# Near-Optimum Decoding of Product Codes: Block Turbo Codes

Ramesh Mahendra Pyndiah, *Member, IEEE*

***Abstract*—This paper describes an iterative decoding algorithm for any product code built using linear block codes. It is based on soft-input/soft-output decoders for decoding the component codes so that near-optimum performance is obtained at each iteration. This soft-input/soft-output decoder is a Chase decoder which delivers soft outputs instead of binary decisions. The soft output of the decoder is an estimation of the log-likelihood ratio (LLR) of the binary decisions given by the Chase decoder. The theoretical justifications of this algorithm are developed and the method used for computing the soft output is fully described. The iterative decoding of product codes is also known as block turbo code (BTC) because the concept is quite similar to turbo codes based on iterative decoding of concatenated recursive convolutional codes. The performance of different Bose–Chaudhuri–Hocquenghem (BCH)–BTC's are given for the Gaussian and the Rayleigh channel. Performance on the Gaussian channel indicates that data transmission at 0.8 dB of Shannon's limit or more than 98% ($R/C >$ 0.98) of channel capacity can be achieved with high-code-rate BTC using only four iterations. For the Rayleigh channel, the slope of the bit-error rate (BER) curve is as steep as for the Gaussian channel without using channel state information.**

***Index Terms*— BCH coding, block codes, codes, maximum-likelihood decoding, product codes, sequential decoding.**

## I. Introduction

THE REAL difficulty in the field of channel coding is essentially a problem of decoding complexity of powerful codes. One approach to this problem is to reduce the decoding complexity of the powerful codes available. This approach has produced substantial reductions in the decoding complexity of block [1], [2] or convolutional codes, but the decoding complexity remains prohibitive for the most powerful codes. Another way to tackle the problem is to construct good codes which exhibit reasonable decoding complexity, and one possible solution is to use concatenated codes [3]. The strategy of concatenated coding is to build powerful error-correcting codes by associating two or more codes with reasonable decoding complexity. The concatenated codes are decoded one component code after the other so that the overall decoding complexity remains acceptable. A practical example of concatenated coding is the coding scheme based on a convolutional code concatenated with a Reed–Solomon (RS) code [4], [5]. This coding scheme exhibits a low bit-error rate (BER) for a code rate close to or even above the channel cutoff rate [6], which was considered as the practical channel capacity until very recently.

In 1993 Berrou [7] showed that it was possible to transmit data with a code rate above the channel cutoff rate. He even achieved an exceptionally low BER with a signal-to-noise ratio (SNR) per information bit $(E_b/N_0)$ close to Shannon's theoretical limit on a Gaussian channel. This new coding scheme [7], [8] consists of two recursive systematic convolutional codes concatenated in parallel and which are decoded using iterative maximum-likelihood decoding (MLD) (or soft decoding) of the component codes. For the decoding of the component codes, Berrou used a maximum *a posteriori* (MAP) algorithm [9] which performs maximum-likelihood (ML) bit estimation and thus yields a reliability information (soft-output) for each bit. This algorithm can be viewed as a soft-input/soft-output decoder. By cascading several of these decoders, one can perform an iterative ML decoding of the component codes which is optimal at each decoding step. Berrou achieved a BER of $10^{-5}$ for an $E_b/N_0$ at 0.5 dB above Shannon's limit for rate-1/2 quadrature phase-shift keying (QPSK) on a Gaussian channel and, very recently, Berrou has narrowed the gap to 0.35 dB [10]. The MAP algorithm has a very large computation complexity and, for practical applications, suboptimal weighting algorithms have already been proposed [11], [12]. These suboptimal algorithms introduce a performance degradation of approximately 1.5 dB.

The new coding scheme described above has been given the name of turbo code and has received much attention since 1993. A lot of papers have been published on turbo codes, but most of the authors have focused on convolutional turbo codes (CTC's) and very few have considered the block turbo code (BTC) [13], [14]. In fact, concatenated coding was first introduced for block codes [3], [15]. Unfortunately, the first algorithms [16], [17] proposed for decoding these codes gave rather poor results because they relied on hard-input/hard-output decoders and lacked the soft-input/soft-output decoders. Recent work by Lodge [13] and Hagenauer [14] have produced solutions with good performance. These solutions are based on the trellis of block codes and are limited to small block codes since the number of states in the trellis of a block code increases exponentially with the number of redundancy bits. In 1994 we proposed a new soft-input/soft-output decoder [18] for all linear block codes and we showed that BTC had performances comparable to those of CTC using suboptimal weighting algorithms. This new BTC offers a good

Paper approved by S. S. Pietrobon, the Editor for Coding Theory and Techniques of the IEEE Communications Society. Manuscript received July 7, 1997; revised December 11, 1997 and March 9, 1998. This work was supported by the Centre National d'Etudes des Télécommunications (CNET), Rennes, France. This paper was presented in part at the GLOBECOM'94 Conference, San Francisco, CA, November 27–December 1, 1994.

The author is with l'Ecole Nationale Supérieure des Télécommunications (ENST) de Bretagne, Technopôle Brest Iroise, BP-832, 29285 Brest, France (e-mail: Ramesh.Pyndiah@enst-bretagne.fr).

Publisher Item Identifier S 0090-6778(98)05574-3.

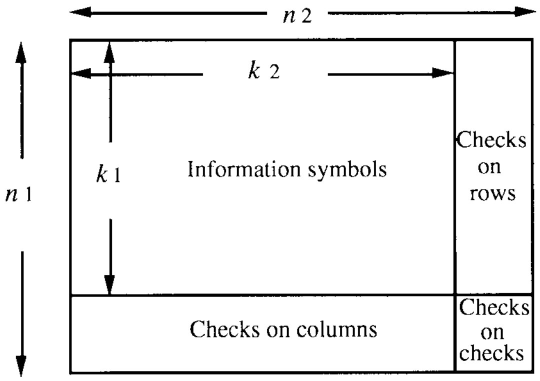


Fig. 1. Construction of product code $\mathcal{P} = \mathcal{C}^1 \otimes \mathcal{C}^2$.

compromise between performance and complexity and is very attractive for implementation.

The aim of this paper is to give a clear and complete description of the BTC proposed in [18]. In Section II we recall the basic concept of product codes and in Section III we give a brief description of the Chase algorithm for soft-input decoding of block codes. The theoretical equations of the soft output of a block decoder is derived in Section IV and the algorithm used for computing the soft output is described in Section V together with justifications of the different simplifications. The concept of block turbo decoding is presented in Section VI and performance is given for different BTC's on Gaussian and Rayleigh channels. The potential applications of BTC are discussed in Section VII.

