Buletinul Ştiinţific al Universităţii "Politehnica" din Timişoara

Seria ELECTRONICĂ şi TELECOMUNICAŢII
TRANSACTIONS on ELECTRONICS and COMMUNICATIONS

Tom 49(63), Fascicola 2, 2004

A study on turbo decoding iterative algorithms

Horia Baltă[1], Maria Kovaci[2]

Abstract - In the paper, a study of some turbo decoding iterative algorithms: MAP, MaxLogMAP, LogMAP, is presented. For the correction of the approximation used in the MAxLogMAP algorithm two methods are proposed obtaining two LogMAP algorithm variants. All algorithms variants have been simulated to make possible a comparison from the bit error rate (BER) point of view, in order to provide an optimization for each algorithm.

The simulations were made for AWGN channel. Two component codes with generator matrix: G_1 =[1, 5/7] and two interleaver types: pseudo-random [1] and S-interleaver (S=29) are used. The interleaving length is N=1784. The number of emitted blocks in one simulation depends of signal to noise ration, SNR, to obtain a good precision of resulted curve.

Keywords: Turbo codes, MAP algorithm, trellis.

I. INTRODUCTION

The Maximum A-Posteriori (MAP) algorithm, proposed by Bahl, Cocke, Jelinek and Raviv (1974), is frequently used after the turbo codes discovery realized by Berrou and *al.* [1]. Essentially, MAP algorithm, [2], calculate the Log Likelihood Ratio, LLR, under the form:

$$L(u_k|\underline{y}) = ln\left(\frac{\sum_{\substack{(\hat{s},s)\Rightarrow \\ u_k=+1}} \alpha_{k-1}(\hat{s})\cdot\gamma_k(\hat{s},s)\cdot\beta_k(s)}{\sum_{\substack{(\hat{s},s)\Rightarrow \\ u_k=-1}} \alpha_{k-1}(\hat{s})\cdot\gamma_k(\hat{s},s)\cdot\beta_k(s)}\right) \qquad (1)$$

where: $\alpha_{k-1}(\hat{s})$ $P(S_{k-1} = \hat{s} \wedge \underline{y}_{j<k})$ is the probability that the encoder trellis was in $\hat{s}$ state at instant $k-1$ and the received channel sequence, before this moment, is $\underline{y}_{j<k}$.

$$\alpha_k(s) = \sum_{\text{alls } \hat{s}} \gamma_k(\hat{s},s)\cdot\alpha_{k-1}(\hat{s}) \qquad (2)$$

$\beta_k(s) = P(\underline{y}_{j>k}/S_k = s)$ is the probability that, having been given the trellis state s at instant k, the received channel sequence, after this moment, to be $\underline{y}_{j>k}$,

$$\beta_{k-1}(\hat{s}) = \sum_{\text{alls } \hat{s}} \gamma_k(\hat{s},s)\cdot\beta_k(s) \qquad (3)$$

$\gamma_k(\hat{s},s) = P(\{\underline{y}_k \wedge S_k = s\}/S_{k-1} = \hat{s})$ is the probability that the encoder trellis took the transition from state $\hat{s}$ to state s and the received channel sequence for this transition is $\underline{y}_k$.

$$\gamma_k(\hat{s},s) = C\cdot e^{(u_k L(u_k)/2)}\cdot exp\left(\frac{E_b}{2\sigma^2}\cdot 2\cdot a\sum_{i=1}^{n} y_{ki}\cdot x_{ki}\right) \qquad (4)$$

In relation (4), u_k is the value of information bit for the trellis branch, $L(u_k)$ is the extrinsic information for the k-th bit, E_b is the energy of the information bit, σ^2 is the noise power, y_{ki} and x_{ki} represent corresponding values of all the bits attached to the branch which makes the liaison of states $\hat{s}$ and s, from reception (y_{ki}), respectively, emission (x_{ki}).

Fig.1 presents the computation way of forward-α and backward-β coefficients, for a part of trellis of convolutional code with constraint length $K=3$.

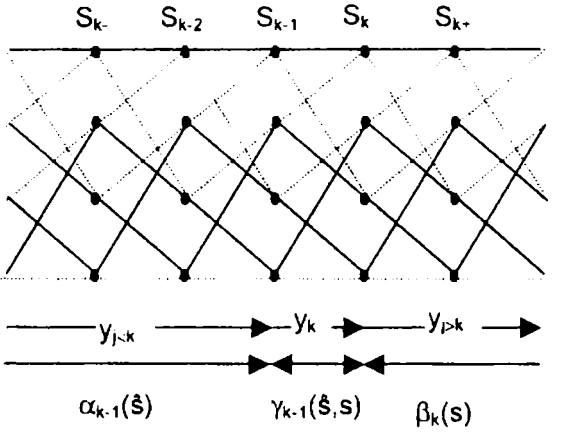

Fig.1 The code trellis with G = [1, 5/7]. The continuous line corresponds of the input bit value 1

Due to the exponential and logarithm operators in relations (1) and (4), the MAP algorithm is difficult to be implemented.

