterations are performed. It shows that the average bound is far away from the simulation curve and that it fails to predict the error floor effect as it is above the simulation curve. Indeed, at high SNR the first term of the summation in equation (2), corresponding to the minimum distance of the code, becomes the most significant and it turns out to be very different from one particular interleaver to another. In particular, the minimum distance with an optimized interleaver is expected to be larger than the one given by the uniform interleaver.

Consequently, for the union bound to reflect the ML decoding performance of the turbo code with one particular interleaver, at least the first terms of the summation in (2) should be the effective ones, instead of the terms given by equation (1). The true first values of D_m are obtained here by feeding the turbo encoder with all possible sequences of information weight below or equal to 5 and measuring the output codewords weights. Low weight codewords are given by low weight input sequences [4] and it is statistically unlikely that the free distance is produced by an input sequence of weight above 5 : indeed, low weight codewords are produced by input sequences that terminate both trellises, that is by sequences that terminate the first trellis and that are interleaved in sequences that also terminate the second one. Such mapping is all the more unlikely as the input weight is high.

The obtained measured truncated distribution is drawn together with the average distribution on Figures 2(a) and 2(b) and the corresponding union bound is plotted on Figure 3. The measured weight distribution D_m is truncated because only low input weight codewords have been enumerated. But as mentioned above, only the first values of D_m are relevant at high SNR, which corresponds to the part of the distributions shown on Figure 2(b). In particular the true free distance is 14, whereas the one obtained with the uniform interleaver is only 8. Unlike the average bound, this bound does not saturate at low SNR because the distribution is truncated. So, no comparison is possible in this region.

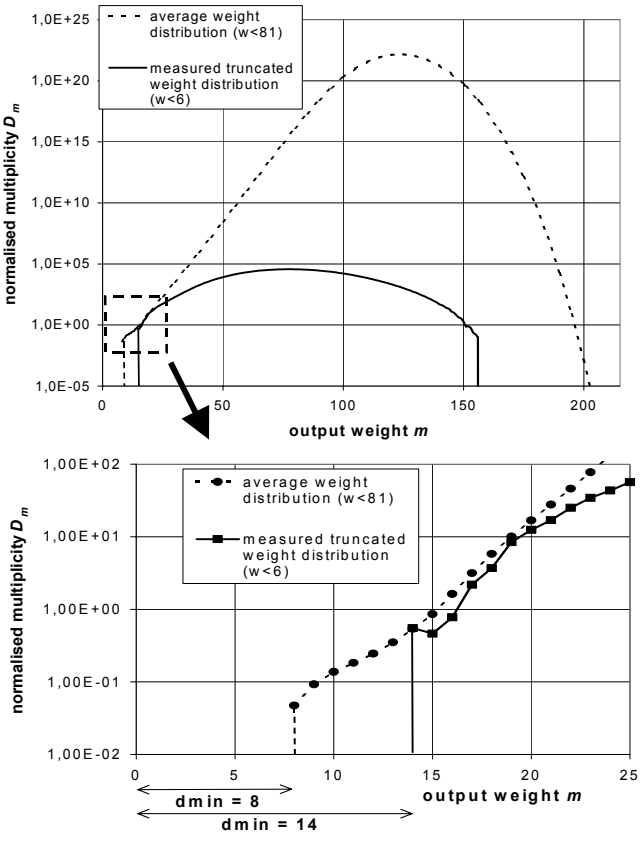

Figure 2. Average and measured truncated ($w < 6$) weight distributions D_m, interleaver length $N = 80$
(a) Complete distributions
(b) First terms (free distances)