## II. Product Codes

Product codes (or iterated codes) are serially concatenated codes [19] which were introduced by Elias in 1954 [15]. The concept of product codes is very simple and relatively efficient for building very long block codes by using two or more short block codes. Let us consider two systematic linear block codes $\mathcal{C}^1$ with parameters $(n_1, k_1, \delta_1)$ and $\mathcal{C}^2$ with parameters $(n_2, k_2, d_2)$, where $n_i, k_i$ and $\delta_i$ $(i = 1, 2)$ stand for codeword length, number of information bits, and minimum Hamming distance, respectively. The product code $\mathcal{P} = \mathcal{C}^1 \otimes \mathcal{C}^2$ is obtained (see Fig. 1) by

1) placing $(k_1 \times k_2)$ information bits in an array of $k_1$ rows and $k_2$ columns;
2) coding the $k_1$ rows using code $\mathcal{C}^2$;
3) coding the $n_2$ columns using code $\mathcal{C}^1$.

The parameters of the product code $\mathcal{P}$ [19] are $n = n_1 \times n_2$, $k = k_1 \times k_2$, $\delta = \delta_1 \times \delta_2$, and the code rate $R$ is given by $R = R_1 \times R_2$, where $R_i$ is the code rate of code $\mathcal{C}^i$. Thus, we can build very long block codes with large minimum Hamming distance by combining short codes with small minimum Hamming distance. Given the procedure used to construct the product code, it is clear that the $(n_2 - k_2)$ last columns of the matrix are codewords of $\mathcal{C}^1$. By using the matrix generator, one can show [19] that the $(n_1 - k_1)$ last rows of matrix $\mathcal{P}$ are codewords of $\mathcal{C}^2$. Hence, all of the rows of matrix $\mathcal{P}$ are codewords of $\mathcal{C}^1$ and all of the columns of matrix $\mathcal{P}$ are codewords of $\mathcal{C}^2$.

As indicated by Elias [15], these codes can be decoded by sequentially decoding the rows and columns of $\mathcal{P}$ in order to reduce decoding complexity. However, to achieve optimum performance, one must use MLD (soft decoding) of the component codes. Thus, we need soft-input/soft-output decoders to maintain optimum performance when decoding the rows and columns of $\mathcal{P}$. Provided we have a soft-input/soft-output decoder for decoding the rows and columns of $\mathcal{P}$, we can iterate the sequential decoding of $\mathcal{P}$ and thus reduce the BER after each iteration as for CTC [7].

In Section III we consider the soft-input decoding of block codes, and soft output (reliability) of the decoded bits will be addressed in Section IV.

## III. Soft Decoding of Linear Block Codes

Let us consider the transmission of binary elements $\{0, 1\}$ coded by a linear block code $\mathcal{C}$ with parameters $(n, k, \delta)$ on a Gaussian channel using binary symbols $\{-1, +1\}$. We shall consider the following mapping of the symbols $0 \rightarrow -1$ and $1 \rightarrow +1$. The observation $\boldsymbol{R} = (r_1, \cdots, r_l, \cdots, r_n)$ at the output of the Gaussian channel for a transmitted codeword $\boldsymbol{E} = (e_1, \cdots, e_l, \cdots, e_n)$ is given by

$$\boldsymbol{R} = \boldsymbol{E} + \boldsymbol{G} \tag{1}$$

where components $g_l$ of $\boldsymbol{G} = (g_1, \cdots, g_l, \cdots g_n)$ are additive white Gaussian noise (AWGN) samples of standard deviation $\sigma$. By using MLD, one can show that the optimum decision $\boldsymbol{D} = (d_1, \cdots, d_l, \cdots, d_n)$ corresponding to the transmitted codeword $\boldsymbol{E}$ is given by

$$\boldsymbol{D} = \boldsymbol{C}^i \text{ if } |\boldsymbol{R} - \boldsymbol{C}^i|^2 \leq |\boldsymbol{R} - \boldsymbol{C}^l|^2 \quad \forall l \in [1, 2^k], \qquad l \neq i \tag{2}$$

where $\boldsymbol{C}^i = (c_1^i, \cdots, c_l^i, \cdots, c_n^i)$ is the $i$th codeword of $\mathcal{C}$ and

$$|\boldsymbol{R} - \boldsymbol{C}^i|^2 = \sum_{l=1}^{n} (r_l - c_l^i)^2 \tag{3}$$

is the squared Euclidean distance between $\boldsymbol{R}$ and $\boldsymbol{C}^i$. When using an exhaustive search for the optimum codeword $\boldsymbol{D}$, the computation complexity increases exponentially with $k$ and becomes prohibitive for block codes with $k > 6$. As suggested by Gallager [20], one needs very long codes in order to approach channel capacity and the exhaustive search is not a realistic solution for those codes considered here with $k > 10$.

In 1972 Chase proposed a suboptimum algorithm of low complexity [21] for near-ML decoding of linear block codes. This algorithm is based on the following observation.

At high SNR, ML codeword $\boldsymbol{D}$ is located in the sphere of radius $(\delta - 1)$ centered on $\boldsymbol{Y} = (y_1, \cdots, y_l, \cdots, y_n)$, where $y_l = 0.5(1 + \mathrm{sgn}(r_l))$ and $y_l \in \{0, 1\}$ with a very high probability.

Thus, we can limit the reviewed codewords in (2) to those in the sphere of radius $(\delta - 1)$ centered on $\boldsymbol{Y}$. To reduce the number of reviewed codewords, only the set of the most probable codewords within the sphere are selected by using channel information $\boldsymbol{R}$. The procedure used to identify the set of the most probable codewords is the following.

*Step 1:* Determine the position of the $p = \lfloor \delta/2 \rfloor$ least reliable binary elements of $\boldsymbol{Y}$ using $\boldsymbol{R}$. The reliability of the elements of $\boldsymbol{Y}$ will be defined later on.

*Step 2:* Form test patterns $\boldsymbol{T}^q$ defined as all the $n$-dimensional binary vectors with a single "1" in the least reliable positions and "0" in the other positions, two "1"s in the least reliable positions and "0" in the other positions, and $\cdots, p$ "1"s in the least reliable positions and "0" in the other positions.

*Step 3:* Form test sequences $\boldsymbol{Z}^q$ where $z_l^q = y_l \oplus t_l^q$ and decode $\boldsymbol{Z}^q$ using an algebraic (or hard) decoder and add the codeword $\boldsymbol{C}^q$ to subset $\Omega$.

