Transactions Papers

Near Optimum Error Correcting Coding And Decoding: Turbo-Codes

Claude Berrou, *Member, IEEE*, and Alain Glavieux

Abstract— **This paper presents a new family of convolutional codes, nicknamed turbo-codes, built from a particular concatenation of two recursive systematic codes, linked together by nonuniform interleaving. Decoding calls on iterative processing in which each component decoder takes advantage of the work of the other at the previous step, with the aid of the original concept of extrinsic information. For sufficiently large interleaving sizes, the correcting performance of turbo-codes, investigated by simulation, appears to be close to the theoretical limit predicted by Shannon.**

I. Introduction

CONVOLUTIONAL error correcting or channel coding has become widespread in the design of digital transmission systems. One major reason for this is the possibility of achieving real-time decoding without noticeable information losses thanks to the well-known soft-input Viterbi algorithm [1]. Moreover, the same decoder may serve for various coding rates by means of puncturing [2], allowing the same silicon product to be used in different applications. Two kinds of convolutional codes are of practical interest: nonsystematic convolutional (NSC) and recursive systematic convolutional (RSC) codes. Though RSC codes have the same free distance d_f as NSC codes and exhibit better performance at low signal to noise ratios (SNR's) and/or when punctured, only NSC codes have actually been considered for channel coding, except in Trellis-coded modulation (TCM) [3]. Section II presents the principle and the performance of RSC codes, which are at the root of the study expounded in this article.

For a given rate, the error-correcting power of convolutional codes, measured as the coding gain at a certain binary error rate (BER) in comparison with the uncoded transmission, grows more or less linearly with code memory ν. Fig. 1 (from [4]) shows the achievable coding gains for different rates, and corresponding bandwidth expansion rates, by using classical NSC codes with $\nu = 2, 4, 6$ and 8, for a BER of 10^{-6}. For instance, with $R = 1/2$, each unit added to ν adds about 0.5 dB more to the coding gain, up to $\nu = 6$; for $\nu = 8$, the additional gain is lower. Unfortunately, the complexity of the decoder is not a linear function of ν and it grows exponentially as $\nu \cdot 2^\nu$. Factor 2 represents the number of states processed by the decoder and the multiplying factor ν accounts for the complexity of the memory part (metrics and survivor memory). Other technical limitations like the interconnection constraint in the silicon decoder lay-out, make the value of six a practical upper limit for ν for most applications, especially for high data rates.

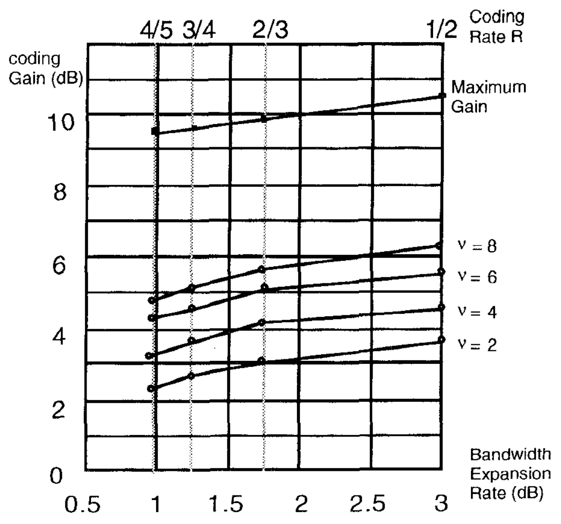


Fig. 1. Coding gains at BER equal to 10^{-6}, achievable with NSC codes (three-bit quantization, from [4], and maximum possible gains for 1/2, 2/3, 3/4, and 4/5 rates in a Gaussian channel, with quaternary phase shift keying (QPSK) modulation.

In order to obtain high coding gains with moderate decoding complexity, concatenation has proved to be an attractive scheme. Classically, concatenation has consisted in cascading a block code (the outer code, typically a Reed-Solomon code) and a convolutional code (the inner code) in a serial structure. Another concatenated code, which has been given the familiar name of *turbo-code*, with an original parallel organization of two RSC elementary codes, is described in Section III. Some comments about the distance properties of this composite

Paper approved by D. Divsalar, the Editor for Coding Theory and Applications of the IEEE Communications Society. Manuscript received March 3, 1995; revised December 6, 1995 and February 26, 1996. This paper was presented at ICC'93, Geneva, Switzerland, May 1993.

The authors are with Ecole Nationale Supérieure des Télécommunications de Bretagne, BP 832 29285 BREST CEDEX, France.

Publisher Item Identifier S 0090-6778(96)07373-4.

Authorized licensed use limited to: Yonsei University. Downloaded on December 26, 2009 at 22:55 from IEEE Xplore. Restrictions apply.

code, are propounded. When decoded by an iterative process, turbo-codes offer near optimum performance. The way to achieve this decoding with the *Maximum A Posteriori* (MAP) algorithm is detailed in Sections IV, V, VI, and some basic results are given in Section VII.

II. Recursive Systematic Convolutional Codes

A. Introduction

Consider a binary rate $R = 1/2$ convolutional encoder with constraint length K and memory $\nu = K - 1$. The input to the encoder at time k is a bit d_k and the corresponding binary couple (X_k, Y_k) is equal to

$$X_k = \sum_{i=0}^{\nu} g_{1i} d_{k-i} \qquad g_{1i} = 0, 1 \tag{1a}$$

$$Y_k = \sum_{i=0}^{\nu} g_{2i} d_{k-i} \qquad g_{2i} = 0, 1 \tag{1b}$$

where $G_1 : \{g_{1i}\}, G_2 : \{g_{2i}\}$ are the two encoder generators, expressed in octal form. It is well known that the BER of a classical NSC code is lower than that of a classical nonrecursive systematic convolutional code with the same memory ν at large SNR's, since its free distance is smaller than that of a NSC code [5]. At low SNR's, it is in general the other way round. The RSC code, presented below, combines the properties of NSC and systematic codes. In particular, it can be better than the equivalent NSC code, at any SNR, for code rates larger than 2/3.

A binary rate RSC code is obtained from a NSC code by using a feedback loop and setting one of the two outputs X_k or Y_k equal to the input bit d_k. The shift register (memory) input is no longer the bit d_k but is a new binary variable a_k. If $X_k = d_k$ (respectively, $Y_k = d_k$), the output Y_k (resp. X_k) is defined by (1b) [respectively, (1a)] by substituting d_k for a_k and variable a_k is recursively calculated as

$$a_k = d_k + \sum_{i=1}^{\nu} \gamma_i a_{k-i} \tag{2}$$

where γ_i is respectively equal to g_{1i} if $X_k = d_k$ and to g_{2i} if $Y_k = d_k$. Equation (2) can be rewritten as

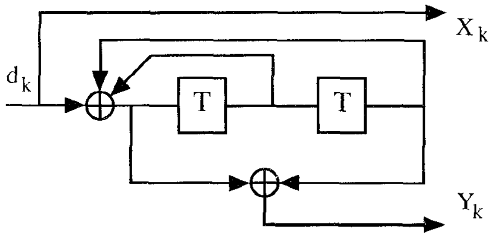
$$d_k = \sum_{i=0}^{\nu} \gamma_i a_{k-i}. \tag{3}$$

Taking into account $X_k = d_k$ or $Y_k = d_k$, the RSC encoder output $C_k = (X_k, Y_k)$ has exactly the same expression as the NSC encoder outputs if $g_{10} = g_{20} = 1$ and by substituting d_k for a_k in (1a) or (1b).

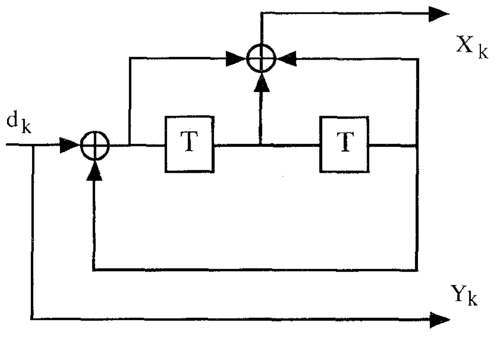
Two RSC encoders with memory $\nu = 2$ and rate $R = 1/2$, obtained from a NSC encoder defined by generators $G_1 = 7, G_2 = 5$, are depicted in Fig. 2.