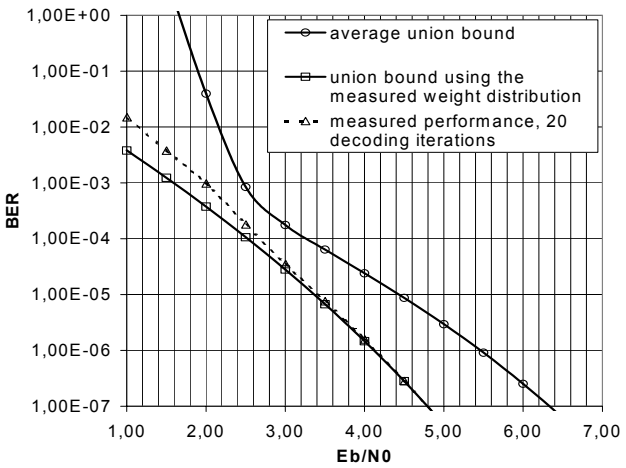

Figure 3. Union bounds using the average and measured truncated weight distributions D_m, interleaver length $N = 80$

The simulation curve converges very close to this second bound at high SNR. The error floor occurrence at a BER around 10^{-5} is obviously related to this bound and, through it, to the first values of D_m.

III. On the Suboptimality of Turbo Decoding

The union bound using the measured truncated weight distribution of the turbo code accurately reflects the ML decoding performance at high SNR. Therefore, if turbo decoding was optimal, the simulation curve should stick to this bound after a sufficiently large amount of decoding iterations.

However, according to graph theory [6,7,8], the turbo decoding algorithm is an instance of belief propagation on the graph representation of the turbo code (see Figure 4), and consequently it converges towards the true a posteriori probabilities (APP) provided that the graph is cycle free, i.e. provided the graph has a tree structure. Turbo codes graphs are never cycle free, nevertheless as recalled in reference [8], turbo decoding seems to converge towards APP provided that the cycles on the graph are long enough. Indeed, on graphs with long cycles, the close environment of any bit has almost a tree structure.

This thesis is consistent with the observation made in [9] : the turbo decoding algorithm converges all the better as the a priori information of each bit is decorrelated from its input extrinsic information. As the correlation between the output extrinsic information of neighboring nodes of the graph decreases along the cycles, this leads to the conclusion that the turbo decoding algorithm converges all the better as the correlation cycles are long. This is illustrated on Figure 4 : the Tanner graph of a turbo code is depicted, presenting a short secondary cycle of length 4, along which the output information remains strongly correlated.

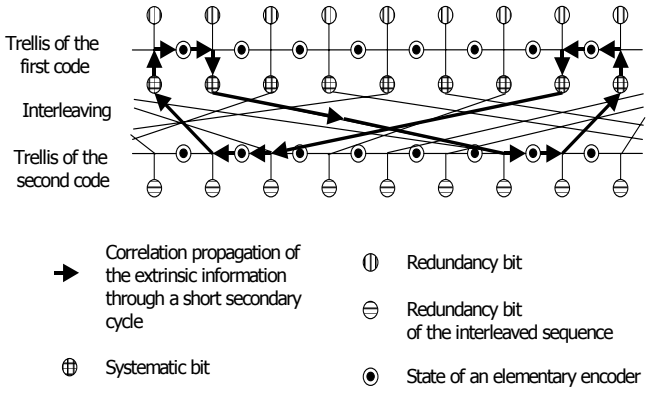

Figure 4. portion of the Tanner graph of a turbo code. Representation of a short cycle

Coming to practical cases, we consider the four following examples listed in Table 1.

Table 1
Example parameters

	N	Polynomials/ Constr. Length K	*dmin*	Minimum cycle length	Rate R_c
Ex. 1	80	$(5,7)_{oct}$ K= 3	14	5	1/3
Ex. 2	80	$(37,21)_{oct}$ K= 5	17	5	1/3
Ex. 3	80	$(5,7)_{oct}$ K = 3	14	2	1/3
Ex. 4	424	$(13,15)_{oct}$ K= 4	3	Long	¾

The measured truncated union bounds obtained as described in chapter II are plotted on Figure 5 together with the simulated performance. 30 decoding iterations are performed to guarantee optimum convergence.