Decision $\boldsymbol{D}$ is then given by applying decision rule (2) with the reviewed codewords restricted to the subset of codewords $\Omega$ found at step 3 above. Note that the components of the codewords are mapped from $\{0, 1\}$ to $\{-1, +1\}$ before computing the Euclidean distance. In step 1 the reliability of component $y_j$ is defined using the log-likelihood ratio (LLR) of decision $y_j$

$$\Lambda(y_j) = \ln\left(\frac{\Pr\{e_j = +1/r_j\}}{\Pr\{e_j = -1/r_j\}}\right) = \left(\frac{2}{\sigma^2}\right) r_j. \qquad (4)$$

If we consider a stationary channel, we can normalize the LLR with respect to constant $2/\sigma^2$, and the relative reliability of $y_j$ is then given by $|r_j|$.

Coming back to the decoding of a product code, the Chase algorithm yields for each row (or column) the decision $\boldsymbol{D}$ of the component block code for a given input data $\boldsymbol{R}$. To iterate the decoding procedure with maximum efficiency, we must now compute the reliability of the decisions given by the Chase algorithm before decoding the columns (or rows).

## IV. Reliability of Decision $\boldsymbol{D}$ Given by Soft-Input Decoder

Once we have determined decision $\boldsymbol{D}$ of a row (or column) of the product code, we have to compute the reliability of each of the components of vector $\boldsymbol{D}$ in order to generate soft decisions at the output of the decoder. The reliability of decision $d_j$ is defined using the LLR of transmitted symbol $e_j$, which is given by

$$\Lambda(d_j) = \ln\left(\frac{\Pr\{e_j = +1/\boldsymbol{R}\}}{\Pr\{e_j = -1/\boldsymbol{R}\}}\right). \qquad (5)$$

Here, the computation of the LLR differs from the previous case [see (4)], in that we must take into account the fact that $\boldsymbol{D}$ is one of the $2^k$ codewords of $\boldsymbol{C}$. Thus, by considering the different codewords of $\boldsymbol{C}$, the numerator of (5) can be written as

$$\Pr\{e_j = +1/\boldsymbol{R}\} = \sum_{\boldsymbol{C}^i \in S_j^{+1}} \Pr\{\mathrm{E} = \boldsymbol{C}^i/\boldsymbol{R}\} \qquad (6)$$

where $S_j^{+1}$ is the set of codewords $\{\boldsymbol{C}^i\}$ such that $c_j^i = +1$ and the denominator of (5) can be written in the form

$$\Pr\{e_j = -1/\boldsymbol{R}\} = \sum_{\boldsymbol{C}^i \in S_j^{-1}} \Pr\{\boldsymbol{E} = \boldsymbol{C}^i/\boldsymbol{R}\} \qquad (7)$$

where $s_j^{-1}$ is the set of codewords $\{\boldsymbol{C}^i\}$ such that $c_j^i = -1$. By applying Bayes' rule to (6) and (7) and assuming that the different codewords are uniformly distributed, we obtain for $\Lambda(d_j)$ the following expression:

$$\Lambda(d_j) = \ln\left(\frac{\displaystyle\sum_{\boldsymbol{C}^i \in S_j^{-1}} p\{\boldsymbol{R}/\boldsymbol{E} = \boldsymbol{C}^i\}}{\displaystyle\sum_{\boldsymbol{C}^i \in S_j^{-1}} p\{\boldsymbol{R}/\boldsymbol{C}^i\}}\right) \qquad (8)$$

where

$$p\{\boldsymbol{R}/\boldsymbol{E} = \boldsymbol{C}^i\} = \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n \exp\left(-\frac{|\boldsymbol{R} - \boldsymbol{C}^i|^2}{2\sigma^2}\right) \qquad (9)$$

is the probability density function of $\boldsymbol{R}$ conditioned on $\boldsymbol{E}$. This function decreases exponentially with the Euclidean distance between $\boldsymbol{R}$ and $\boldsymbol{C}^i$. Let $\boldsymbol{C}^{+1(j)}$ and $\boldsymbol{C}^{-1(j)}$ be the codewords, respectively, in $S_j^{+1}$ and $S_j^{-1}$, at minimum Euclidean distance from $\boldsymbol{R}$. By combining (8) and (9), we obtain the following relation:

$$\Lambda(d_j) = \frac{1}{2\sigma^2}\left(|\boldsymbol{R} - \boldsymbol{C}^{-1(j)}|^2 - |\boldsymbol{R} - \boldsymbol{C}^{+1(j)}|^2\right) + \ln\left(\frac{\displaystyle\sum_i A_i}{\displaystyle\sum_i B_i}\right) \qquad (10)$$

where

$$A_i = \exp\left(\frac{|\mathrm{R} - \boldsymbol{C}^{+1(j)}|^2 - |\boldsymbol{R} - \boldsymbol{C}^i|^2}{2\sigma^2}\right) \le 1 \quad \text{with } \boldsymbol{C}^i \in S^{+1(j)} \qquad (11)$$

and

$$B_i = \exp\left(\frac{|\mathrm{R} - \boldsymbol{C}^{-1(j)}|^2 - |\boldsymbol{R} - \boldsymbol{C}^i|^2}{2\sigma^2}\right) \le 1 \quad \text{with } \boldsymbol{C}^i \in S^{-1(j)}. \qquad (12)$$

For high SNR, that is, $\sigma \to 0$, $\Sigma_i\ A_i \approx \Sigma_i\ B_i \to 1$ and thus the second term in (10) tends to zero. By neglecting the second term in (10), we obtain an approximation for the LLR of decision $d_j$ equal to

$$\Lambda'(d_j) = \frac{1}{2\sigma^2}\left(|\boldsymbol{R} - \boldsymbol{C}^{-1(j)}|^2 - |\boldsymbol{R} - \boldsymbol{C}^{+1(j)}|^2\right). \qquad (13)$$

By expanding (13) using (3), we obtain the following relation:

$$\Lambda'(d_j) = \frac{2}{\sigma^2}\left(r_j + \sum_{l=1, l \ne j}^{n} r_l c_l^{+1(j)} p_l\right) \qquad (14)$$

where

$$p_l = \begin{cases} 0, & \text{if } c_l^{+1(j)} = c_l^{-1(j)} \\ 1, & \text{if } c_l^{+1(j)} \ne c_l^{-1(j)} \end{cases}. \qquad (15)$$

If we suppose that $\sigma$ is constant, we can normalize $\Lambda'(d_j)$ with respect to the constant $2/\sigma^2$ and we obtain the following equation:

$$r_j' = r_j + w_j \qquad (16)$$

with

$$w_j = \sum_{l=1, l \ne j}^{n} r_l c_l^{+1(j)} p_l. \qquad (17)$$

The normalized LLR $r'_j$ is taken as the soft output of the decoder. It has the same sign as $d_j$ and its absolute value indicates the reliability of the decision. Equation (16) indicates that $r'_j$ is given by the soft-input data $r_j$ plus a term $w_j$ which is a function of the two codewords at minimum Euclidean distance from $\boldsymbol{R}$ and $\{r_l\}$ with $l \neq j$. The term $w_j$ is a correction term applied to the input data and it plays the same role as the extrinsic information in CTC [8]. The extrinsic information is a random variable with a Gaussian distribution since it is a linear combination of identically distributed random variables. Furthermore, it is uncorrelated with the input data $r_j$. As for CTC, the extrinsic information plays a very important role in the iterative decoding of product codes.