Generally, we assume that the input bit d_k takes values zero or one with the same probability. From (2), we can show that variable a_k exhibits the same statistical property

$$\Pr\{a_k = 0 / a_{k-\nu} = \varepsilon_\nu, \cdots a_{k-i} = \varepsilon_i, \cdots a_{k-1} = \varepsilon_1\}$$
$$= \Pr\{d_k = \varepsilon\} = \tfrac{1}{2} \tag{4}$$



(a)



(b)

Fig. 2. Two associated Recursive systematic convolutional (RSC) encoders with memory $\nu = 2$, rate $R = 1/2$ and generators $G_1 = 7, G_2 = 5$.

where ε is equal to

$$\varepsilon = \sum_{I=1}^{\nu} \gamma_i \varepsilon_i; \quad \varepsilon_i = 0, 1. \tag{5}$$

Thus, the transition state probabilities $\pi(\boldsymbol{S}_k = \boldsymbol{m} / \boldsymbol{S}_{k-1} = \boldsymbol{m}')$, where $\boldsymbol{S}_k = \boldsymbol{m}$ and $\boldsymbol{S}_{k-1} = \boldsymbol{m}'$ are, respectively, the encoder state at time k and at time $(k-1)$, are identical for the equivalent RSC and NSC codes; moreover these two codes have the same free distance d_f. However, for a same input sequence $\{d_k\}$, the two output sequences $\{X_k\}$ and $\{Y_k\}$ are different for RSC and NSC codes.

When puncturing is considered, some output bits X_k or Y_k are deleted according to a chosen perforation pattern defined by a matrix $\boldsymbol{P}$. For instance, starting from a rate $R = 1/2$ code, the matrix $\boldsymbol{P}$ of rate 2/3 punctured code can be equal to

$$\boldsymbol{P} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}. \tag{6}$$

For the punctured RSC code, bit d_k must be emitted at each time k. This is obviously done if all the elements belonging to the first or second row of matrix $\boldsymbol{P}$ are equal to one. When the best perforation pattern for a punctured NSC code is such that matrix $\boldsymbol{P}$ has null elements in the first and the second rows, the punctured recursive convolutional code is no longer systematic. In order to use the same matrix $\boldsymbol{P}$ for both RSC and NSC codes, the RSC encoder is now defined by (3), (7), and (8)

$$Y_k = \sum_{i=0}^{\nu} \lambda_i a_{k-i} \tag{7}$$

$$X_k = d_k \tag{8}$$

Authorized licensed use limited to: Yonsei University. Downloaded on December 26, 2009 at 22:55 from IEEE Xplore. Restrictions apply.

where coefficients γ_i [see (3)] and λ_i are, respectively, equal to g_{1i} and g_{2i} when element $p_{1j}; 1 \leq j \leq n$ of matrix $\boldsymbol{P}$ is equal to one and to g_{2i} and g_{1i} when p_{1j} is equal to zero.

B. Recursive Systematic Code Performance

In order to compare the performance of RSC and NSC codes, we determined their weight spectrum and their BER. The weight spectrum of a code is made up of two sets of coefficients $a(d)$ and $W(d)$ obtained from two series expansions related to the code transfer function $T(D, N)$ [6]

$$T(D,N)\Big|_{N=1} = \sum_{d=d_f}^{\infty} a(d)D^d \tag{9}$$

$$\frac{\partial T(D,N)}{\partial N}\Big|_{N=1} = \sum_{d=d_f}^{\infty} W(d)D^d \tag{10}$$

where d_f is the free distance of the code, $a(d)$ is the number of paths at Hamming distance d from the "null" path and $W(d)$ is the total Hamming weight of input sequences $\{d_k\}$ used to generate all paths at distance d from the "null" path. In general, it is not easy to calculate the transfer function of a punctured code, that is why the first coefficients $a(d)$ and $W(d)$ are directly obtained from the trellis by using an algorithm derived from [7]. From coefficients $W(d)$, a tight upper bound of error probability can be calculated for large SNR's [6]

$$P_e \leq \sum_{d=d_f}^{\infty} W(d)P(d). \tag{11}$$

For a memoryless Gaussian channel with binary modulation (PSK, QPSK), probability $P(d)$ is equal to

$$P(d) = \frac{1}{2}\operatorname{erfc}\left[\sqrt{dR\frac{E_b}{N_0}}\right] \tag{12}$$

where E_b/N_0 is the energy per information bit to noise power spectral density ratio and R is the code rate.

In [8], a large number of RSC codes have been investigated and their performance was compared to that of NSC codes, in term of weight spectrum and of BER. Coefficients $a(d)$ are the same for RSC and NSC codes but the coefficients $\{W_{\mathrm{RSC}}(d)\}$ of RSC codes have a tendency to increase more slowly in function of d than the coefficients $\{W_{\mathrm{NSC}}(d)\}$ of NSC codes, whatever the rate R and whatever the memory ν. Thus, at low SNR's, the BER of the RSC code is always smaller than the BER of the equivalent NSC code.

In general, for rates $R \leq 2/3$, the first coefficients $(W_{\mathrm{RSC}}(d_f)), W_{\mathrm{RSC}}(d_f+1))$ are larger than those of NSC codes, therefore, at large SNR's, the performance of NSC codes is a little better than that of RSC codes. When the rate is larger than 2/3, it is easy to find RSC codes whose performance is better than that of NSC codes at any SNR.

In order to illustrate the performance of RSC codes, the BER of RSC and NSC codes are plotted in Fig. 3 for different values of R and for an encoder with memory $\nu = 6$ and generators 133, 171. For instance, at $P_e = 10^{-1}$, the coding gain with this RSC code, relative to the equivalent NSC code,

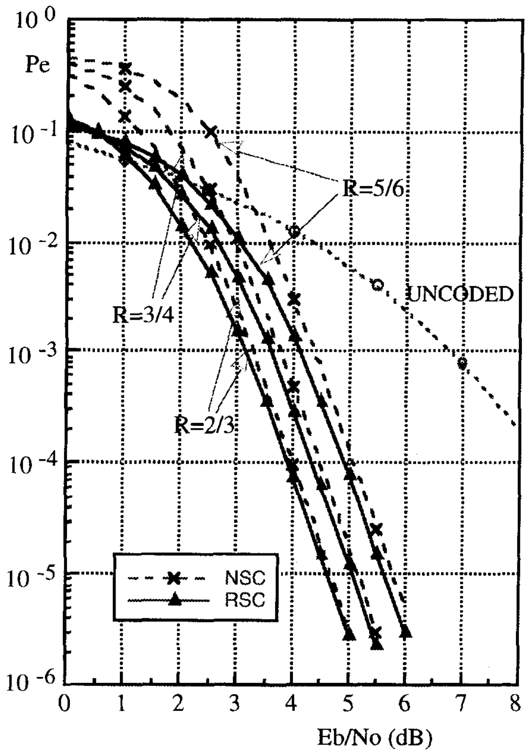


Fig. 3. P_e of punctured RSC and NSC codes for different values of rate R and memory $\nu = 6$, generators $G_1 = 133, G_2 = 171$.

is approximately 0.8 dB at $R = 2/3$, whereas at $R = 3/4$, it reaches 1.75 dB.

III. Parallel Concatenation with Non Uniform Interleaving: Turbo-Code

A. Construction of the Code

The use of systematic codes enables the construction of a concatenated encoder in the form given in Fig. 4, called *parallel concatenation.* The data flow (d_k at time k) goes directly to a first elementary RSC encoder C_1 and after interleaving, it feeds (d_n at time k) a second elementary RSC encoder C_2. These two encoders are not necessarily identical. Data d_k is systematically transmitted as symbol X_k and redundancies Y_{1k} and Y_{2k} produced by C_1 and C_2 may be completely transmitted for an $R = 1/3$ encoding or punctured for higher rates. The two elementary coding rates R_1 and R_2 associated with C_1 and C_2, after puncturing, may be different, but for the best decoding performance, they will satisfy$R_1 \leq R_2$. The global rate R of the composite code, R_1 and R_2 are linked by

$$\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2} - 1. \tag{13}$$

Unlike the classical (serial) concatenation, parallel concatenation enables the elementary encoders, and therefore the associated elementary decoders, to run with the same clock. This point constitutes an important simplification for the design of the associated circuits, in a concatenated scheme.

Authorized licensed use limited to: Yonsei University. Downloaded on December 26, 2009 at 22:55 from IEEE Xplore. Restrictions apply.

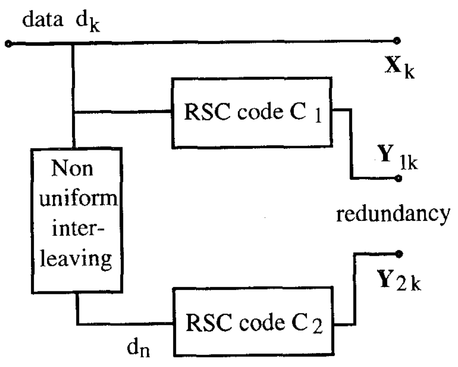


Fig. 4. Basic turbo-encoder (rate 1/3).