Like an alternative, *the MaxLogMAP algorithm* is easier to be implemented, due to the approximation:

[1] [2] Facultatea de Electronică şi Telecomunicaţii, Departamentul Comunicaţii Bd. V. Pârvan Nr. 2, 300223 Timişoara, balta@etc.utt.ro; kmaria@etc.utt.ro

$$\ln\left(\sum_i e^{x_i}\right) \approx \max_i(x_i) \quad (5)$$

So, the computing relations, in the MaxLogMAP algorithm case, are the following:

$$A_k(s) \doteq \ln(\alpha_k(s)) \approx \max_{\hat{s}}(A_{k-1}(\hat{s}) + \Gamma_k(\hat{s}, s)) \quad (6)$$

$$B_k(s) \doteq \ln(\beta_k(s)) \approx (B_k(s) + \Gamma_k(\hat{s}, s)) \quad (7)$$

$$\Gamma_k(\hat{s}, s) \doteq \ln(\gamma_k(s)) =$$

$$= \hat{C} + \frac{1}{2} \cdot u_k \cdot L(u_k) + \frac{L_c}{2} \sum_{i=1}^{n} y_{ki} \cdot x_{ki} \quad (8)$$

$$L(u_k|y) = \max_{\substack{(\hat{s},s)\Rightarrow \\ u_k=+1}} (A_{k-1}(\hat{s}) + \Gamma_k(\hat{s},s) + B_k(s)) - \max_{\substack{(\hat{s},s)\Rightarrow \\ u_k=-1}} (A_{k-1}(\hat{s}) + \Gamma_k(\hat{s},s) + B_k(s)) \quad (9)$$

The implementation simplification price of the MaxLogMAP algorithm is the reduction of the performances (of the BER) with 0,2 dB versus the MAP algorithm.

LogMAP Algorithm, proposed by Robertson and *al.* [3], corrects the approximation used by MaxLogMAP algorithm and is a little bit more complicated than it.

$$\ln(e^{x_1} + e^{x_2}) = \max(x_1, x_2) + \ln(1 + e^{-|x_1 - x_2|}) = \max(x_1, x_2) + f_c(|x_1 - x_2|) \quad (10)$$

II. THE TRELLIS CLOSING

In function of the trellis closing the alpha and beta coefficients are initialized. The initial trellis closing means the coder initialization with a predefined state. This state is also known by the decoder. So, the initialization of the alpha and beta coefficients can be done. With the exception of circular coding, this initial state is zero. The final trellis closing is more difficult to be realized. It is done (excepting the circular code) with the price of insertion of the M (the code memory) redundant bits in the information sequence. This fact realizes the reduction of the transmission rate from 1/2 to the following value:

$$R_{cc} \quad (N - M) \cdot 2N. \quad (11)$$

The trellis closing gives the advantage of the initial state knowledge (and/or of the final state), fact which leads to the firm knowledge of the alpha coefficients (at the beginning of trellis) and beta coefficients (at the end of the trellis). In the case of the unclosed trellis these coefficients can only be predicted probabilistically.

A turbo code implies at least two coders, C1 and C2. At each coder corresponds a trellis. Different trellis closing techniques can be used for these coders. In this paper we investigate few trellis closing methods for a turbo code (parallel). The table 1 presents these methods.

Table 1

Variant	Start C1, C2	Final C1, C2	Coding rate
01	0, 0	0, ?	(N-M)/3N
11	0, 0	?, ?	1/3
C	Sx, Sy	Sx, Sy	1/3

01. In this case, the first coder closes the trellis on both extremities, it inserts M redundant bits after the N-M information bits. The second coder can not do the same final closing due to the interleaving of the input sequence. So, the second trellis is not closed. The first decoder initializes the alpha coefficient, which corresponds at the front end of the trellis to the zero state, with the probability 1, and the other alpha coefficients with the probability zero. The first decoder treats the beta coefficients in the same way at the end of the trellis. The second decoder acts in the same way, like the first, for the alpha coefficients. The beta coefficients of the second decoder, can be initialized by the one of the following methods:

01s. –The beta coefficients are met equals with the values of the alpha coefficients obtained at the last iteration. This is called the "soft" initialization;

01h. –This method initializes the beta coefficient, that corresponds to the state with the highest alpha coefficient, at probability 1 and the others beta coefficients at the probability zero. This is called the "hard" initialization;

01e. – This method makes the beta coefficients to have the same probability. This is called the "equal probability" initialization;

11. None of the trellises is final closed. The advantage, in this case, is a higher coding rate. But this coding rate increase can not be observed if N>>M. Both decoders must initialize the beta coefficients in one of the three ways enounced above. In this paper we implement only the soft initialization. The case of both trellis final closing is possible only with some modifications of the interleaving between the two coders.

C. There is the possibility, using a pre coding technique, to find, for any data sequence x, an initial state S_0 of the coder identically with its final state. So the coding becomes circular. The decoder does not know the state S_0, but knows that it can use the final state like initial state. So, it must to do at least a forward recurrence. We are implemented and simulated the following variants of circular turbo code:

C1 – the decoder realizes a forward recurrence and computes a final state, S_0. The alpha and beta coefficients are initialized with S_0, the backward recurrence is made and the forward recurrence is

remade. It memorizes the new state S_0 for the starting of the next iteration.

C2 – the decoder realizes the both recurrences in the soft variant and retains, for the next iteration, with the role of S_0, the beta coefficients values from the end of the backward recurrence.

C3 – the decoder realizes the both recurrences in the soft variant plus one for the alpha coefficients, only. The initial state for that second recurrence is done by the finals values of beta coefficients of the last iteration. The final coefficients of the second forward recurrence give the state to be stored for the next iteration.

C4 – the decoder realizes the forward recurrence and build a S_0 state in a hard decision (it searches the alpha coefficients maximum). It retains this state for the next iteration and also makes the backward recurrence and remakes the forward recurrence.

III. THE LOGMAP ALGORITHM. IMPLEMENTATION.

The variants of LogMAP algorithm differs by the correction term approximation way, described in equation (10):

$$f_c(x) \quad ln(1+e^{-x}), \quad x \geq 0 \tag{12}$$

The functions that approximate $f_c(x)$ must be easy to implement and they must reproduce the most exactly possible the form of this function. Two approximations were proposed in this paper, indicated in Fig. 2 and Fig.3.

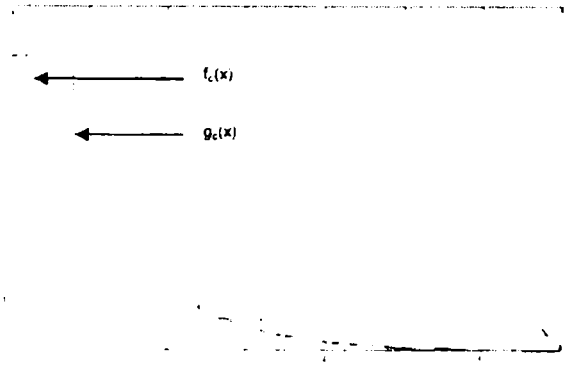

Fig.2 The rectangular approximation way

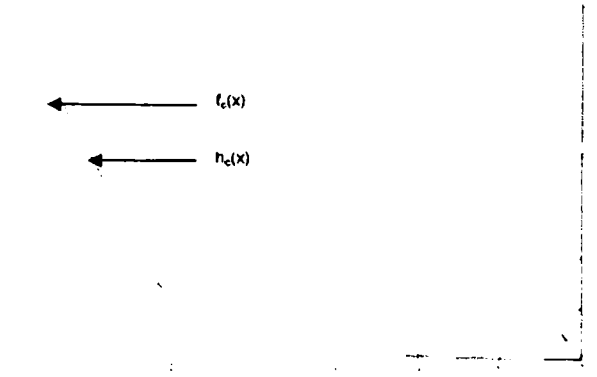

Fig.3 The linear approximation way

The rectangular variant proposed by Robertson and ail. [3] is a zero order extrapolation of the function $f_c(x)$. The values of the function $g_c(x)$ are in the set {0.6, 0.3, 0.14, 0.065, 0.03, 0.014, 0.005, 0.002, 0}. The linear variant corresponds to an approximation of $f_c(x)$ of the form:

$$h_C(x) = \begin{cases} 0.7 - \dfrac{0.7}{x_O} \cdot x, & x \leq x_O \\ 0, & x > x_O \end{cases} \tag{13}$$

and by numerical approximation was obtained the value $x_o = 2.347$ for which $h_c(x)$ realizes the better approximation of $f_c(x)$.