Based on the theoretical justifications elaborated in this section, we shall now describe in Section V the algorithm we use for computing the reliability of decision $d_j$.

## V. Computing the Soft Decision at the Output of the Soft-Input Decoder

Computing the reliability of decision $d_j$ at the output of the soft-input decoder requires two codewords $\boldsymbol{C}^{+1(j)}$ and $\boldsymbol{C}^{-1(j)}$; see (13). Obviously, soft decision $\boldsymbol{D}$ is one of these two codewords and we must find the second one, which we shall call $\boldsymbol{C}$. $\boldsymbol{C}$ can be viewed as a competing codeword of $\boldsymbol{D}$ at minimum Euclidean distance from $\boldsymbol{R}$ with $c_j \neq d_j$. Given codeword $\boldsymbol{C}$ and $\boldsymbol{D}$, one can show that the soft output is given by the following equation:

$$r'_j = \left(\frac{|\boldsymbol{R}-\boldsymbol{C}|^2 - |\boldsymbol{R}-\boldsymbol{D}|^2}{4}\right)d_j. \tag{18}$$

To find codeword $\boldsymbol{C}$, one must increase the size of the space scanned by the Chase algorithm (see Section III). For that purpose, we increase the number of least reliable bits $p$ used in the Chase decoder and also the number of test patterns. It is clear that the probability of finding $\boldsymbol{C}$ increases with the value of $p$. On the other hand, the complexity of the decoder increases exponentially with $p$ and we must find a tradeoff between complexity and performance. This implies that in some cases we shall not be able to find a competing codeword $\boldsymbol{C}$. In the event where codeword $\boldsymbol{C}$ is not found, we must find another method for computing the soft output. The solution we propose is to use the following equation:

$$r'_j = \beta \times d_j \text{ with } \beta \geq 0. \tag{19}$$

This solution, which is very simple, happens to be very efficient as will be shown by the simulation results in Section VI. This very rough approximation of the soft output is justified by the fact that:

- the sign of soft output $r'_j$ is equal to $d_j$ [see (18)], while only its absolute value or reliability is a function of $\boldsymbol{C}$;
- if $\boldsymbol{C}$ is not found in the space scanned by the Chase algorithm, then $\boldsymbol{C}$ is most probably far from $\boldsymbol{R}$ in terms of Euclidean distance;
- if $\boldsymbol{C}$ is very far from $\boldsymbol{R}$, then the probability that decision $d_j$ is correct is relatively high and the reliability of $d_j$ is also relatively high.

Thus, we propose to give a predefined value $\beta$ to the reliability of those components of $\boldsymbol{D}$ for which there is no competing codeword $\boldsymbol{C}$ in subset $\Omega$. The value of $\beta$ was initially optimized by trial and error [18]. An equation is given in [23] for computing $\beta$

$$\beta \approx \left|\ln\left(\frac{\Pr\{d_j = e_j\}}{\Pr\{d_j \neq e_j\}}\right)\right| = \ln\left(\frac{\Pr\{d_j = e_j\}}{\Pr\{d_j \neq e_j\}}\right) \tag{20}$$

where $\Pr\{d_j = e_j\}$ represents the probability that the decoder takes the correct decision and it takes its values in the interval [0.5, 1]. When $\Pr\{d_j = ej\} \rightarrow 1$, then $\beta \rightarrow \infty$, and when $\Pr\{d_j = e_j\} \rightarrow 0.5$, then $\beta \rightarrow 0$. Note that a similar equation was proposed by Hagenauer in 1996 for computing the probability of the ML path in a trellis [14]. Equation (20) for computing $\beta$ is coherent with the notion of reliability of decision $d_j$. When the probability of taking the correct decision tends to one, the reliability of $d_j$ tends to infinity, and when it tends to 0.5, the reliability tends to zero. In fact, $\beta$ can be considered as an average value of the reliability of those decisions $d_j$ for which there is no competing codeword $\boldsymbol{C}$ in subset $\Omega$, while (18) gives a bit-by-bit estimation of the reliability. It is clear that the soft output given by (19) is less accurate than the one using (18). However, one can understand that we need a more accurate estimation of the soft output for those decisions where a competing codeword $\boldsymbol{C}$ is at a slightly greater distance from $\boldsymbol{R}$ than $\boldsymbol{D}$. On the other hand, when $\boldsymbol{C}$ is very far from $\boldsymbol{R}$, an average value of the reliability can be considered as sufficient.

Now that we have defined the soft output of the block decoder, we shall consider in the next section the turbo decoding of product codes.

## VI. Turbo Decoding of Product Codes

Let us consider the decoding of the rows and columns of a product code $\mathcal{P}$ described in Section II and transmitted on a Gaussian channel using QPSK signaling. On receiving matrix $[\boldsymbol{R}]$ corresponding to a transmitted codeword $[\boldsymbol{E}]$, the first decoder performs the soft decoding of the rows (or columns) of $\mathcal{P}$ using as input matrix $[\boldsymbol{R}]$. Soft-input decoding is performed using the Chase algorithm (see Section III) and the soft output is computed using (18) or (19). By subtracting the soft input from the soft output [see (16)] we obtain the extrinsic information $[\boldsymbol{W}(2)]$ where index 2 indicates that we are considering the extrinsic information for the second decoding of $\mathcal{P}$ which was computed during the first decoding of $\mathcal{P}$. The soft input for the decoding of the columns (or rows) at the second decoding of $\mathcal{P}$ is given by

$$[\boldsymbol{R}(2)] = [\boldsymbol{R}] + \alpha(2)[\boldsymbol{W}(2)] \tag{21}$$

where $\alpha(2)$ is a scaling factor which takes into account the fact that the standard deviation of samples in matrix $[\boldsymbol{R}]$ and in matrix $[\boldsymbol{W}]$ are different (see [7] and [8]). The standard deviation of the extrinsic information is very high in the first decoding steps and decreases as we iterate the decoding. This scaling factor $\alpha$ is also used to reduce the effect of the extrinsic information in the soft decoder in the first decoding steps when the BER is relatively high. It takes a small value in the first