B. Distance Properties

Consider for instance elementary codes C_1 and C_2 with memory $\nu = 4$ and encoder polynomials $G_1 = 23, G_2 = 35$. Redundancy Y_k is one time every second time, either Y_{1k} or Y_{2k}. Then the global rate is $R = 1/2$ with $R_1 = R_2 = 2/3$. The code being linear, the distance properties will be considered relatively to the "all zero" or "null" sequence. Both encoders C_1, C_2 and the interleaver are initialized to the "all zero" state and a sequence $\{d_k\}$ containing w "1"s, is fed to the turbo-encoder. w is called the *input weight.*

Some definitions: let us call a *Finite Codeword* (FC) of an elementary RSC encoder an output sequence with a finite distance from the "all zero" sequence (i.e., a limited number of "1"s in output sequences $\{X_k\}$ and $\{Y_k\}$). Because of recursivity, only some input sequences, fitting with the linear feedback register (LFR) generation of Y_k, give FC's. These particular input sequences are named *FC (input) patterns.* Let us also call a *global FC* an output sequence of the turbo-code, with a finite distance from the "all zero" sequence (i.e., a limited number of "1"s in output sequences $\{X_k\}, \{Y_{1k}\}$ and $\{Y_{2k}\}$). The distance $d_q(w)$ of an elementary FC ($q = 1$ for $C_1, q = 2$ for C_2), associated with an input sequence $\{d_k\}$ with weight w, is the sum of the two contributions of $\{X_k\}$ and $\{Y_{qk}\}$

$$d_q(w) = d_{X_q}(w) + d_{Y_q}(w). \tag{14}$$

Since the codes are systematic, $d_{X_q}(w) = w$

$$d_q(w) = w + d_{Y_q}(w). \tag{15}$$

The distance $d(w)$ of a global FC is given likewise by

$$d(w) = w + d_{Y1}(w) + d_{Y2}(w). \tag{16}$$

1) Uniform Interleaving: Consider a uniform block interleaving using an $M \cdot M$ square matrix with M large enough (i.e., ≥ 32), and generally a power of 2. Data are written linewise and read columnwise. As explained above, the matrix is filled with "0"s except for some "1"s and we are now going to state some of the possible patterns of "1"s leading to global FC's and evaluate their associated distances. Beforehand, note that, for each elementary code, the minimal value for input weight w is two, because of their recursive structure. For the particular case of $w = 2$, the delay between the two data at "1" is 15 or a multiple of 15, since the LFR associated with the redundancy generation, with polynomial $G = 23$, is a maximum length LFR. If the delay is not a multiple of 15, then the associated sequence $\{Y_k\}$ will contain "0"s and "1"s indefinitely.

Global FC's with Input Weight $w = 2$

Global FC's with $w = 2$ have to be FC's with input weight $w = 2$ for each of the constituent codes. Then the two data at "1" in the interleaving memory must be located at places which are distant by a multiple of 15, when both writing and reading. For lack of an extensive analysis of the possible patterns which satisfy this double constraint, let us consider only the case where the two "1"s in $\{d_k\}$ are time-separated by the lowest value (i. e. 15). It is also a FC pattern for code C_2 if the two data at "1" are memorized on a single line, when writing in the interleaving memory, and thus the span between the two "1"s is $15 \cdot M$, when reading. From (16), the distance associated with this input pattern is approximately equal to

$$\begin{aligned} d(2) &\approx 2 + \min\{d_{Y1}(2)\} + \mathrm{INT}((15 \cdot M + 1)/4) \\ &= 2 + 4 + \mathrm{INT}((15 \cdot M + 1)/4) \end{aligned} \tag{17}$$

where INT$(\cdot)$ stands for integer part of $(\cdot)$. The first term represents input weight w, the second term the distance added by redundancy Y_1 of C_1 (rate 2/3), the third term the distance added by redundancy Y_2. The latter is given assuming that, for this second 2/3 rate code, the $(15 \cdot M + 1)/2$ Y_2 symbols are at "1", one out of two times statistically, which explains the additional division by 2. With $M = 64$ for instance, the distance is $d(2) \approx 246$. This value is for a particular case of a pattern of two "1"s but we imagine it is a realistic example for all FC's with input weight 2. If the two "1"s are not located on a single line when writing in the interleaving matrix, and if M is larger than 30, the span between the two "1"s, when reading from the interleaving matrix, is higher than $15 \cdot M$ and the last term in (17) is increased.

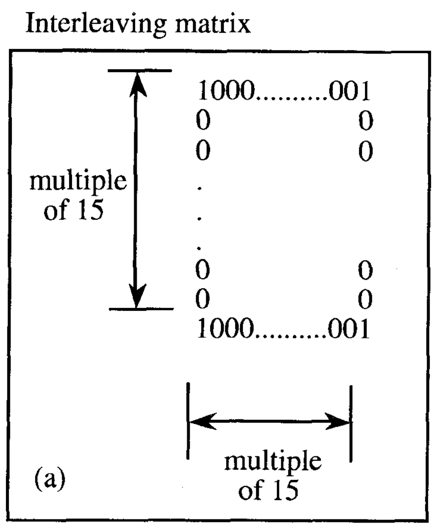
Global FC's with $w = 3$

Global FC's with input weight $w = 3$ have to be elementary FC's with input weight $w = 3$ for each of the constituent codes. The inventory of the patterns with three "1"s which satisfy this double constraint is not easy to make. It is not easier to give the slightest example. Nevertheless, we can consider that associated distances are similar to the case of input weight two codewords, because the FC for C_2 is still several times M long.

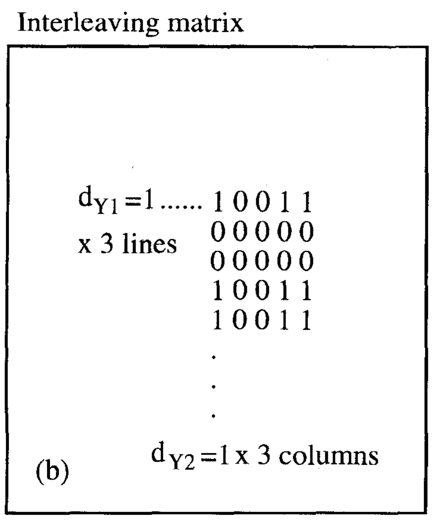
Global FC's with $w = 4$

Here is the first pattern of a global FC which can be viewed as the separate combination of two minimal (input weight $w = 2$) elementary FC patterns both for C_1 and C_2. When writing in the interleaving memory, the four data at "1" are located at the four corners of a square or a rectangle, the sides of which have lengths equal to 15 or a multiple of 15 [Fig. 5(a)]. The minimum distance is given by a square pattern, with side length equal to 15, corresponding to minimum values

Authorized licensed use limited to: Yonsei University. Downloaded on December 26, 2009 at 22:55 from IEEE Xplore. Restrictions apply.



(a)



(b)

Fig. 5. (a) Input pattern for separable global FC's with input weight $w = 4$. (b) Input pattern for a global FC with $w = 9$ and total distance of 15.

of $d_{Y1}(2)$ and $d_{Y2}(2)$

$$\begin{aligned} d(4) &= 4 + 2.\min\{d_{Y1}(2)\} + 2.\min\{d_{Y2}(2)\} \\ &= 4 + 8 + 8 = 20. \end{aligned} \tag{18}$$

Global FC's with $w > 4$

As previously, one has to distinguish between cases where the global FC is separable into two, or more, elementary FC's for both codes, or not. The shortest distances are given by separable codewords (this occurs for weights 6, 8, 9, 10, $12, \cdots$). We made an exhaustive research of possible patterns up to $w = 10$ and found a minimum distance of 15 for the pattern given in Fig. 5(b), with $w = 9$, which corresponds to three separable FC's for each of the two codes, and the six FC's having d_{Y1} or d_{Y2} equal to 1.

To conclude this review of the global FC's with the lowest values of we can retain the result obtained for the case in Fig. 5(b) as the minimum distance d_m of this particular turbo-code ($\nu = 4$, polynomials 23, 35, $R_1 = R_2 = 2/3$). Other codes were considered with different values of ν and various polynomials; in all cases, the minimum distance seems to correspond to input sequences with weights 4, 6, or 9. The values of d_m are within 10 and 20, which is not a remarkable property. So as to obtain larger values of d_m, turbo-coding employs nonuniform interleaving.