IV. EXPERIMENTAL RESULTS

In the figure 4 are presented the curves BER(SNR) obtained with the three MAP algorithms variants "01" plus the MAP algorithm "11s". Despite the fact that for signal to noise ratios inferior to 1 dB the performances are identical, up to this value the results show that the variant 01e is better. It is followed, in order, by the variants: 01s, 11s and 01h. These results show that at low signal to noise ratios the errors are produced exclusively by the bad selection of the path in the trellis and up 1 dB the errors due to the trellis non closing have a higher weight.

In figure 5 are represented the BER(SNR) curves obtained with the four variants of the circular MAP algorithm already defined in comparison with the best MAP algorithm: 01e. The first three circular MAP variants have similar performances, inferior to the performances of the variant 01e.

Tacking into account all the results already presented it results that the hard variant is not a good solution.

The simulation results realized with 1/3 rate RSC turbo code (parallel) with G=[1,5/7], which utilizes in the variant 00s the LogMAP algorithms are compared in Fig.6 with the results obtained with the best MAP variant: 01e. From figure results that all the two LogMAP variants are better than the MAP at least for values of the signal to noise ratio inferior to 1 dB. Up this value the curves are not very accurate but obviously the performances are similar. The curves reduced precision is due to the reduced volume of simulations.

- Despite the fact that practical implementations of the LogMAP algorithm are faster than those of the MAP algorithm the simulation programs work slower in the case of the LogMAP algorithm.

Between the two LogMAP algorithm variants the results show that the linear one is better. These conclusions must be verified also for other component codes.

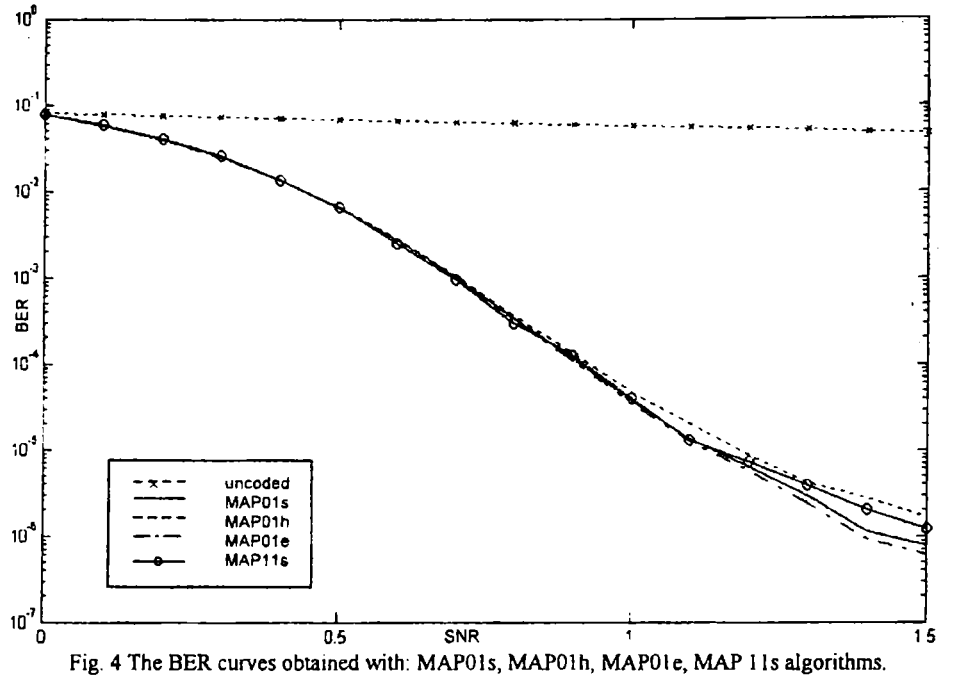

Fig. 4 The BER curves obtained with: MAP01s, MAP01h, MAP01e, MAP 11s algorithms.

