On the Error Statistics of Turbo Decoding for Hybrid Concatenated Codes Design

Fulvio Babich and Francesca Vatta

***Abstract*—In this paper we propose a model for the generation of error patterns at the output of a turbo decoder using a *Context Tree* based modelling technique. This model can be used not only to generate the decoder error pattern behaviour with little effort, avoiding simulations, but also to investigate – with no need of performing neither a turbo code distance spectrum analysis, nor the probabilistic characterization of log-likelihood ratios or of the extrinsic information at a turbo decoder output – the performance of hybrid concatenated coding (HCC) schemes having a turbo code as component code. These coding schemes combine the features of parallel and serially concatenated codes and thus offer more freedom in code design. It has been demonstrated, in fact, that HCCs can perform closer to capacity than serially concatenated codes while still maintaining a minimum distance that grows linearly with block length.**

I. Introduction

Turbo codes, first presented in 1993 [2], which are also known as parallel concatenated convolutional codes (PCCCs), are now widely recognised as a highly performing class of concatenated codes. They combine two convolutional codes or more than two (in this latter case they are known as Multiple Parallel Concatenated Codes (MPCCs) [3]) along with one or more pseudorandom interleavers. In particular, the information bits at the input of the first encoder are permuted by the interleaver before entering the second encoder. Thus, the codewords of the parallel concatenated code consist of the information bits followed by the parity check bits of both encoders.

Since their introduction, turbo codes have become the coding technique of choice in many communication and storage systems due to their near Shannon limit error correction capability [4]. These applications include 3GPP, Consultative Committee for Space Data Systems (CCSDS) telemetry channel coding, worldwide interoperability for microwave access (WiMAX), and UMTS, which require throughputs in the range from two to several hundred Mbps.

Manuscript received December 12, 2018; revised April 04, 2019. Date of publication June 06, 2019. Date of current version June 07, 2019. The associate editor Prof. Roberto Garello has been coordinating the review of this manuscript and approved it for publication.

This work has been partially supported by the Italian Ministry of University and Research (MIUR) within the project FRA 2015 (University of Trieste, Italy), entitled "Peer-to-peer millimeter-wave communications in 5G networks: Theoretical modeling and algorithms for massive MIMO systems"

Part of this work ([1]) was presented at the International Conference on Software, Telecommunications and Computer Networks, SOFTCOM '17, Split, Croatia, September 21-23, 2017.

Authors are with the DIA, University of Trieste, Trieste, Italy. E-mails: babich@units.it, vatta@units.it.

Digital Object Identifier (DOI): 10.24138/jcomss.v15i2.669.

Due to the presence of the interleaver, the length of which equals that of the information sequence, k, a maximum-likelihood (ML) sequence decoder would be far too complex for a turbo code since its complexity is related to the number of codewords, 2^k, which becomes huge just for k in the order of some tens, i.e., for moderate or even short values of the interleaver length. Therefore, a suboptimal decoder was proposed in [2], that implements an iterative algorithm which offers near-ML performance, being its central core a Maximum *A Posteriori* (MAP) symbol-by-symbol decoder.

This suboptimum MAP iterative decoding allows to achieve a performance very close to the Shannon limit. However, the corresponding code ensembles are known to be asymptotically bad, namely it is well known that their minimum Hamming distance does not grow linearly with block length [5], [6]. As a result, their minimum distance may not be sufficient to yield very low error rates at moderate-to-high signal-to-noise ratios (SNRs). In particular, it was shown in [7] that they exhibit a bit-error rate floor (the term error floor, for turbo codes, refers to the E_b/N_0 region in which their bit-error rate performance flattens [8]) due to their relatively small free distance.

On the other hand, multiple serially concatenated code (MSCC) ensembles with three or more component encoders can be asymptotically good. This has been shown for repeat multiple accumulate codes in [9] and [10]. There also exist variations of standard repeat accumulate codes that are asymptotically good [11] but are more complex to encode than classical repeat accumulate codes. MSCCs, in general, exhibit good error floor performance due to their large minimum distance, but they have the drawback of converging at an SNR further from capacity than parallel concatenated codes.