2) Nonuniform interleaving: It is obvious that patterns giving the shortest distances, such as those represented in Fig. 5, can be "broken" by appropriate nonuniform interleaving, in order to transform a separable FC pattern into either a nonseparable or a non FC. Nonuniform interleaving must satisfy two main conditions: the maximum scattering of data, as in usual interleaving, and the maximum disorder in the interleaved data sequence. The latter, which may be in conflict with the former, is to make redundancy generation by the two encoders as diverse as possible. In this case, if the decision by the decoder associated with C_1 about particular data implies a few items of redundancy Y_1, then the corresponding decision by the decoder associated with C_2 will rely on a large number of values Y_2, and vice-versa. Then, the minimum distance of the turbo-code may be increased to much larger values than that given by uniform interleaving.

The mathematical aspects of nonuniform interleaving are not trivial and have still to be studied. For instance, how can it be ensured that an interleaver, able to "break" patterns corresponding to $w = 4, 6$, or 9, will not drastically lower the distance for a particular case of a $w = 2$ input sequence? On the other hand, the interleaving equations have to be limited in complexity for silicon applications, because several interleavers and de-interleavers have to be employed in a real-time turbo-decoder (see Section VI-A). However that may be, as we have tried to explain up to now, one important property of the turbo-code is that its minimum distance d_m is not fixed, chiefly, by the constituent RSC codes but by the interleaving function; and finding out the optimum interleaver for turbo-codes remains a real challenge.

For the results which are presented in Section VII, we used an empirical approach and chose an interleaving procedure in which, for reading, the column index is a function of the line index. Let i and j be the addresses of line and column for writing, and i_r and j_r the addresses of line and column for reading. For an $M \cdot M$ memory (M being a power of two), i, j, i_r and j_r have values between 0 and $M - 1$. Nonuniform interleaving may be described by

$$\begin{aligned} i_r &= (M/2 + 1)(i + j) && \bmod \cdot M \\ \xi &= (i + j) && \bmod \cdot 8 \\ j_r &= [P(\xi) \cdot (j + 1)] - 1 && \bmod \cdot M \end{aligned} \tag{19}$$

$P(\cdot)$ is a number, relatively prime with M, which is a function of line address $(i + j)$ mod $\cdot 8$. Note that reading is performed diagonally in order to avoid possible effects of a relation between M and the period of puncturing. A multiplying factor $(M/2 + 1)$ is used to prevent two neighboring data written on two consecutive lines from remaining neighbors in reading, in a similar way as given in [9].

Authorized licensed use limited to: Yonsei University. Downloaded on December 26, 2009 at 22:55 from IEEE Xplore. Restrictions apply.

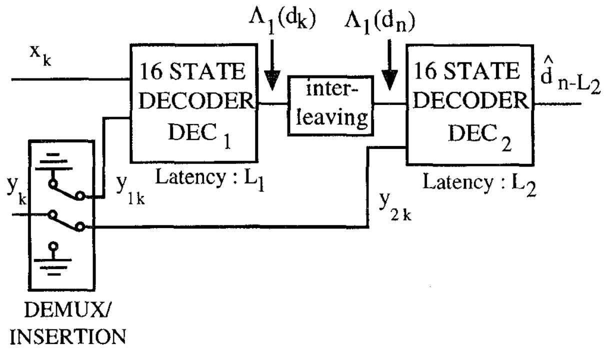


Fig. 6. Principle of the decoder in accordance with a serial concatenation scheme.

IV. Decoding Turbo-Codes

The decoder depicted in Fig. 6, is made up of two elementary decoders (DEC_1 and DEC_2) in a serial concatenation scheme. The first elementary decoder DEC_1 is associated with the lower rate R_1 encoder C_1 and yields a weighted decision.

For a discrete memoryless Gaussian channel and a binary modulation, the decoder input is made up of a couple R_k of two random variables x_k and y_k, at time k

$$\begin{aligned} x_k &= (2X_k - 1) + i_k \\ y_k &= (2Y_k - 1) + q_k \end{aligned} \tag{20}$$

where i_k and q_k are two independent noises with the same variance σ^2. The redundant information y_k is demultiplexed and sent to decoder DEC_1 when $Y_k = Y_{1k}$ and toward decoder DEC_2 when $Y_k = Y_{2k}$. When the redundant information of a given encoder (C_1 or C_2) is not emitted, the corresponding decoder input is set to analog zero. This is performed by the DEMUX/INSERTION block.

The logarithm of likelihood ratio (LLR), $\Lambda_1(d_k)$ associated with each decoded bit d_k by the first decoder DEC_1 can be used as a relevant piece of information for the second decoder DEC_2

$$\Lambda_1(d_k) = \log \frac{\Pr\{d_k = 1/\text{observation}\}}{\Pr\{d_k = 0/\text{observation}\}} \tag{21}$$

where $\Pr\{d_k = i/\text{observation}\}, i = 0, 1$ is the *a posteriori* probability (APP) of the bit d_k.

A. Optimal Decoding of RSC Codes with Weighted Decision

The VITERBI algorithm is an optimal decoding method which minimizes the probability of sequence error for convolutional codes. Unfortunately, this algorithm is not able to yield directly the APP for each decoded bit. A relevant algorithm for this purpose has been proposed by BAHL *et al.* [10]. This algorithm minimizes the bit error probability in decoding linear block and convolutional codes and yields the APP for each decoded bit. For RSC codes, the BAHL *et al.* algorithm must be modified in order to take into account their recursive character.

Modified BAHL et al. Algorithm for RSC Codes: Consider an RSC code with constraint length K; at time k the encoder state $\boldsymbol{S}_k$ is represented by a K-uple

$$\boldsymbol{S}_k = (a_k, a_{k-1} \cdots\cdots a_{k-K+2}). \tag{22}$$

Let us also suppose that the information bit sequence $\{d_k\}$ is made up of N independent bits d_k, taking values zero and one with equal probability and that the encoder initial state $\boldsymbol{S}_0$ and final state $\boldsymbol{S}_N$ are both equal to zero, *i.e*

$$\boldsymbol{S}_0 = \boldsymbol{S}_N = (0, 0 \cdots\cdots 0) = \boldsymbol{0}. \tag{23}$$

The encoder output codeword sequence, noted $\boldsymbol{C}_1^N = \{C_1 \cdots\cdots C_k \cdots\cdots C_N\}$ where $C_k = (X_k, Y_k)$ is the input to a discrete Gaussian memoryless channel whose output is the sequence $\boldsymbol{R}_1^N = \{R_1 \cdots\cdots R_k \cdots\cdots R_N\}$ where $R_k = (x_k, y_k)$ is defined by (20).

The APP of a decoded bit d_k can be derived from the joint probability $\lambda_k^i(\boldsymbol{m})$ defined by

$$\lambda_k^i(\boldsymbol{m}) = \Pr\{d_k = i, \boldsymbol{S}_k = \boldsymbol{m}/\boldsymbol{R}_1^N\} \tag{24}$$

and thus, the APP of a decoded bit d_k is equal to

$$\Pr\{d_k = i/\boldsymbol{R}_1^N\} = \sum_{\boldsymbol{m}} \lambda_k^i(\boldsymbol{m}), \qquad i = 0, 1. \tag{25}$$

From (21) and (24), the LLR $\Lambda(d_k)$ associated with a decoded bit d_k can be written as

$$\Lambda(d_k) = \log \frac{\sum_{\boldsymbol{m}} \lambda_k^1(\boldsymbol{m})}{\sum_{\boldsymbol{m}} \lambda_k^0(\boldsymbol{m})}. \tag{26}$$

Finally the decoder can make a decision by comparing $\Lambda(d_k)$ to a threshold equal to zero

$$\begin{aligned} \hat{d}_k &= 1 \quad \text{if} \quad \Lambda(d_k) \geq 0 \\ \hat{d}_k &= 0 \quad \text{if} \quad \Lambda(d_k) < 0. \end{aligned} \tag{27}$$

From the definition (24) of $\lambda_k^i(\boldsymbol{m})$, the LLR $\Lambda(d_k)$ can be written as shown at the bottom of the page.