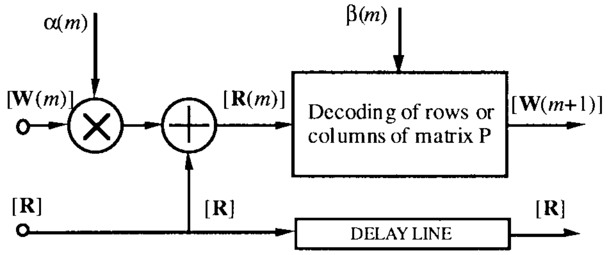


Fig. 2. Block diagram of elementary block turbo decoder.

decoding steps and increases as the BER tends to zero. The decoding procedure described above is then generalized by cascading elementary decoders illustrated in Fig. 2.

The turbo decoding algorithm described in this paper applies to any product code based on linear block codes. The results we present here concern Bose–Chaudhuri–Hocqenghem (BCH) product codes. Performance of RS product codes are given in [22]. For complexity considerations, we consider extended single and double error correcting BCH codes. Using extended BCH codes is advantageous because it gives a significant increase to the minimum distance of the product code for a very small reduction in code rate and a negligible additional decoding complexity [23]. For example, if we consider single error correcting BCH codes, the minimum distance of the product code is nine for the BCH code and 16 for the extended BCH code. The same is true for expurgated codes.

Before proceeding to the simulation results, we shall now give the different parameters used by this turbo decoding algorithm. For the sake of clarity we have deliberately used the same parameters in the turbo decoding algorithms for all of the BTC's presented here. These parameters have been determined experimentally. From one BTC to the other, the only modifications concern the binary BCH encoder and the algebraic decoder. The characteristics of the turbo decoding algorithm are as follows.

1) *Test sequences:* The number of test patterns is 16 and are generated by the four least reliable bits $(p = 4)$.
2) *Weighting factor $\alpha$:* To reduce the dependency of $\alpha$ on the product code, the mean absolute value of the extrinsic information $|w|$ derived using (18) is normalized to one. The evolution of $\alpha$ with the decoding number is

$$\alpha(m) = [0.0, 0.2, 0.3, 0.5, 0.7, 0.9, 1.0, 1.0].$$

3) *Reliability factor $\beta$:* To operate under optimal conditions, the reliability factor should be determined as a function of the BER. For practical considerations, we have fixed the evolution of $\beta$ with the decoding step to the following values:

$$\beta(m) = [0.2, 0.4, 0.6, 0.8, 1.0, 1.0, 1.0, 1.0].$$

In the first decoding step $\beta$ is set to 20% of the mean of the normalized extrinsic information computed using (18) and is gradually increased to 100%. Note that experimental results indicate no significant performance degradation if values of $\beta$ is modified by ±10%.

TABLE I
Parameters of BTC Considered Here

| Product code | δ | R |
|---|---|---|
| $(32,26,4)^2$ | 16 | 0.660 |
| $(32,21,6)^2$ | 36 | 0.431 |
| $(64,57,4)^2$ | 16 | 0.793 |
| $(64,51,6)^2$ | 36 | 0.635 |
| $(128,120,4)^2$ | 16 | 0.879 |
| $(128,113,6)^2$ | 36 | 0.779 |
| $(256,247,4)^2$ | 16 | 0.931 |
| $(512,502,4)^2$ | 16 | 0.961 |

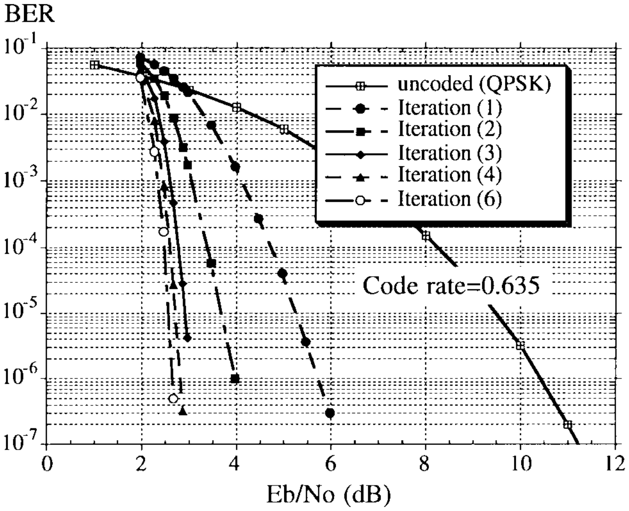


Fig. 3. BER versus $E_b/N_0$ of product code $[\mathrm{BCH}(64, 51, 6)]^2$ on a Gaussian channel using QPSK signaling.

Note that the normalization of the extrinsic information is introduced to reduce the dependency of parameter $\beta$ on the code parameters $(n, k, \delta)$. It is clear that for a given application one should remove this normalization function and reoptimize $\beta$ with initial values determined using (20). In our investigation we have considered product codes combining identical BCH codes as well as different BCH codes. In terms of performance optimality we have not observed any significant amelioration when combining BCH codes of different length or different minimum distance. In this paper we consider product codes using identical BCH codes, which are more suitable for implementation. The parameters of the product codes considered here are listed in Table I. We shall now comment on the performance of the BTC in Table I on the Gaussian and the Rayleigh channel using QPSK signaling. One iteration of the turbo decoder corresponds to a row followed by a column decoding of the product code.