An alternative to the above schemes are hybrid concatenated codes (HCCs), first introduced in [12]. The hybrid structure shown in [12] includes a PCCC or turbo code C_p with rate $R_c^p = k/n_1$, an outer code C_o with rate $R_c^o = k/p$, and an inner code C_i with rate $R_c^i = p/n_2$. This gives a HCC with overall rate $R_c = k/(n_1 + n_2)$. The analysis in [12] is based on analytical performance bounds, based on the input-output weight coefficients of the codes involved in the analysis, averaged over all possible interleavers (i.e., performing a uniform interleaver analysis). With the objective of designing an HCC structure, a spectral based analysis can be found also in [13], where the distance spectrum analysis is performed hypothesizing a rectangular interleaver, and in [14], where it is observed that "Turbo codes inherently provide unequal error protection, because only certain bit positions are affected by the dominant error events, as determined by

the interleaving mapping". HCCs combine the features of parallel and serially concatenated codes and thus offer more freedom in code design. It has been demonstrated in [16], that HCCs can be designed that perform closer to capacity than MSCCs while still maintaining a minimum distance that grows linearly with block length. In particular, small memory-one component encoders are sufficient to yield asymptotically good code ensembles for such schemes. The resulting codes provide low complexity encoding and decoding and, in many cases, can be decoded using relatively few iterations.

In the cited works on HCCs ([12], [13], and [14]), and in [15], the turbo decoder output statistical characterization is accomplished through a distance spectrum analysis, accurately predicting the behavior of the iterative decoder for large interleaving depth and for moderate to high E_b/N_0, and validated by means of simulation. Other methods of statistical characterization of the turbo decoder output include the probabilistic characterization of the input-output signals in a turbo decoder, expressed using log-likelihood ratios (LLRs), when the received signal is subject to additive white Gaussian noise (AWGN) [17], or in more complicated communications systems, such as cognitive relay systems [18], and the extrinsic information transfer (EXIT) chart analysis, allowing the prediction of turbo cliff position and bit error rate after an arbitrary number of iterations [19].

Even if never proposed in the literature, the turbo decoder output could be directly modelled following the correlation characteristics of the process given by the bit errors at the turbo decoder output itself. This technique has the advantage of providing a quite accurate analytical tool to get an insight into a PCCC residual bit error probability (BER) or frame error probability (FER) performance in any of the three regions characterizing this performance (see Section IV, in which this argument has been addressed), unlike, e.g., the distance spectrum analysis, accurately predicting the behavior of the iterative decoder for moderate to high E_b/N_0, i.e., in the error floor region, or the EXIT chart analysis, allowing the prediction of turbo cliff position, and thus providing an insight into a PCCC BER or FER performance in the pinch-off and waterfall regions. To model the turbo decoder output, from the viewpoint of the process given by the bit errors at the turbo decoder output itself, first and second order Markov assumptions could be hypothesized, but using these models could constitute an excessive simplification, since the order of the model is completely unknown. Thus, we have decided to consider more complex models, allowing to follow better the correlation characteristics of the process, while being possibly easily inferable from available experimental data. A class of models that meet these requirements are the models based on the *Context Tree (CT)* [21]. They had so far found application in data compression problems (see [22]), in equalization problems (see [23]), and mobile channel modelling problems (see [24]).

Using a CT based modelling technique, in this paper we obtain a model for the generation of error patterns at the output of a turbo decoder that can be used not only to generate the output sequence with little cost, avoiding simulations, but also to investigate – with no need of performing neither a turbo code distance spectrum analysis, nor the probabilistic characterization of LLRs or of the extrinsic information at a turbo decoder output – the performance of HCC schemes having a turbo code as component code (see, for instance, [25] and [26]).