Using the BAYES rule and taking into account that events after time k are not influenced by observation $\boldsymbol{R}_1^k$ and bit d_k

$$\Lambda(d_k) = \log \frac{\sum_{\boldsymbol{m}} \sum_{\boldsymbol{m}'} \Pr\{d_k = 1, \boldsymbol{S}_k = \boldsymbol{m}, \boldsymbol{S}_{k-1} = \boldsymbol{m}', \boldsymbol{R}_1^{k-1}, R_k, R_{k+1}^N\}}{\sum_{\boldsymbol{m}} \sum_{\boldsymbol{m}'} \Pr\{d_k = 0, \boldsymbol{S}_k = \boldsymbol{m}, \boldsymbol{S}_{k-1} = \boldsymbol{m}', \boldsymbol{R}_1^{k-1}, R_k, R_{k+1}^N\}} \tag{28}$$

Authorized licensed use limited to: Yonsei University. Downloaded on December 26, 2009 at 22:55 from IEEE Xplore. Restrictions apply.

if state $\boldsymbol{S}_k$ is known, the LLR $\Lambda(d_k)$ is equal

$$\Lambda(d_k) = \log \frac{\sum_{\boldsymbol{m}}\sum_{\boldsymbol{m}'} \Pr\{\boldsymbol{R}_{k+1}^N/\boldsymbol{S}_k = \boldsymbol{m}\} \Pr\{\boldsymbol{S}_{k-1} = \boldsymbol{m}'/\boldsymbol{R}_1^{k-1}\}}{\sum_{\boldsymbol{m}}\sum_{\boldsymbol{m}'} \Pr\{\boldsymbol{R}_{k+1}^N/\boldsymbol{S}_k = \boldsymbol{m}\} \Pr\{\boldsymbol{S}_{k-1} = \boldsymbol{m}'/\boldsymbol{R}_1^{k-1}\}} \cdot \frac{\Pr\{d_k = 1, \boldsymbol{S}_k = \boldsymbol{m}, R_k/\boldsymbol{S}_{k-1} = \boldsymbol{m}'\}}{\Pr\{d_k = 0, \boldsymbol{S}_k = \boldsymbol{m}, R_k/\boldsymbol{S}_{k-1} = \boldsymbol{m}'\}}. \tag{29}$$

In order to compute the LLR $\Lambda(d_k)$, as in [11] let us introduce the probability functions $\alpha_k(\boldsymbol{m}), \beta_k(\boldsymbol{m})$ and $\gamma_i(\boldsymbol{R}_k, \boldsymbol{m}', \boldsymbol{m})$ defined by

$$\alpha_k(\boldsymbol{m}) = \Pr\{\boldsymbol{S}_k = \boldsymbol{m}/\boldsymbol{R}_1^k\} \tag{30a}$$

$$\beta_k(\boldsymbol{m}) = \frac{\Pr\{\boldsymbol{R}_{k+1}^N/\boldsymbol{S}_k = \boldsymbol{m}\}}{\Pr\{\boldsymbol{R}_{k+1}^N/\boldsymbol{R}_1^k\}} \tag{30b}$$

$$\gamma_i(R_k, \boldsymbol{m}', \boldsymbol{m}) = \Pr\{d_k = i, \boldsymbol{S}_k = \boldsymbol{m}, R_k/\boldsymbol{S}_{k-1} = \boldsymbol{m}'\}. \tag{30c}$$

Using the LLR definition (29) and (30a), (30b) and (30c), $\Lambda(d_k)$ is equal to

$$\Lambda(d_k) = \log \frac{\sum_{\boldsymbol{m}}\sum_{\boldsymbol{m}'} \gamma_1(R_k, \boldsymbol{m}', \boldsymbol{m})\alpha_{k-1}(\boldsymbol{m}')\beta_k(\boldsymbol{m})}{\sum_{\boldsymbol{m}}\sum_{\boldsymbol{m}'} \gamma_0(R_k, \boldsymbol{m}', \boldsymbol{m})\alpha_{k-1}(\boldsymbol{m}')\beta_k(\boldsymbol{m})} \tag{31}$$

where probabilities $\alpha_k(\boldsymbol{m})$ and $\beta_k(\boldsymbol{m})$ can be recursively calculated from probability $\gamma_i(R_k, \boldsymbol{m}', \boldsymbol{m})$ as in [12]

$$\alpha_k(\boldsymbol{m}) = \frac{\sum_{\boldsymbol{m}'}\sum_{i=0}^{1} \gamma_i(R_k, \boldsymbol{m}', \boldsymbol{m})\alpha_{k-1}(\boldsymbol{m}')}{\sum_{\boldsymbol{m}}\sum_{\boldsymbol{m}'}\sum_{i=0}^{1} \gamma_i(R_k, \boldsymbol{m}', \boldsymbol{m})\alpha_{k-1}(\boldsymbol{m}')} \tag{32}$$

$$\beta_k(\boldsymbol{m}) = \frac{\sum_{\boldsymbol{m}'}\sum_{i=0}^{1} \gamma_i(R_{k+1}, \boldsymbol{m}', \boldsymbol{m})\beta_{k+1}(\boldsymbol{m}')}{\sum_{\boldsymbol{m}}\sum_{\boldsymbol{m}'}\sum_{i=0}^{1} \gamma_i(R_{k+1}, \boldsymbol{m}, \boldsymbol{m}')\alpha_k(\boldsymbol{m}')}. \tag{33}$$

The probability $\gamma_i(R_k, \boldsymbol{m}', \boldsymbol{m})$ can be determined from transition probabilities of the discrete Gaussian memoryless channel and transition probabilities of the encoder trellis. From (30c), $\gamma_i(R_k, \boldsymbol{m}', \boldsymbol{m})$ is given by

$$\begin{aligned}\gamma_i(R_k, \boldsymbol{m}', \boldsymbol{m}) &= p(R_k/d_k = i, \boldsymbol{S}_k = \boldsymbol{m}, \boldsymbol{S}_{k-1} = \boldsymbol{m}') \\ &\cdot q(d_k = i/\boldsymbol{S}_k = \boldsymbol{m}, \boldsymbol{S}_{k-1} = \boldsymbol{m}') \\ &\cdot \pi(\boldsymbol{S}_k = \boldsymbol{m}/\boldsymbol{S}_{k-1} = \boldsymbol{m}')\end{aligned} \tag{34}$$

where $p(\cdot/\cdot)$ is the transition probability of the discrete Gaussian memoryless channel. Conditionally to $(d_k = i, \boldsymbol{S}_k = \boldsymbol{m}, \boldsymbol{S}_{k-1} = \boldsymbol{m}'), x_k$ and y_k $(R_k = (x_k, y_k))$ are two uncorrelated Gaussian variables and thus we obtain

$$\begin{aligned}p(R_k/d_k = i, \boldsymbol{S}_k = \boldsymbol{m}, \boldsymbol{S}_{k-1} = \boldsymbol{m}') &= p(x_k/d_k = i, \boldsymbol{S}_k = \boldsymbol{m}, \boldsymbol{S}_{k-1} = \boldsymbol{m}') \\ &\cdot p(y_k/d_k = i, \boldsymbol{S}_k = \boldsymbol{m}, \boldsymbol{S}_{k-1} = \boldsymbol{m}').\end{aligned} \tag{35}$$

Since the convolutional encoder is a deterministic machine, $q(d_k = i/\boldsymbol{S}_k = \boldsymbol{m}, \boldsymbol{S}_{k-1} = \boldsymbol{m}')$ is equal to 0 or 1. The transition state probabilities $\pi(\boldsymbol{S}_k = \boldsymbol{m}/\boldsymbol{S}_{k-1} = \boldsymbol{m}')$ of the trellis are defined by the encoder input statistic. Generally, $\Pr\{d_k = 1\} = \Pr\{d_k = 0\} = 1/2$ and since there are two possible transitions from each state, $\pi(\boldsymbol{S}_k = \boldsymbol{m}/\boldsymbol{S}_{k-1} = \boldsymbol{m}') = 1/2$ for each of these transitions.

Modified BAHL et al. Algorithm:

Step 0: Probabilities $\alpha_0(\boldsymbol{m})$ are initialized according to condition (23)

$$\alpha_0(0) = 1; \quad \alpha_0(\boldsymbol{m}) = 0 \quad \forall \boldsymbol{m} \neq \boldsymbol{0}. \tag{36a}$$

Concerning the probabilities $\beta_N(\boldsymbol{m})$, the following conditions were used

$$\beta_N(\boldsymbol{m}) = \frac{1}{N} \forall \boldsymbol{m} \tag{36b}$$

since it is very difficult to put the turbo encoder at state 0, at a given time. Obviously with this condition, the last bits of each block are not taken into account for evaluating the BER.

Step 1: For each observation R_k, probabilities $\alpha_k(\boldsymbol{m})$ and $\gamma_i(R_k, \boldsymbol{m}', \boldsymbol{m})$ are computed using (32) and (34), respectively.