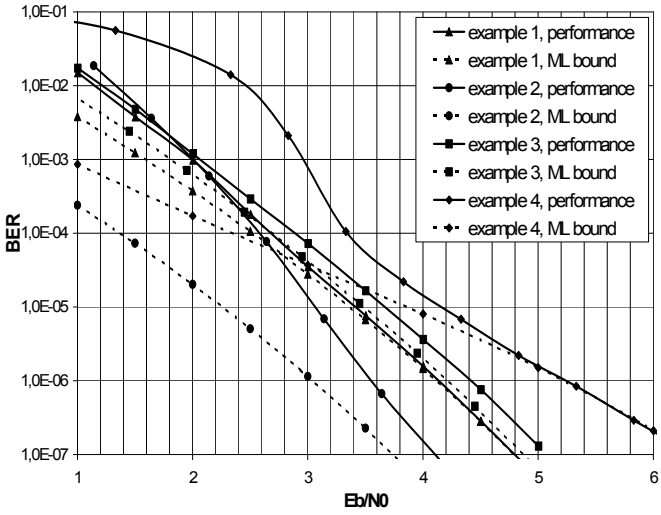

Figure 5. Simulated performance and ML performance (union bound using the measured truncated weight distribution) for the 3 examples of table 1

Turbo decoding is optimal in example 1 but it is suboptimal by 0.4dB at a BER of 10^{-7} in example 2. The minimum cycle lengths of examples 1 and 2 are the same but the constraint lengths of the constituent codes are different. This suggests that the minimum cycle

length to achieve optimal decoding may increase with the constraint length of the constituent codes : in example 1, K is smaller than the minimum cycle length and in the second case they are equal.

In example 3, the interleaver has been optimized only with respect to the minimum distance but the minimum cycle length is only 2, which is less than the constraint length of the constituent codes. The ML bound is close to the one of example 1 because the free distance is the same but the turbo code performance is worse because the turbo decoding is suboptimal. In fact, simulation results over several interleavers show that both the minimum cycle length and the multiplicity of short cycles play a role in the convergence of the turbo decoder. In a similar manner that the weight distribution of the code determines the error bound, the distribution of the cycles length seems to determine the convergence of the turbo decoder.

In example 4, the minimum distance is small due to the puncturing but the cycles are long : the ML bound is quite high and the turbo decoding is optimal.

Thus, the observations made on these examples are consistent with what is predicted by graph theory.

Conclusion : when designing an interleaver, both the minimum cycle length and the free distance of the code shall be maximized, as explained in [9]. However, in some systems, the interleaver may be poorly designed without any possibility to improve it. In other cases the interleaver is too small to produce cycles significantly larger than the constraint length of the constituent codes. For instance, in example 2, it may be difficult to optimize the interleaver so as to produce cycles larger than 5. In those cases, the turbo decoding algorithm is suboptimal and a performance gain is achievable by improving the decoding process. For instance in example 2 the potential gain is 0.4dB at a BER of 10^{-7}, and in example 3 the potential gain is 0.2dB at a BER of 10^{-6}.

IV. Improving short frame turbo decoding

The scheme that is proposed below aims at improving the turbo decoding towards ML decoding when the interleaver design or size prevent it from converging properly. The basic principle is the following : when errors are detected in the decoded block after a large number of iterations, the decoded erroneous binary sequence is turbo encoded and modulated. The resulting modulated bits sequence is then multiplied by a coefficient α, which is in the order of magnitude of 10^{-2}, and subtracted from the corresponding input sequence. The turbo decoding process is then applied over to this modified input sequence. Again, if residual errors are found in the decoded block, the same "post processing" scheme of re-encoding, re-modulation and subtraction is applied, and so on until there is no error left or until a maximum number of post processing iterations is reached. The proposed scheme is depicted on Figure 6. The switch on the left hand side is down when the received sequence is going to be turbo decoded for the first time, and up when post processing is performed.