### *A. Gaussian Channel Using QPSK Signaling*

The performance of BTC $(64, 51, 6)^2$ is given in Fig. 3 where $(64, 51, 6)^2$ is product code (64, 51, 6) $\otimes$ (64, 51, 6) defined in Section II. We observe the turbo decoding effect as for CTC [8]. For an $E_b/N_0$ of 2.7 dB, the BER is equal to $3 \times 10^{-2}$ at iteration 1, $9 \times 10^{-3}$ at iteration 2, $5 \times 10^{-4}$ at iteration 3, $3 \times 10^{-5}$ at iteration 4, and $5 \times 10^{-7}$ at

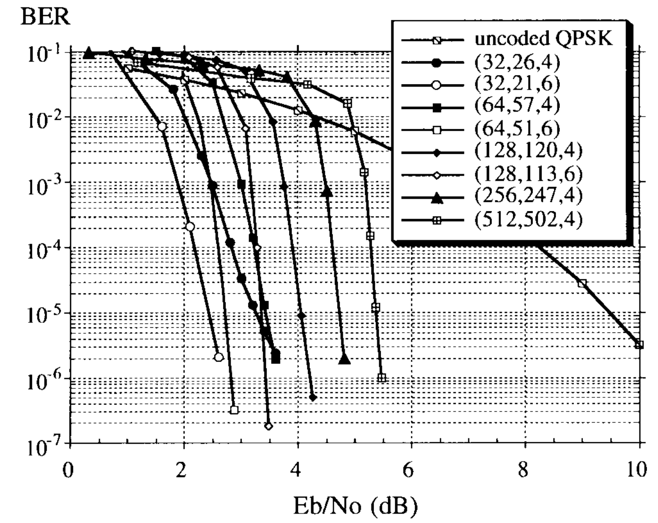


Fig. 4. BER versus $E_b/N_0$ of BCH product codes on a Gaussian channel using QPSK signaling at iteration 4.

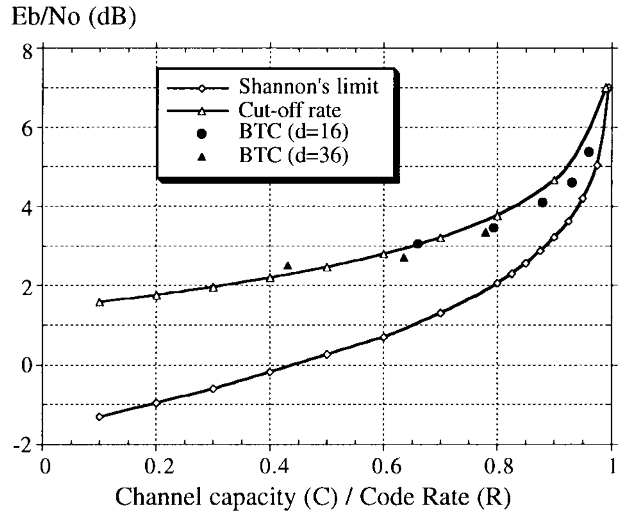


Fig. 5. $E_b/N_0$ for a BER of $10^{-5}$ at iteration 4 of BCH product codes on a Gaussian channel using QPSK signaling compared with theoretical limits.

iteration 6. For each additional iteration, we obtain a reduction of the BER. However, for a BER of $10^{-5}$, the reduction in terms of $E_b/N_0$ becomes negligible for additional iterations beyond iteration 4. In Fig. 4 we have plotted the BER against $E_b/N_0$ for different BTC's on a Gaussian channel using QPSK signaling at iteration 4. We observe that the slope of the BER curves increases with parameter $n$ (or $k$) and also with $\delta$.

In Fig. 5 we have considered the $E_b/N_0$ required by the different BTC at iteration 4 in order to achieve a BER of $10^{-5}$. This $E_b/N_0$ at $10^{-5}$ has been plotted versus the code rate and compared to two theoretical limits: 1) the cutoff rate and 2) channel capacity for a binary input Gaussian channel [24]. Except for product code (32, 21, 6)$^2$, all of the other codes considered here achieve a BER of $10^{-5}$ at a code rate above their respective cutoff rates. For product codes of same minimum distance, the performance achieved gets closer to Shannon's limit as we increase the parameter $k$. This is consistent with Gallager's theory [20]. We observe that for product code (512, 502, 4)$^2$, the $E_b/N_0$ for a BER of $10^{-5}$ is at less than 0.8 dB of its Shannon's limit and that this coding system is operating at near channel capacity $R/C > 0.98$. By increasing the number of iterations to 50, we have been able to achieve a BER of $10^{-5}$ at 0.45 dB of Shannon's limit. To our knowledge, the closest result to Shannon's limit using a parallel concatenation of code (511, 502, 3) is actually 0.35 dB and has been achieved by Nickl and Hagenauer [25]. The algorithm used in [25] is nearly optimum but has a complexity which increases exponentially with the number of redundancy bits. Although this algorithm has a prohibitive complexity for practical applications, it is nevertheless a reference in terms of performance. Thus we can consider that the algorithm proposed here is nearly optimum since we are at 0.1 dB of Hagenauer's reference.

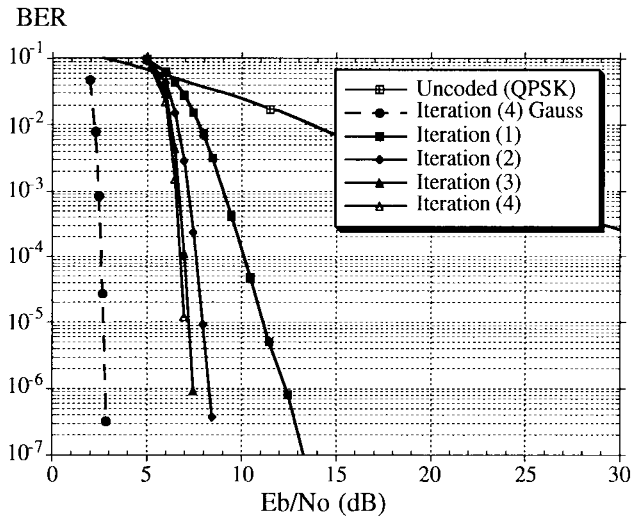


Fig. 6. BER versus $E_b/N_0$ of BTC (64, 51, 6)$^2$ on a Rayleigh channel using QPSK signaling.

### B. Rayleigh-Fading Channel Using QPSK Signaling

The BER versus $E_b/N_0$ curves of BTC (64, 51, 6)$^2$ on the Rayleigh-fading channel using QPSK signaling are given in Fig. 6. As for the Gaussian channel, we observe the turbo decoding effect and a very steep slope of the curve at iteration 4. Although the curve at iteration 1 is relatively flat, there is a significant improvement of the curve at iteration 2 with a saturation of the improvement beyond iteration 3. The curve at iteration 4 is as steep as for the Gaussian channel (see Fig. 6) with a constant shift of 4.6 dB to the right. This behavior of the BTC on a Rayleigh channel can be explained by the fact that the soft output of the elementary decoder tends to a Gaussian distribution according to the central limit theorem [see (14)]. At iteration 1, the soft data has a Rayleigh distribution which explains the flatness of the curve, while beyond iteration 1 the soft data tends to a Gaussian distribution. It is worth noting that these results have been established in the case where the decoder does not have any information on the channel state. If we introduce the channel-state information in the decoder, the improvement is negligible (BER is divided by two).