Step 2: When sequence $\boldsymbol{R}_1^N$ has been completely received, probabilities $\beta_k(\boldsymbol{m})$ are computed using (33), and the LLR associated with each decoded bit d_k is computed from (31).

V. Extrinsic Information from the Decoder

In this chapter, we will show that the LLR $\Lambda(d_k)$ associated with each decoded bit d_k, is the sum of the LLR's of d_k at the decoder input and of other information called *extrinsic information,* generated by the decoder.

Since the encoder is systematic $(X_k = d_k)$, the transition probability $p(x_k/d_k = i, \boldsymbol{S}_k = \boldsymbol{m}, \boldsymbol{S}_{k-1} = \boldsymbol{m}')$ in expression $\gamma_i(R_k, \boldsymbol{m}', \boldsymbol{m})$ is independent of state values $\boldsymbol{S}_k$ and $\boldsymbol{S}_{k-1}$. Therefore we can factorize this transition probability in the numerator and in the denominator of (31)

$$\Lambda(d_k) = \log \frac{p(x_k/d_k = 1)}{p(x_k/d_k = 0)} + \log \frac{\sum_{\boldsymbol{m}}\sum_{\boldsymbol{m}'} \gamma_1(y_k, \boldsymbol{m}', \boldsymbol{m})\alpha_{k-1}(\boldsymbol{m}')\beta_k(\boldsymbol{m})}{\sum_{\boldsymbol{m}}\sum_{\boldsymbol{m}'} \gamma_0(y_k, \boldsymbol{m}', \boldsymbol{m})\alpha_{k-1}(\boldsymbol{m}')\beta_k(\boldsymbol{m})}. \tag{37}$$

Conditionally to $d_k = 1$ (resp. $d_k = 0$), variables x_k are Gaussian with mean 1 (resp. -1) and variance σ^2, thus the LLR $\Lambda(d_k)$ is still equal to

$$\Lambda(d_k)\frac{2}{\sigma^2}x_k + W_k \tag{38}$$

Authorized licensed use limited to: Yonsei University. Downloaded on December 26, 2009 at 22:55 from IEEE Xplore. Restrictions apply.

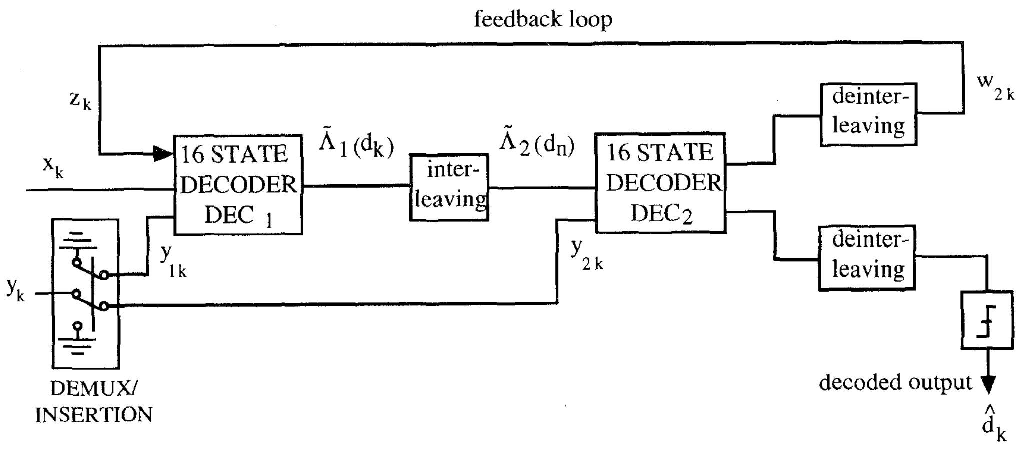


Fig. 7. Feedback decoder (under 0 internal delay assumption).

where

$$W_k = \Lambda(d_k)\,|_{x_k=0} = \log \frac{\sum\limits_{m}\sum\limits_{m'} \gamma_1(y_k, \boldsymbol{m}', \boldsymbol{m})\alpha_{k-1}(\boldsymbol{m}')\beta_k(\boldsymbol{m})}{\sum\limits_{m}\sum\limits_{m'} \gamma_0(y_k, \boldsymbol{m}', \boldsymbol{m})\alpha_{k-1}(\boldsymbol{m}')\beta_k(\boldsymbol{m})}. \tag{39}$$

W_k is a function of the redundant information introduced by the encoder. In general, W_k has the same sign as d_k; therefore W_k may improve the LLR associated with each decoded bit d_k. This quantity represents the extrinsic information supplied by the decoder and does not depend on decoder input x_k. This property will be used for decoding the two parallel concatenated encoders.

VI. Decoding Scheme of Parallel Concatenation Codes

In the decoding scheme represented in Fig. 6, decoder DEC_1 computes LLR $\Lambda_1(d_k)$ for each transmitted bit d_k from sequences $\{x_k\}$ and $\{y_k\}$, then decoder DEC_2 performs the decoding of sequence $\{d_k\}$ from sequences $\{\Lambda_1(d_k)\}$ and $\{y_k\}$. Decoder DEC_1 uses the modified BAHL *et al.* algorithm and decoder DEC_2 may use the VITERBI algorithm. The global decoding rule is not optimal because the first decoder uses only a fraction of the available redundant information. Therefore it is possible to improve the performance of this serial decoder by using a feedback loop.

A. Decoding with a Feedback Loop

Both decoders DEC_1 and DEC_2 now use the modified BAHL *et al.* algorithm. We have seen in section V that the LLR at the decoder output can be expressed as a sum of two terms if the noises at the decoder inputs are independent at each time k. Hence, if the noise at the decoder DEC_2 inputs are independent, the LLR $\Lambda_2(d_k)$ at the decoder DEC_2 output can be written as

$$\Lambda_2(d_k) = f(\Lambda_1(d_k)) + W_{2k} \tag{40}$$

with

$$\Lambda_1(d_k) = \frac{2}{\sigma^2} x_k + W_{1k}. \tag{41}$$

From (39), we can see that the decoder DEC_2 extrinsic information W_{2k} is a function of the sequence $\{\Lambda_1(d_n)\}_{n\neq k}$. Since $\Lambda_1(d_n)$ depends on observation $\boldsymbol{R}_1^N$, extrinsic information W_{2k} is correlated with observations x_k and y_{1k}, regarding the noise. Nevertheless, from (39), the greater $|n-k|$ is, the less correlated are $\Lambda_1(d_n)$ and observations x_k, y_k. Thus, due to the presence of interleaving between decoders DEC_1 and DEC_2, extrinsic information W_{2k} and observations x_k, y_{1k} are weakly correlated. Therefore extrinsic information W_{2k} and observations x_k, y_{1k} can be used jointly for carrying out a new decoding of bit d_k, the extrinsic information $z_k = W_{2k}$ acting as a diversity effect.

In Fig. 7, we have depicted a new decoding scheme using the extrinsic information W_{2k} generated by decoder DEC_2 in a feedback loop. For simplicity this drawing does not take into account the different delays introduced by decoder DEC_1 and DEC_2, and interleaving.

The first decoder DEC_1 now has three data inputs: (x_k, y_k^1, z_k), and probabilities $\alpha_{1k}(\boldsymbol{m})$ and $\beta_{1k}(\boldsymbol{m})$ are computed by substituting $R_k = (x_k, y_{1k}, z_k)$ for $R_k = (x_k, y_{1k})$ in (32) and (33). Taking into account that z_k is weakly correlated with x_k and y_{1k} and supposing that z_k can be approximated by a Gaussian variable with variance $\sigma_z^2 \neq \sigma^2$, the transition probability of the discrete gaussian memoryless channel can be now factorized in three terms

$$\begin{aligned} p(R_k/d_k = i, \boldsymbol{S}_k = \boldsymbol{m}, \boldsymbol{S}_{k-1} = \boldsymbol{m}') \\ = p(x_k/\cdot)p(y_k^1/\cdot)p(z_k/\cdot). \end{aligned} \tag{42}$$

Encoder C_1 with initial rate R_1, through the feedback loop, is now equivalent to a rate R_1' encoder with

$$R_1' = \frac{R_1}{1+R_1}. \tag{43}$$

The first decoder obtains additional redundant information with Z_k that may significantly improve its performance ; the term turbo-code is given for this feedback decoder scheme with reference to the principle of the turbo engine.

Authorized licensed use limited to: Yonsei University. Downloaded on December 26, 2009 at 22:55 from IEEE Xplore. Restrictions apply.