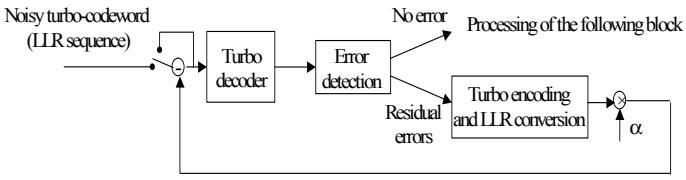

Figure 6. Post processing scheme

The underlying idea is that the contribution of the decoded sequence is subtracted from the received input sequence if the turbo decoder fails to correct all errors. Consequently, when turbo decoding is performed on the modified sequence, the path metric of the erroneous sequence – i.e. that was previously produced – in the turbo code trellis is reduced so that a neighboring sequence will be produced. If the residual errors were due to the suboptimality of the turbo decoding, the probability of error free convergence at the next step is increased.

Error detection can be implemented in various ways : an error detection code such as Cyclic Redundancy Check (CRC) may be concatenated to the data sequence before turbo encoding and used at the receiver side to check the integrity of the decoded data (CRC is envisaged in the UMTS standard [1]). Convergence detection based on the cross entropy criterion [10,12] or derived criteria [11] may also be used. In product codes such as Block Turbo Codes (BTC) [13], the syndrome of each elementary code is a straightforward error detection scheme. Performance evaluation is presented below with various detection schemes.

IV.1 Ideal error detection

The proposed scheme is applied to the following PCCC : $N = 80$, $R_c = 1/3$, constituent codes $(13,15)_{oct}$, $K = 4$, minimum cycle length = 4, $d_{min} = 15$, with a post processing coefficient $\alpha = 10^{-2}$, and a maximum of 20 post processing iterations. Perfect error detection is assumed in this first case. The results are plotted on Figure 7. The dashed line represents the performance without post processing.

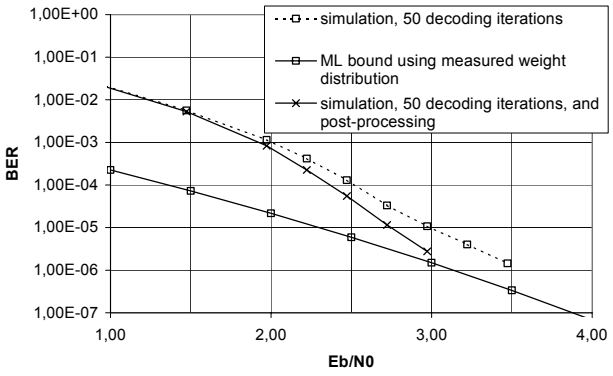

Figure 7. Performance with post processing, $\alpha = 10^{-2}$, until 20 post processing iterations, perfect error detection

The results indicate that the performance loss of turbo decoding compared to ML decoding (0.5dB in this case) are effectively due to the weakness of the decoding scheme itself and that it can be partially overcome with a simple scheme.

Assuming an unrealistic perfect error detection, and provided a very long decoding time, all possible codewords could be tried until the right one is found. However, because of the low number of post-processing iterations, the search space of the turbo decoder is several orders of magnitude below the total number of possible codewords.

IV.2 Error detection with CRC

Residual errors can be detected with Cyclic Redundancy Check bits (CRC) added at the end of the useful data sequence. In the case of UMTS, CRC are envisaged for any size of block and they were originally introduced for ARQ. When used as the error detection scheme within the post processing scheme they provide a performance gain, as shown on Figure 8. The turbo code parameters are the same as previously. 16 CRC bits are added before turbo encoding. At the receiver side, 20 turbo decoding iterations are performed. When using post processing, at most 10 post processing iterations are performed. Error detection is used both to interrupt the turbo decoding process if there is no error left before the last iteration, and in the post processing loop.