This paper is organised as follows: in Section II we recall some basic concepts of the principal Markov assumptions, whereas in Sections II-A and II-B we give the definition of the Context Tree (CT) and the description of the Context Tree Pruning (CTP) algorithm, on which our modelling is based, respectively. In Section III-A we briefly review the basic concepts of turbo codes iterative decoding, whereas in Section III-B we show the application of the CTP algorithm to a turbo decoder output, presenting a model for the generation of error sequences with the same statistical characteristics as those output from a turbo decoder. In Section IV we show the resulting model, considering a particular turbo code with given rate, states number and interleaver length. In Section V, to validate the model, we show that it is well suited to represent the error patterns at the output of a turbo decoder, through the performance analysis of a serial concatenated scheme having the turbo code considered in the previous section as inner code. This analysis shows a very good agreement between the performance of the fitted model and simulation results. Finally, Section VI states our conclusions.

II. Modelling Error Statistics

In probability theory and statistics, the term *Markov property* refers to the memoryless property of a stochastic process. It is named after the Russian mathematician Andrey Markov [20]. A stochastic process has the Markov property if the conditional probability distribution of future states of the process (conditional on both past and present states) depends only upon the present state, not on the sequence of events that preceded it.

Given a sequence of observations $(x_1, x_2, \ldots, x_n)$, if we can make the simplifying assumption that the probability of an observation at time n only depends on the observation at time $n-1$, i.e.,

$$P(x_n|x_{n-1}, x_{n-2}, \ldots, x_1) \approx P(x_n|x_{n-1}) \tag{1}$$

we can call this assumption a *first-order* Markov assumption or simply *Markov assumption*. We can also express the joint probability using this assumption:

$$P(x_1, x_2, \ldots, x_n) \approx \prod_{i=1}^{n} P(x_i|x_{i-1}) \tag{2}$$

A *second order* Markov assumption would have the observation at time n depend on the observations at times $n-1$ and $n-2$, i.e.,

$$P(x_n|x_{n-1}, x_{n-2}, \ldots, x_1) \approx P(x_n|x_{n-1}, x_{n-2}) \tag{3}$$

A. The Context Tree Definition

A Context Tree (CT) is a minimum parametric representation of a Markov chain. Consider a sequence with values in the alphabet $\mathcal{A}$, $x^n = \{x_k\}_{k=1}^n \in \mathcal{A}^n$, with $x_k \in \mathcal{A}$. Let $P(x^n)$

be the probability function associating to each sequence its probability value. Defining $P(x|x^n) = \frac{P(x^n x)}{P(x^n)}$, let's assume that there exists a *state function* $s(\cdot)$:

$$s(x^n) : \mathcal{A}^n \longmapsto S \tag{4}$$

such that

$$P(x|x^n) = P[x|s(x^n)],\ \forall x \in \mathcal{A}, \forall x^n \in \mathcal{A}^n, \forall n \tag{5}$$

Let's define this function as the *suffix*, or *context*, of the sequence x^n:

$$s(x^n) = \{x_{n-k}\}_{k=1}^{m(x^n)},\ 1 \leq m(x^n) \leq n \tag{6}$$

where $m(x^n)$ may *not* be the same value for each sequence[1]. Let's define Context Tree the couple CT: $\{\mathcal{C}, \Theta\}$, where $\mathcal{C} = \{s_k\}_{k=1}^{|\mathcal{C}|}$ and $\Theta = \{P(x|s)\}_{s\in\mathcal{C}, x\in\mathcal{A}}$. The CT model induces a P_{CT} probability function defined as:

$$P_{CT}(x^n) = \prod_{t=0}^{n-1} P[x_{t+1}|s(x^t)] \tag{7}$$

so that it defines completely the statistical behavior of the sequence x^n.

This model can be easily represented in terms of a tree (hence the name) as shown in Fig. 1, where $\mathcal{A} = \{a, b, c\}$ and $\mathcal{C} = \{a, ba, bb, bca, bcb, bcc, c\}$. Each leaf of the tree is a context s and the bar chart within each leaf represents the set $\{P(x|s)\}_{x\in\mathcal{A}}$. Note that by completing the tree in such a way that it is balanced, by replacing each added leaf with the leaf in the parent node, we obtain a Markov model of order equal to the depth of the tree d.