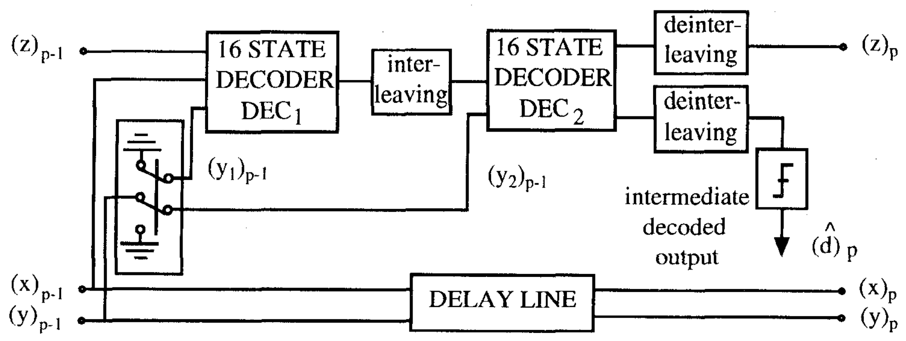


Fig. 8. Decoding module (level p).

With the feedback decoder, the LLR $\Lambda_1(d_k)$ generated by decoder DEC_1 is now equal to

$$\Lambda_1(d_k) = \frac{2}{\sigma^2} x_k + \frac{2}{\sigma_z^2} z_k + W_{1k} \tag{44}$$

where W_{1k} depends on sequence $\{z_n\}_{n \neq k}$. As indicated above, information z_k has been built by decoder DEC_2. Therefore z_k must not be used as input information for decoder DEC_2. Thus decoder DEC_2 input sequences will be sequences $\{\tilde{\Lambda}_1(d_n)\}$ and $\{y_{2k}\}$ with

$$\tilde{\Lambda}_1(d_n) = \Lambda_1(d_n)_{z_n=0}. \tag{45}$$

Finally, from (40), decoder DEC_2 extrinsic information $z_k = W_{2k}$ after deinterleaving can be written as

$$z_k = W_{2k} = \Lambda_2(d_k)|_{\tilde{\Lambda}_1(d_k)=0} \tag{46}$$

and the decision at the decoder DEC output is

$$\hat{d}_k = \mathrm{sign}[\Lambda_2(d_k)]. \tag{47}$$

The decoding delays introduced by the component decoders, the interleaver and the deinterleaver imply that the feedback piece of information z_k must be used through an iterative process.

The global decoder circuit is made up of P pipelined identical elementary decoders. The pth decoder DEC (Fig. 8) input, is made up of demodulator output sequences $(x)_p$ and $(y)_p$ through a delay line and of extrinsic information $(z)_p$ generated by the $(p-1)$th decoder DEC. Note that the variance σ_z^2 of $(z)_p$ and the variance of $\tilde{\Lambda}_1(d_k)$ must be estimated at each decoding step p.

For example, the variance σ_z^2 is estimated for each M^2 interleaving matrix by the following:

$$\sigma_z^2 = \frac{1}{M^2} \sum_{k=1}^{M^2} (|z_k| - m_z)^2 \tag{48a}$$

where m_z is equal to

$$m_z = \frac{1}{M^2} \sum_{k=1}^{M^2} |z_k|. \tag{48b}$$

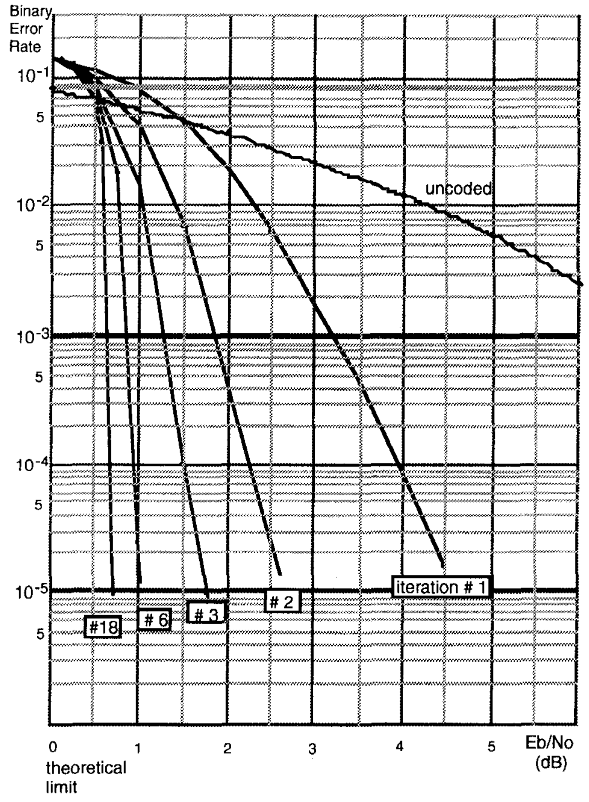


Fig. 9. BER given by iterative decoding ($p = 1, \cdots 18$) of a rate $R = 1/2$ encoder, memory $\nu = 4$, generators $G_1 = 37, G_2 = 21$, with interleaving 256×256.

B. Interleaving

The interleaver is made up of an $M \cdot M$ matrix and bits $\{d_k\}$ are written row by row and read following the nonuniform rule given in Section III-B2. This nonuniform reading procedure is able to spread the residual error blocks of rectangular form, that may set up in the interleaver located behind the first decoder DEC_1, and to give a large free distance to the concatenated (parallel) code.

For the simulations, a 256.256 matrix has ben used and from (19), the addresses of line i_r and column j_r for reading are the following:

$$i_r = 129(i + j) \qquad \mathrm{mod} \cdot 256$$

Authorized licensed use limited to: Yonsei University. Downloaded on December 26, 2009 at 22:55 from IEEE Xplore. Restrictions apply.

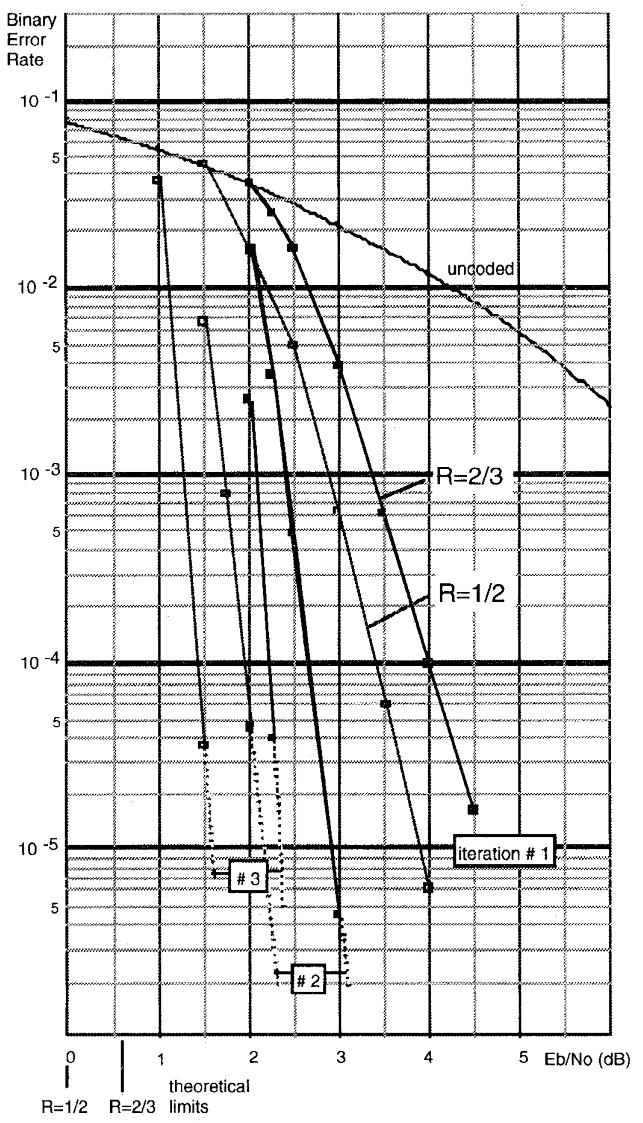


Fig. 10. BER given by iterative decoding ($p = 1, 2, 3$) of code with memory $\nu = 4$, generators $G_1 = 23, G_2 = 35$, rate $R = 1/2$ and $R = 2/3$ and interleaving 256×256 decoding.

$$\begin{aligned} \xi &= (i + j) && \text{mod} \cdot 8 \\ j_r &= [P(\xi) \cdot (j + 1)] - 1 && \text{mod} \cdot 256 \end{aligned} \tag{49a}$$

with

$$\begin{aligned} &P(0) = 17; \quad P(1) = 37; \quad P(2) = 19; \quad P(3) = 29 \\ &P(4) = 41; \quad P(5) = 23; \quad P(6) = 13; \quad P(7) = 7. \end{aligned} \tag{49b}$$