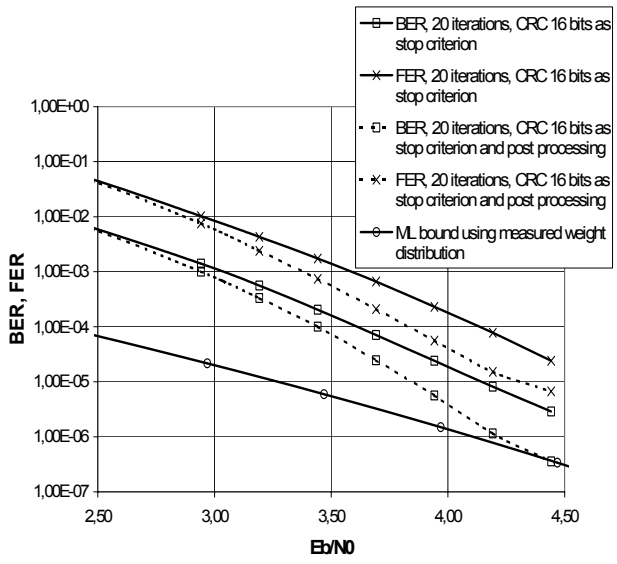

Figure 8. Performance with post processing, $\alpha = 10^{-2}$, until 10 post processing iterations, error detection with 16 CRC bits

The ML bound is the same as on figures 4 and 6, shifted by 0.96dB = 10log(80/(80-16)) to account for the E_b/N_0 penalty due to the 16 CRC bits. The performance gain is around 0.5 dB at a BER of 10^{-6} and the simulation curve is close to the ML bound. Approximately the same gain is achieved in terms of FER.

In this particular case, post processing can be seen as an alternative to ARQ with very small latency as no retransmission is needed.

IV.3 Convergence detection based on cross entropy

Stop criteria are used in turbo decoding to interrupt the iterative decoding process when a sequence is properly decoded before the last iteration, or, more precisely, when further iterations will provide no additional performance gain. Indeed, one wishes to avoid unnecessary computations, i.e. unnecessary latency and energy consumption. Various stop criteria proposed in the literature [10,11,12] are based on the Cross Entropy (CE) between the distributions of the estimates of the decoders outputs at each iteration.

In this part we evaluate the performance of the post processing scheme when the simplified CE criterion proposed in [11] is used as a stop criterion and as an error detection scheme : if the CE ratio between the current iteration and the first one drops below a threshold of 10^{-4} the sequence is assumed error free. On the contrary, if the ratio remains above this threshold until the last turbo decoding iteration, the sequence is considered erroneous and post processing is performed. The turbo coding and turbo decoding parameters are the same as in chapter IV.2.

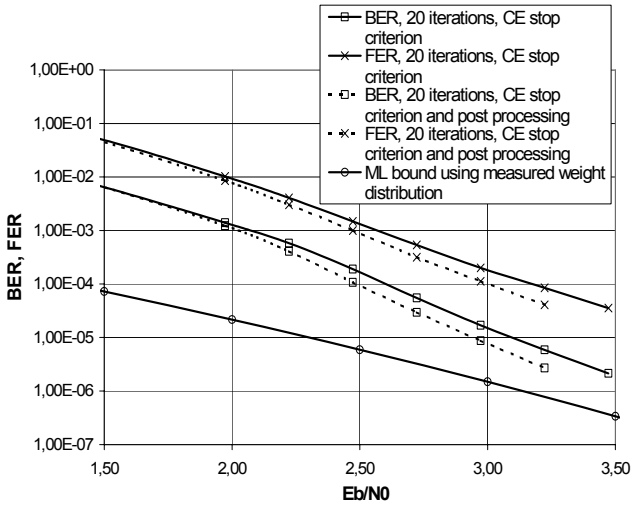

Figure 9. Performance with post processing, $\alpha = 10^{-2}$, until 10 post processing iterations, error detection using cross entropy

The performance gain is rather small but the BER curve is closer to the ML bound. The convergence detection scheme used may not provide accurate enough error detection. More sophisticated convergence detection may produce better results.