A first-order Markov model is associated with each CT model $M_{CT} = \{\mathcal{S}_M, P_M\}$, where $\mathcal{S}_M$ is the space of the states and P_M is the transition matrix.

Define $\mathcal{S}_M = \{s'_k\}_{k=1}^{|\mathcal{S}_M|}$, where s'_k is a set of symbols of the alphabet such that $s'_k = \{s'_k(i)\}_{i=1}^{|s'_k|}$, $s'_k(i) \in \mathcal{A}$. $\mathcal{S}_M$ must be chosen in such a way that, first of all, it can be partitioned as $\mathcal{S}_M = \cup_{h=1}^{|\mathcal{C}|}\mathcal{S}_h$, with each $\mathcal{S}_h = \{s'_{h,n}\}_{n=1}^{|\mathcal{S}_h|}$ satisfying the following condition

$$\{s'_{h,n}(i)\}_{i=1}^{|s_h|} = s_h,\ \forall n \tag{8}$$

This condition means that the elements of $\mathcal{S}_h$ are extensions of the suffix $s_h \in \mathcal{C}$. Furthermore, the choice of these extensions must be made in such a way that, if we set

$$v = (xs'_k),\ x \in \mathcal{A} \tag{9}$$

and

$$f(s'_k, x) = \{v(i)\}_{i=1}^{m(v)},\ 1 \leq m(v) \leq |v| \tag{10}$$

it is[2]

$$f(s'_k, x) \in \mathcal{S}_M,\ \forall k,\ \forall x \in \mathcal{A} \tag{11}$$

At this point, the P_M matrix is immediately determined: in fact, given the definition of $\mathcal{S}_M$,

$$\begin{aligned} &P_M(j,k) = P(s_{n+1} = s'_k | s_n = s'_j) = 0, \\ &\text{if } \nexists x \in \mathcal{A} : f(s'_j, x) = s'_k \end{aligned} \tag{12}$$

[1]Note that this does not define a finite state machine, as, in general, it may not be true that there exists a function f such that $s(x^{n+1}) = f[s(x^n), x_{n+1}]$.

[2]Note that, in fact, a Markov model is a finite state machine.

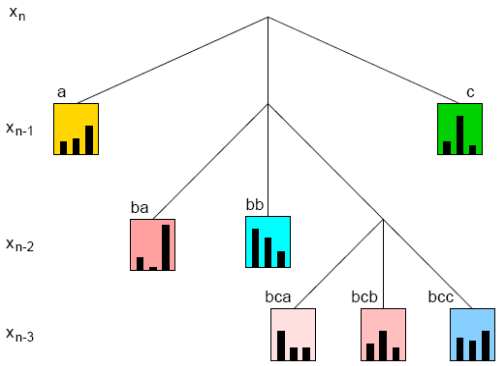

Fig. 1. Example of type CT model.

and

$$\begin{aligned} &P_M(j,k) = P(\bar{x}|s_l(s'_j)), \\ &\text{if } \bar{x} \in \mathcal{A} : f(s'_j, \bar{x}) = s'_k \end{aligned} \tag{13}$$

being $s_l(s'_j)$ the suffix corresponding to the partition of $\mathcal{S}_M$, $\mathcal{S}_l(s'_j) \ni s'_j$.

It is straightforward to prove that $|\mathcal{C}| \leq |\mathcal{S}_M|$ and $|\Theta| \leq |P_M|(|P_M| - 1)$, where the second quantity is the minimum number of parameters that univocally describe P_M. In the case shown in Fig. 1, $\mathcal{S}_M = \{a, ba, bb, bca, bcb, bcc, ca, cb, cc\}$.

B. The Context Tree Pruning (CTP) Algorithm

The *Context Tree Pruning (CTP)* algorithm (see, e.g., [24]) is the algorithm used to estimate, starting from a known sequence $\mathbf{x} = x^n$, $x \in \mathcal{A}$, called *training sequence*, originated from a source with unknown properties, the CT model that generated it. Therefore, it allows estimating a Markov model of unknown order and is, with respect to this kind of estimate, an optimal algorithm in the sense of complexity of the model (model with minimum description length) [21]. Namely, given the training sequence, the CTP algorithm is optimal in estimating the best fitting CT model $\{\mathcal{C}, \Theta\}$ when a maximum order M is assigned *a priori*, and an immediate conversion from this model to a Markov model is possible.