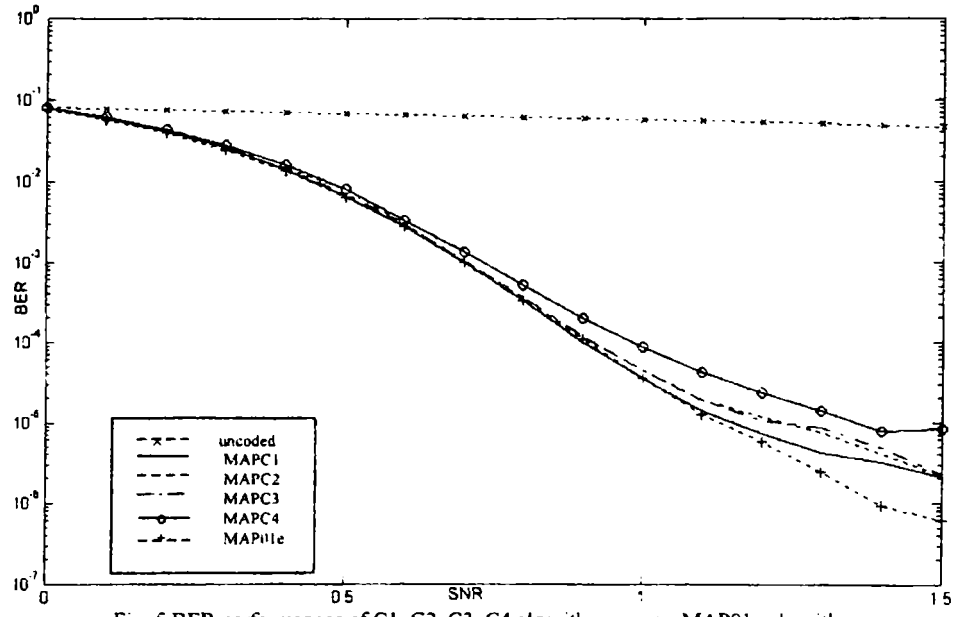

Fig. 5 BER performances of C1, C2, C3, C4 algorithms versus MAP01e algorithm.

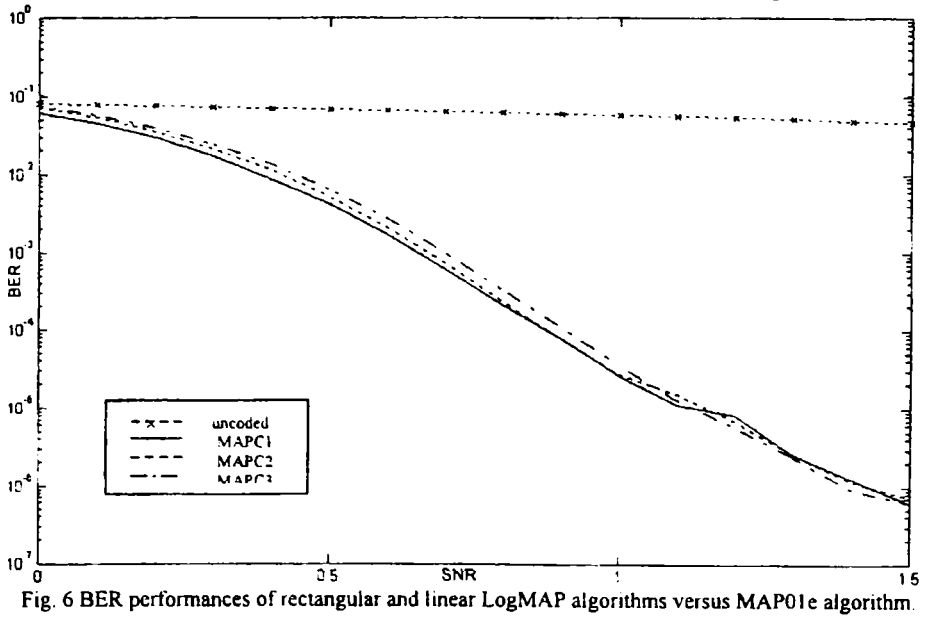

Fig. 6 BER performances of rectangular and linear LogMAP algorithms versus MAP01e algorithm.

V. CONCLUSIONS

In the paper, a study of some turbo decoding iterative algorithms: MAP, MaxLogMAP, LogMAP, was presented. For the correction of the approximation used in the MAxLogMAP algorithm, two methods were proposed, obtaining two LogMAP algorithm variants. All algorithms variants have been simulated to make possible a comparison from the bit error rate point of view, in order to provide an optimization for each algorithm.

VI. REFERENCES

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

[2] L.Hanzo, T.H.Liew, B.L.Yeap, "*Turbo Coding, Turbo Equalisation and Space-Time Coding for Transmission over Fading Channels*", John Wiley & Sons Ltd, England, 2002

[3] P. Robertson, E.Villebrun, P.Hoeher, „*A Comparison of Optimal and Sub*-Optimal MAP Decoding Algorithms Operating in the Log Domain", Proceedings of the International Conference on Communications, Seattle, USA, pag. 1009-1013, iunie 1995