VII. Results

For a rate $R = 1/2$ encoder with memory $\nu = 4$, and for generators $G_1 = 37, G_2 = 21$ and parallel concatenation 2/3//2/3 ($R_1 = 2/3, R_2 = 2/3$), the BER has been computed after each decoding step using the Monte Carlo method, as a function of signal to noise ratio E_b/N_0. The interleaver consists of a 256×256 matrix and the modified BAHL *et al.* algorithm has been used with a data block length of $N = 65\,536$ bits. In order to evaluate a BER equal to 10^{-5}, we have considered 256 data blocks *i.e.* approximately 16×10^6 b d_k. The BER versus E_b/N_0, for different values of p is plotted in Fig. 9. For any given SNR greater than 0 dB, the BER decreases as a function of the decoding step p. The coding gain is fairly high for the first values of $p (p = 1, 2, 3)$ and carries on increasing for the subsequent values of p. For $p = 18$ for instance, the BER is lower than 10^{-5} at $E_b/N_0 = 0.7$ dB with generators $G_1 = 37, G_2 = 21$. Remember that the Shannon limit for a binary modulation is $P_e = 0$ (several authors take $P_e = 10^{-5}$ as a reference) for $E_b/N_0 = 0$ dB. With the parallel concatenation of RSC convolutional codes and feedback decoding, the performance is 0.7 dB from Shannon's limit. The influence of the memory ν on the BER has also been examined. For memory greater than 4, at $E_b/N_0 = 0.7$ dB, the BER is slightly worse at the first ($p = 1$) decoding step and the feedback decoding is inefficient to improve the final BER. For ν smaller than 4, at $E_b/N_0 = 0.7$ dB, the BER is slightly better at the first decoding step than for ν equal to four, but the correction capacity of encoders C_1 and C_2 is too weak to improve the BER with feedback decoding. For $\nu = 3$ (i.e., with only eight-states decoders) and after iteration 18, a BER of 10^{-5} is achieved at $E_b/N_0 = 0.9$ dB.

For ν equal to 4, we have tested several generators (G_1, G_2) and the best results were achieved with $G_1 = 37, G_2 = 21$ and at least six iterations ($p \geq 6$). For p smaller than four, generators $G_1 = 23, G_2 = 35$ lead to better performance than generators $G_1 = 37, G_2 = 21$, at large SNR's (Fig. 10). This result can be understood since, with generators $G_1 = 23, G_2 = 35$, the first coefficients $W_{RSC}(d)$ are smaller than with generators $G_1 = 37, G_2 = 21$. For very low SNR's, the BER can sometimes increase during the iterative decoding process. In order to overcome this effect the extrinsic information z_k has been divided by $[1 + \theta|\tilde{\Lambda}_1(d_k)|]$.$\theta$ acts as a stability factor and its value of 0.15 was adopted after several simulation tests at $E_b/N_0 = 0.7$ dB.

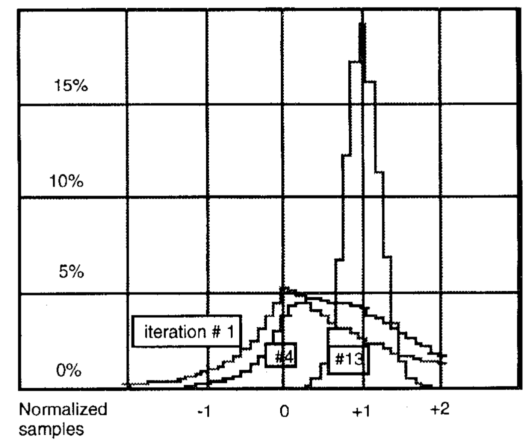


Fig. 11. Histograms of extrinsic information z_k after iterations #1, 4, 13 at $E_b/N_0 = 0.8$ dB; all information bits $d_k = 1$.

We have also plotted the BER for a rate $R = 2/3$ encoder with memory $\nu = 4$, generators $G_1 = 23, G_2 = 35$ and

Authorized licensed use limited to: Yonsei University. Downloaded on December 26, 2009 at 22:55 from IEEE Xplore. Restrictions apply.

parallel concatenation 4/5//4/5 in Fig. 10. For a nonuniform interleaver consisting of a 256×256 matrix and for $p = 3$, the BER is equal to 10^{-5} at $E_b/N_0 = 1.6$ dB and thus the performance is only 1 dB from Shannon's limit.

In Fig. 11, the histogram of extrinsic information $(z)_p$ (generators $G_1 = 37, G_2 = 21.$) has been drawn for several values of iteration p, with all bits equal to one and for a low SNR ($E_b/N_0 = 0.8$ dB). For $p = 1$ (first iteration), extrinsic information $(z)_p$ is very poor about bit d_k, and furthermore, the gaussian hypothesis made above for extrinsic information $(z)_p$, is not satisfied! Nevertheless, when iteration p increases, the histogram merges toward a Gaussian law with a mean equal to one, after normalization. For instance, for $p = 13$, extrinsic information $(z)_p$ becomes relevant information concerning bits d_k.

Note that the concept of turbo-codes has been recently extended to block codes, in particular by Pyndiah *et al.* [13].

ACKNOWLEDGMENT

The authors wish to express their grateful acknowledgment to D. Divsalar for his constructive remarks about the organization of this paper.

REFERENCES

[1] G. D. Forney, "The Viterbi algorithm," *Proc. IEEE,* vol. 61, no. 3, pp. 268–278, Mar. 1973.
[2] J. B. Cain, G. C. Clark, and J. M. Geist, "Punctured convolutional codes of rate (n-1)/n and simplified maximum likelihood decoding," *IEEE Trans. Inform. Theory,* vol. IT-25, pp. 97–100, Jan 1979.
[3] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Trans. Inform. Theory,* vol. IT-28, no. 1, pp. 55–67, Jan. 1982.
[4] Y. Yasuda, K. Kashiki, and Y. Hirata, "High-rate punctured convolutional codes for soft-decision Viterbi decoding," *IEEE Trans. Commun.,* vol. COM-32, no. 3, Mar. 1984.
[5] A. J. Viterbi and J. K. Omura. *Principles of digital communication and coding.* New York: MacGraw-Hill, 1979.
[6] A. J. Viterbi, "Convolutional codes and their performance in communications systems," *IEEE Trans. Commun.,* vol. COM-19, Oct. 1971.
[7] M. Cedervall and R. Johannesson, "A fast algorithm for computing distance spectrum of convolutional codes," *IEEE Trans. Inform. Theory,* vol. 35, no. 6, Nov. 1989.
[8] P. Thitimajshima, "Les codes convolutifs récursifs systématiques et leur application à la concaténation parallèle," (in French), Ph.D. no. 284, Université de Bretagne Occidentale, Brest, France, Dec. 1993.
[9] E. Dunscombe and F. C. Piper, "Optimal interleaving scheme for convolutional coding," *Electron. Lett.,* vol. 25, no. 22, pp. 1517–1518, Oct. 1989.
[10] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory,* vol. IT-20, pp. 248–287, Mar. 1974.
[11] J. Hagenauer, P. Robertson, and L. Papke, "Iterative ("TURBO") decoding of systematic convolutional codes with the MAP and SOVA algorithms," in *Proc. ITG'94,* 1994
[12] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *ICC'93,* Geneva, Switzerland, May 93, pp. 1064–1070.
[13] R. Pyndiah, A. Glavieux, A. Picart, and S. Jacq, "Near optimum decoding of products codes," in *GLOBECOM'94,* San Francisco, CA, Dec. 94, pp. 339–343.

Claude Berrou (M'84) was born in Penmarc'h, France in 1951. He received the electrical engineering degree from the "Institut National Polytechnique", Grenoble, France, in 1975.

In 1978, he joined the Ecole Nationale Supérieure des Télécommunications de Bretagne (France Télécom University), where he is currently a Professor of electronic engineering. His research focuses on joint algorithms and VLSI implementations for digital communications, especially error-correcting codes, and synchronization techniques.

Alain Glavieux was born in France in 1949. He received the engineering degree from the Ecole Nationale Supérieure des Télécommunications, Paris, France in 1978.

He joined the Ecole Nationale Supérieure des Télécommunications de Bretagne, Brest, France, in 1979, where he is currently Head of Department Signal and Communications. His research interests include channel coding, communications over fading channels, and digital modulation.

Authorized licensed use limited to: Yonsei University. Downloaded on December 26, 2009 at 22:55 from IEEE Xplore. Restrictions apply.