The recursive algorithm described in [21] allows to build a complete CT of maximum depth M. It provides two rules:

1) Initialize the root with its symbol counts all zero. Recursively, having built the tree, possibly incomplete, $\mathcal{T}_t$ at step t (from x^t, $1 \leq t < n$), read the symbol x_{t+1}. Follow the tree according to the sequence defined by $x_t x_{t-1} \cdots$, and increment by one the count of symbol x_{t+1} for every node visited, up to the deepest node, say $x_t \cdots x_{t-j+1}$;
2) if the last updated count becomes at least 2, create a new node $x_t \cdots x_{t-j}$, and initialize its symbol counts to zero, except for the symbol x_{t+1}, whose count is set to 1: this completes the construction of the new tree $\mathcal{T}_{t+1}$.

The basic structure of the CTP algorithm is the following:

1) Following the recursive algorithm described above, build a complete tree of maximum depth M, chosen a priori[3],

[3]M must be large enough to include all models of interest.

from the training sequence $\mathbf{x}$, identifying a set of possible contexts, $\tilde{\mathcal{C}}_{\mathbf{x}} \supseteq \mathcal{C}_{\mathbf{x}}$, and estimating the conditional probabilities $P_{\mathbf{x}}(x|s)$, $x \in \mathcal{A}$, $s \in \tilde{\mathcal{C}}_{\mathbf{x}}$, by measuring the relative frequency of transitions:

$$P_{\mathbf{x}}(x|s) = \frac{n_{\mathbf{x}}(x,s)}{\sum_{y\in\mathcal{A}} n_{\mathbf{x}}(y,s)},\ \forall x \in \mathcal{A}, |s| \le M \quad (14)$$

with $n_{\mathbf{x}}(x,s) := \sum_{i=1}^{n} \mathbf{1}[x_i = a, \sigma(x_{i-1}\cdots x_{i-M}) = s]$, where $\sigma(\cdot)$ is the suffix operator (to distinguish it from the suffix s), and $n_{\mathbf{x}}(s) = \sum_{x\in\mathcal{A}} n_{\mathbf{x}}(x,s)$.

2) Generate the *test statistics* (based on Kullback-Leibler distance $D(p||q)$[4]) for each child context node, where if s is a father context, sy, with $y \in \mathcal{A}$, is its child context:

$$\Delta(sy) = \sum_{x\in\mathcal{A}} n_{\mathbf{x}}(x,sy)\log\frac{P_{\mathbf{x}}(x|sy)}{P_{\mathbf{x}}(x|s)} = n_{\mathbf{x}}(sy)D(sy||s) \quad (15)$$

where $\log(\cdot)$ is the base 2 logarithm.

3) *Prune* the tree keeping the nodes that satisfy the condition:

$$\Delta(sy) \ge T_p = 2(|\mathcal{A}|+1)\log(n+1) \quad (16)$$

with $|s| \le \min\left[\frac{\log n}{\log|\mathcal{A}|}, M\right]$.

4) *Complete* the tree in the sense that, if in a set of child nodes of the same father node one is maintained, the others are also maintained. In other words, if one child of a parent is retained, then all the children of that parent are retained. This means that if a node is retained (i.e., not pruned), then every node on the path from that node to the root is retained. In this way, the set of contexts $\mathcal{C}_{\mathbf{x}} = \{s_k\}_{k=1}^{|\mathcal{C}_{\mathbf{x}}|}$ is defined together with the operator $\sigma(\cdot)$.