The BER versus $E_b/N_0$ of different BTC's at iteration 4 on the Rayleigh channel using QPSK signaling are given in

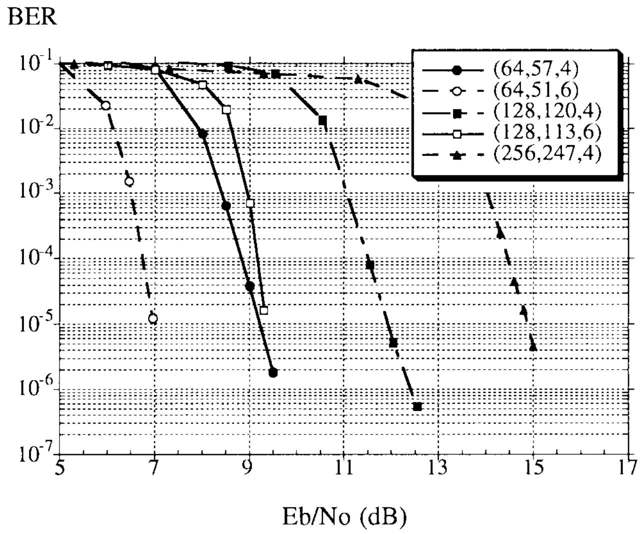


Fig. 7. BER versus $E_b/N_0$ of different BTC's on a Rayleigh channel using QPSK signaling at iteration 4.

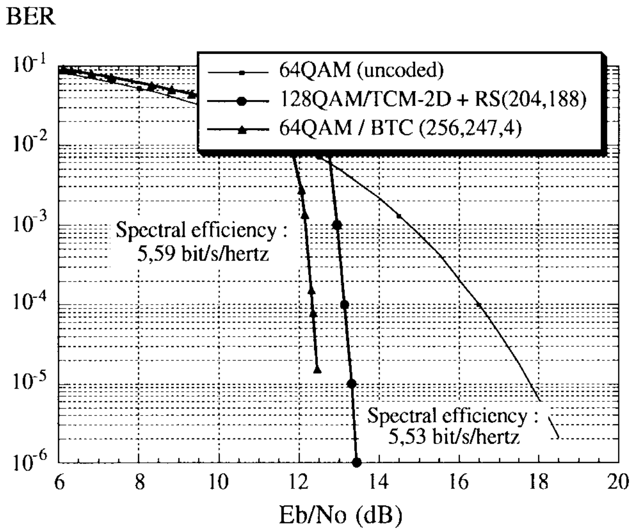


Fig. 8. Comparison of a high-spectral-efficiency BTC with a concatenated scheme combining a TCM with an outer RS code.

Fig. 7. We observe that the slope of the curves increases with minimum distance of the code and code length.

### C. Gaussian Channel Using QAM Signaling

The performance of several BTC's associated with quadrature amplitude modulation (QAM) signaling have already been investigated in [27]. This study considered the pragmatic approach proposed by Viterbi [26] to combine QAM signaling with BTC. In this approach the QPSK modulator is replaced by a QAM modulator. In the receiver, the LLR of the binary elements used for labeling the signals in the modulator are computed [27] and passed on to the iterative decoder described previously. This solution is very attractive since the same iterative decoder can be used for any phase-shift keying (PSK) or QAM modulation scheme. We presented the performance of several BTC's combined with 16QAM and 64QAM in [27], but we did not compare our scheme with the classical concatenated scheme using a trellis-coded modulation (TCM) concatenated with an outer RS code. In Fig. 8 we give the BER versus $E_b/N_0$ of 128QAM coded by an inner 16-state two-dimensional TCM code concatenated with an outer RS(204, 188) code and assuming perfect interleaving. This transmission system has a spectral efficiency of 5.53 b/s/Hz. On the same graph, we have plotted the BER vesus $E_b/N_0$ of BTC (256, 247, 4)$^2$ combined with a 64QAM signal set [27] at iteration 4. This system has a spectral efficiency of 5.59 b/s/Hz, which is slightly higher than that of the classical scheme. In Fig. 8 we observe that the BTC outperforms the classical scheme by at least 0.85 dB at a BER of $10^{-5}$ and that the slopes of the two BER curves are practically the same.

## VII. Discussion and Conclusion

The performance achieved by the BTC's presented in this paper indicate that they are the most efficient known codes for high code rate applications. For code rates greater than 0.95, digital transmission systems can transmit data on a Gaussian channel at more than 98% of channel capacity $R/C > 0.98$ by using BTC. This result is confirmed by Nickl and Hagenauer [25] using a more optimal decoding algorithm, but with a much higher complexity. The algorithm proposed here can be considered as near optimum for the decoding of product codes since there is no significant difference ($\approx$0.1 dB) in performance between our results and those obtained in [25]. On the other hand, the algorithm proposed here is of a much lower complexity. In [28] an example of a BTC (32, 26, 4)$^2$ implemented on a field programmable gate array (FPGA) is given. The elementary decoder requires less than 6000 gates and measured performance is close to the simulated one given here. The BTC's proposed here are also very attractive for digital signal processor (DSP) implementation because many of the operations can be performed simultaneously in a DSP. A block turbo decoder has been implemented on a 20-million-instructions-per-second (MIPS) Motorola 56 002 fixed point DSP [29]. For decoding code (32, 26, 4)$^2$, this decoder requires a program memory of 1300 words and a data memory of 4300 words giving a data rate of 80 kb/s for one iteration. The BTC presented here offers an excellent compromise between complexity and performance. On the Rayleigh channel, BTC's exhibit a BER versus $E_b/N_0$ curve as steep as on the Gaussian channel because in the iterative decoding process, the soft output tends to a Gaussian distribution for any identically distributed input data.

Currently, most of the work on turbo codes have essentially focused on CTC's, and BTC's have been partially neglected. Yet, the BTC solution is more attractive for a wide range of applications. The results presented here indicate that BTC is one of the best solutions in terms of performance and complexity for high-code-rate applications. The comparison of a BTC with a concatenated TCM-RS coding scheme shows that BTC [27] yields a 0.85-dB improvement in terms of $E_b/N_0$ for high-spectral-efficiency applications. Moreover, for small data block (<150 b) used in TDMA applications, the BTC is more efficient than CTC [23]. The main reason is because the minimum distance of a turbo code becomes crucial when the interleaver size is small. A product code can guarantee a minimum distance of 16, 36 (or more), while the minimum distance of a CTC can be relatively small. Another attractive application for BTC concerns high-data-rate systems. Indeed, the decoding speed of a BTC can be increased by using several elementary decoders for the parallel decoding of the rows (or columns) of a product code since they are independent.