IV.4 Complexity considerations

The decoder hardware complexity is almost unchanged if a stop criterion is implemented. However the decoding latency of erroneous blocks may increase dramatically when the post processing scheme is applied : it may increase at most by a factor n_{max} where n_{max} is the

maximum number of post processing iterations (typically 10). Nevertheless, at high SNR, as a large majority of blocks is properly turbo decoded, the average latency only increases by a factor β given by :

$$\beta \approx (1 - FER) + n_{\max} * FER \quad (4)$$

In practical cases β is approximately 1. For instance $FER = 10^{-3}$, $n_{max} = 10$ yields $\beta = 1.009$.

The scheme may not be applied to highly delay sensitive applications because of the maximum latency, but it can be applied to any other application as an alternative to ARQ, for instance when the latency of the ARQ scheme is too long.

V. Conclusion

To investigate the suboptimality of turbo decoding applied to a specific turbo code, we compared the simulated performance of turbo decoding to the union bound using the measured truncated weight distribution of the code. Indeed, this bound provides the accurate performance of the considered turbo code with ML decoding at high SNR, unlike the union bound obtained with the so-called uniform interleaver that only gives the ML performance averaged on all possible interleavers. We observed through simulations that turbo decoding is suboptimal for interleavers yielding short cycles in the graph of the turbo code, which is very likely for short turbo codes. This observation is consistent with graph theory. Comparing the simulation performance with the error bound, we measured a performance loss around 0.5dB in some cases. At last, we proposed a simple and novative scheme based on error detection and re-encoding to partially overcome this loss. Depending on the robustness of the error detection scheme, a performance gain between 0.1 dB and 0.5 dB is obtained at a BER of 10^{-6}.

VI. References

[1] 3GPP TSG RAN WG1 25.212, v3.1.0 : "Multiplexing and Channel Coding"

[2] C. Berrou, A. Glavieux, P. Thitimajshima : "Near Shannon limit error-correction coding : Turbo codes", in Proc. IEEE ICC'93, Geneva, Switzerland, pp. 1064-1070, May 1993

[3] S. Benedetto, G. Montorsi : "Unveiling Turbo Codes : Some Results on Parallel Concatenated Coding Schemes", IEEE Transactions on Information Theory, September 28, 1995

[4] S. Benedetto, G. Montorsi : "Design of Parallel Concatenated Convolutional Codes", IEEE Transactions on Communications, vol. 44, pp. 591-600, May 1996

[5] T.M. Duman, M. Salehi : "New Performance Bounds for Turbo Codes", IEEE Transactions on Communications, vol. 46, No.6, June 1997

[6] N. Wiberg : "Codes and decoding on general graphs", PhD thesis no; 440, Dept. Elect. Eng., Linköping Univ., Sweden, 1996

[7] R.J. McElliece, D.J.C. McKay, J.F. Cheng : "Turbo Decoding as an Instance of Pearls's Belief Propagation Algorithm", IEEE J. on Selected Areas in Communications, V. 16 Number 2, February 1998

[8] E. Fabre, A. Guyader : "Dealing with short cycles in graphical codes", paper submitted to ISIT 2000

[9] J. Hokfelt, O. Edfors and T. Maseng : "Turbo codes : correlated extrinsic information and its impact on iterative decoding performance", IEEE Vehicular Technology Conference, Houston, Texas, May 1999

[10] J. Hagenhauer, E. Offer, L. Papke : "Iterative decoding of binary block and convolutional codes", IEEE Transactions on Information Theory, vol. 42, pp.429-445, March 1996

[11] R.Y. Shao, S. Lin, M.P.C. Fossorier : "Two Simple Stopping Criteria for Turbo Decoding", IEEE Transactions on Communications, vol. 47, No.8, August 1998

[12] M. Moher : "Decoding via cross entropy minimization", in Proc. IEEE Globecom Conf., Houston, TX, December 1993, pp.809-813

[13] R. Pyndiah, P. Combelles, and P. Adde, "A very low complexity block turbo decoder for product codes", in Proc., IEEE Globecom, pp. 101-105, 1996