5) Define the probability system $P_{\mathbf{x}}(x^n)$:

$$P_{\mathbf{x}}(x^n) = \prod_{t=0}^{n-1} P_{x^t}(x_{t+1}|s_t) \quad (17)$$

where $P_{x^t}(a|s_t)$ may be computed as:

$$P_{x^t}(a|s_t) = \frac{n_{x^t}(a,s_t) + \frac{1}{2}}{\sum_{a\in\mathcal{A}} n_{x^t}(a,s_t) + \frac{|\mathcal{A}|}{2}} \quad (18)$$

where $s_t = \sigma(x^t)$ and $n_{x^t}(\cdot)$ is the frequency measured through x^t.

If the source is not a CT source, when M and n are increased towards infinity the algorithm estimates a sequence of trees approximating the source with increasing detail (theoretically, a CT with infinite depth would be required).

III. Modelling the Turbo Decoder Output

A. The Sum-product Algorithm

The sum-product algorithm is the basic "decoding" algorithm for codes on graphs. For finite cycle-free graphs, it is finite and exact. However, because all its operations are local, it may also be applied to graphs with cycles; then it becomes iterative and approximate, but in coding applications it often works very well. It has become the standard decoding algorithm for capacity-approaching codes, e.g., turbo codes and Low Density Parity Check (LDPC) codes.

There are many variants and applications of the sum-product algorithm. The most straightforward application is to a Maximum a Posteriori (MAP) decoding. In fact, the aim of the sum-product algorithm is to approximate MAP decoding, or equivalently to compute the a posteriori log-likelihoods of the individual transmitted bits given the received vector. The MAP decoding algorithm for the constituent convolutional codes can be implemented with the well known forward-backward or BCJR [28] algorithm (named after its inventors: Bahl, Cocke, Jelinek and Raviv, which is feasible in this case because these codes have a short constraint length).

Let us consider a block or convolutional encoder described by a trellis, and a sequence $\mathbf{x} = (x_1, x_2, \ldots, x_L)$ of L n-bit codewords or symbols at its output, where x_k is the symbol generated by the encoder at time k. The corresponding information or message input bit, u_k, can take on the values -1 or $+1$ with an *a priori* probability $P(u_k)$, from which we can define the so-called *log-likelihood ratio* (LLR):

$$L(u_k) = \log\frac{P(u_k=+1)}{P(u_k=-1)} \quad (19)$$

Given an a priori estimate on each information bit and an LLR for each transmitted bit[5], the BCJR algorithm outputs the correct a posteriori LLR for each information bit, a real number defined by the ratio

$$L(u_k|\mathbf{y}) = \log\frac{P(u_k=+1|\mathbf{y})}{P(u_k=-1|\mathbf{y})} \quad (20)$$

where it was supposed that the coded vector $\mathbf{x}$, transmitted over a memoryless additive white gaussian noise (AWGN) channel, was received as a real vector $\mathbf{y}$. The soft information provided by $L(u_k|\mathbf{y})$ can then be transferred to another decoding block, if there is one, or simply converted into a bit value through a hard decision.

A turbo decoder consists of two single soft-in soft-out (SISO) decoders that work iteratively. The output of the first (upper decoder) feeds into the second to form a turbo decoding iteration. Interleaver and deinterleaver blocks re-order data in this process. The turbo decoding algorithm iterates between the MAP decoders corresponding to the two constituent codes. The received values corresponding to the systematic bits are used to initialize the a priori LLR's for the information bits. One of the constituent decoders then outputs the a posteriori LLR's by running the BCJR algorithm, the idea being to use these as a priori LLR's for the other decoder. However, in order not to form short loops in the so-called "computation tree," the difference between the a posteriori and the a priori LLR's (this is known as extrinsic information) is fed to the other decoder as a priori LLR's, and the same operation is repeated over and over again. Various stopping rules are used to decide on convergence and guard against limit-cycles (see, e.g., [29], where their categorization into three main classes has been addressed).

[4] If $p(x)$ and $q(x)$ are two probability functions on X, with values $x \in \mathcal{X}$, $D(p||q)$ is the Kullback-Leibler distance, i.e., the relative entropy [27]: $D(p||q) = \sum_{x\in\mathcal{X}} p(x)\log\frac{p(x)}{q(x)} = E_p\{\log\frac{p(X)}{q(X)}\}$

[5] This log-likelihood ratio is zero with equally likely input bits.