A lot of exciting work remains to be done on BTC. Among others, joint error detection and error correction [23] seems very promising, as well as unequal error protection and BTC for small data blocks.

## Acknowledgment

The author would like to thank P. Adde, O. Aitsab, A. Glavieux, A. Goalic, S. Jacq, A. Picart, and H. Sari for their contribution or support to the development of block turbo codes, as well as Dr. S. Pietrobon and the three anonymous reviewers for their very helpful suggestions and comments.

## References

[1] J. L. Massey, "Shift-register synthesis and BCH decoding," *IEEE Trans. Inform. Theory*, vol. IT-15, pp. 122–127, Jan. 1969.

[2] G. D. Forney, "Generalized minimum distance decoding," *IEEE Trans. Inform. Theory*, vol. IT-12, pp. 125–131, Apr. 1966.

[3] ———, *Concatenated Codes*. Cambridge, MA: MIT Press, 1966.

[4] A. Morello, G. Montorosi, and M. Visintin, "Convolutional and trellis coded modulations concatenated with block codes for digital HDTV," in *Int. Workshop Digital Communications*, Tirennia Italy, Sept. 1993, pp. 237–250.

[5] E. C. Posner, L. L. Rauch, and B. D. Madsen, "Voyager mission telecommunication firsts," *IEEE Commun. Mag.*, vol. 28, pp. 22–27, Sept. 1990.

[6] S. Dolinar and M. Belongie, "Enhanced decoding for the Galileo low-gain antenna mission: Viterbi redecoding with four decoding stages," *JPL TDA Progress Rep.*, vol. 42-121, pp. 96–109, May 1995.

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

[8] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Trans. Commun.*, vol. 44, pp. 1261–1271, Oct. 1996.

[9] L. R. Bahl, J. Cocke, F. Jelinek, and J. Rativ, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284–287, Mar. 1974

[10] C. Berrou, "Some clinical aspects of turbo codes," in *Proc. IEEE Int. Symp. Turbo Codes & Related Topics*, Brest, France, Sept. 1997, pp. 26–31.

[11] C. Berrou, P. Adde, E. Angui, and S. Faudeuil, "A low complexity soft-output Viterbi decoder architecture," in *Proc. IEEE ICC'93*, Geneva, Switzerland, May 1993, pp. 737–740.

[12] J. Hagenauer and P. Hoeher, "A viterbi algorithm with soft-decision outputs and its applications," in *Proc. IEEE GLOBECOM'89 Conf.*, Dallas, TX, Nov. 1989, pp. 1680–1686.

[13] J. Lodge, R. Young, P. Hoeher, and J. Hagenauer, "Separable MAP 'filters' for the decoding of product and concatenate codes," in *IEEE ICC'93*, May 1993, pp. 1740–1745.

[14] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 429–445, Mar. 1996.

[15] P. Elias, "Error-free coding," *IRE Trans. Inform Theory*, vol. IT-4, pp. 29–37, Sept. 1954.

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

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

[18] R. Pyndiah, A. Glavieux, A. Picart, and S. Jacq, "Near optimum decoding of products codes," in *Proc. IEEE GLOBECOM'94 Conf.*, vol. 1/3, San Francisco, CA, Nov.–Dec. 1994, pp. 339–343.

[19] F. J. Macwilliams and N. J. A. Sloane, *The Theory of Error Correcting Codes.* Amsterdam, The Netherlands: North-Holland, 1978, pp. 567–580.

[20] R. G. Gallager, "A simple derivation of the coding theorem and some applications," *IEEE Trans. Inform. Theory*, vol. IT-11, pp. 3–18, Jan. 1965.

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

[22] O. Aitsab and R. Pyndiah, "Performance of Reed-Solomon block turbo codes," in *Proc. IEEE GLOBECOM'96 Conf.*, vol. 1/3, London, U.K., Nov. 1996, pp. 121–125.

[23] R. Pyndiah, "Iterative decoding of product codes: Block turbo codes," in *Proc. IEEE Int. Symp. Turbo Codes & Related Topics*, vol. 1/1, Brest, France, pp. 71–79, Sept. 1997.

[24] J. G. Proakis, *Digital Communications*, 2nd ed. New York: McGraw-Hill, 1989, ch. 2.

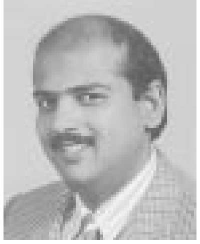
[25] H. Nickl, J. Hagenauer, and F.Burkert, "Approaching Shannon's capacity limit by 0.27 dB using simple Hamming codes," submitted for publication.

[26] A. J. Viterbi, E. Zehavi, R. Padovani, and J. K. Wolf, "A pragmatic approach to treillis-coded modulation," *IEEE Commun. Mag.*, vol. 27, pp. 11–19, July 1989.

[27] R. Pyndiah, A. Picart, and A.Glavieux, "Performance of block turbo coded 16-QAM and 64-QAM modulations," in *Proc. IEEE GLOBECOM'95 Conf.*, vol. 2/3, Nov. 1995, pp. 1039–1044.

[28] P. Adde, R. Pyndiah, O. Raoul, and J. R. Inisan, "Block turbo decoder design," in *Proc. IEEE Int. Symp. Turbo Codes & Related Topics*, vol. 1/1, Brest, France, Sept. 1997, pp. 166–169.

[29] A. Goalic and R. Pyndiah, "Real-time turbo decoding of product codes on a digital signal processor," in *Proc. IEEE Int. Symp. Turbo Codes & Related Topics*, Brest, France, Sept. 1997, pp. 267–270.



**Ramesh Mahendra Pyndiah** (M'95) was born in Mauritius on September 23, 1958. He received the M.S. degree in physics from l'Université de Saint Jérôme, Marseille, France, in 1983, qualified as electronics engineer from l'Ecole Nationale Supérieure des Télécommunications de Bretagne, Brest, France, in 1985, and received the Ph.D. degree in electronics engineering from l'Université de Bretagne Occidentale, France, in 1994.

From 1985 to 1990 he was a Senior Research Engineer at the Philips Research Laboratory (LEP), France, where he was involved in the design of monolithic microwave integrated circuits (MMIC) for digital radio links. In 1991 he was with TRT (Philips Telecommunications Division, France) as Development Project Manager for microwave components. In October 1991 he joined the Signal & Communications Department, l'Ecole Nationale Supérieure des Télécommunications de Bretagne, Brest, France. His current research interests are modulation, channel coding, and joint source channel coding.