B. Application of the CTP to the Turbo Decoder Output

It is well known that the bit errors at the output of a turbo decoder are not independent but rather tend to group in *error bursts* (see, e.g., [13] and [14]). To obtain a model for the generation of error sequences with the same statistical characteristics as those output from a turbo-decoder, we apply the CTP algorithm described in Section II-B by splitting the whole error process into two parts:

1) a binary process β_i, modelling the block error process, such that:
$$\beta_i = \begin{cases} 1 & \text{if the } i\text{-th data block is in error} \\ 0 & \text{otherwise} \end{cases} \quad (21)$$
where the block length is assumed equal to the interleaver length and a block is in error if it presents at least one error bit;
2) a process modelling the number of error bits j within a block.

Consider an error sequence x^n with values in the alphabet $\mathcal{A} = \mathrm{GF}(2)$, $x^n = \{x_k\}_{k=1}^n \in \mathcal{A}^n = \mathrm{GF}(2^n)$, with $x_k \in \mathrm{GF}(2)$. Let $P(x^n)$ be the probability function associating to each sequence its probability value.

Let's assume that there exists a *state function* $f(\cdot)$:
$$f(x^n) : \mathrm{GF}(2^n) \longmapsto \mathcal{N} \quad (22)$$
where $\mathcal{N}$ denotes the set of natural numbers. The context $z(t)$ of the symbol $x(t)$, immediately following the past string $x^n = x_1 \cdots x_n$, is the class of strings $(x^n)' = x'_1 \cdots x'_n$ such that $f((x^n)') = f(x^n)$, namely, the context of $x(t)$ is some computable function of the past string. In other words,
$$P(x(t)|x^n) = P[x(t)|f(x^n)],\ \forall x \in \mathrm{GF}(2), \forall x^n \in \mathrm{GF}(2^n) \quad (23)$$
and the function (22) is definable as the *suffix*, or *context*, of the sequence x^n. One way of specifying this function is by a finite state machine as done in [21].

In order to follow the 5 steps of the CTP algorithm given in Section II-B, first of all a complete tree of maximum depth M, chosen a priori and large enough to include all models of interest, has to be built from the training sequence $\mathbf{x}$. The idea is to grow two binary trees, one for the case where the current symbol $x(t)$, which we denote by u, has the value 0, and the other when it has the value 1. We are interested in the intersection of the two trees, which we actually generate in the following way, directly applying the recursive algorithm given in [21]:

1) Declare the context tree of the first symbol $x(1)$ in the string to be the 1-leaf tree $T(0)$, where the only node, the root, is marked with the pair of counts $(c(0,\lambda), c(1,\lambda)) = (1,1)$, having defined the empty string $\lambda = x_0$ as in [21].
2) Proceeding recursively, let $T(t-1)$ be the last constructed tree with $(c(0,z), c(1,z))$ denoting the pair of the counts at the generic node z. After the next symbol $u = x(t)$ is observed, generate the next tree $T(t)$ as follows: climb the tree $T(t-1)$, starting at the root and taking the branch, left for 0 and right for 1, indicated by each of the successive symbols in the past sequence $\sigma(x(1)\cdots x(t-1)) = z_1 z_2 \cdots$, being $\sigma(\cdot)$, the suffix operator, the same as that defined in [21]. For each node z visited, increment the component count $c(u,z)$ by one. Continue until a node w is reached whose count $c(u,w) = 1$ before the update.
3) If w is an internal node with node $w0$ as the left and $w1$ as the right successor, increment the component counts $c(u,w0)$ and $c(u,w1)$ by one. Define the resulting tree to be $T(t)$. If, again, w is a leaf, extend the tree by creating two new leaves $w0$ and $w1$. Assign to both leaves the same counts: $c(u,w0) = c(u,w1) = 1$ and $c(u',w0) = c(u',w1) = 0$, where u' denotes the opposite symbol to u. Call the resulting tree $T(t)$.

This completes the description of the recursive algorithm called "Context" given in [21] when applied to binary sequences.

The basic structure of the CTP algorithm applied to binary error sequences is the following:

1) Build a complete tree from the training sequence $\mathbf{x}$ as described above, and then estimate the conditional probabilities (23) by measuring the relative frequency of transitions (14).
2) Generate the *test statistics* (15) for each child context node.
3) Prune the tree keeping the nodes that satisfy the condition (16).
4) Complete the tree in the sense that, if one child of a parent is retained, then all the children of that parent are retained. This means that if a node is retained (i.e., not pruned), then every node on the path from that node to the root is retained.
5) Define the probability system $P_{\mathbf{x}}(x^n)$ as in (17).

Applying this algorithm, first, we have analysed the properties of the binary process β_i in order to investigate about blocks interdependence: the memory of this process has been obtained from long error sequences. Each of these can constitute the training sequence $\mathbf{x}$, which allows to evaluate a CT model using the above described CTP algorithm. Second, we have analysed the properties of the error bits number j within a block.

It should be noted that, due to the nature of the CTP, as the length N of the sequence increases, CT models of increasing complexity may be obtained. If the source is really describable with a CT model, then there exists an N above which the complexity of the model gets stable, otherwise the complexity of the model will grow indefinitely with N. This means that, by increasing the length of the pilot sequence, the models obtained approximate the source more accurately.

IV. The Resulting Model

The basic turbo codes are made up with two parallel concatenated recursive convolutional (RSC) encoders, separated by an interleaver. The upper encoder RSC1 is a systematic rate-1/2 recursive convolutional encoder that generates a systematic codeword which consists of the information bit u_k followed by the parity bit x_{k1}. The lower encoder RSC2 is a non-systematic rate-1 recursive convolutional encoder that

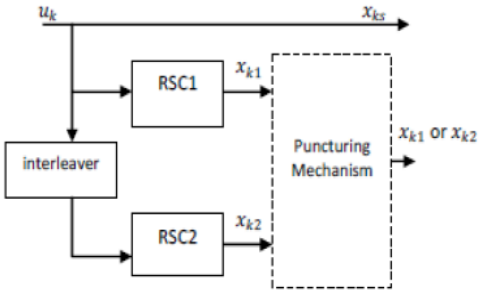

Fig. 2. Turbo encoder block diagram.

generates a non-systematic codeword which consists of the parity bit x_{k2} only. A typical structure of a rate-1/2 turbo code (the whole scheme including the puncturing mechanism) and of its rate-1/3 mother code (the whole scheme excluding the puncturing mechanism) is shown in Fig. 2. The configuration is assumed to be symmetrical, i.e., the constituent encoders are assumed to have the same constraint length and generator polynomials: this means that the rate-1 lower encoder is simply the non-systematic version of the upper rate-1/2 encoder, obtained eliminating from the latter the systematic bit u_k.

We consider a rate-1/3, four state turbo code, obtained by parallel concatenating one rate-1/2 systematic convolutional code with generator polynomials $(1, 5/7)$ in octal and its non-systematic rate-1 version with generator polynomials $(5/7)$ in octal. As explained in [30], the PCCC can work in three co-decoding configurations, namely in conjunction with continuous, trellis truncated and trellis terminated co-decoding:

1) The first alternative consists of keeping both constituent encoders in the states they were in at the end of the previous block and starting the encoding process of the new block from those states. Thus, since the trellis of each encoder will evolve in a continuous fashion, this first operation mode is called *continuous co-decoding* (see Fig. 2(a) in [30]).
2) The second alternative resets the states of both encoders at the end of each block, so that, when the encoded sequences of both decoders end at states different from the zero state. Since there will be a discontinuity at the end of each block, this second operation mode is called *truncated co-decoding* (see Fig. 2(b) in [30]).
3) The third alternative, called *terminated co-decoding* consists in terminating the constituent convolutional codes trellises at the end of each block (see Fig. 2(c) in [30]). Various ways for terminating the trellises have been suggested in the literature, including ad hoc designed interleavers [31], [32]. The simplest, yet general way [3], adopted also in this paper, requires the transmission of extra data symbols, corresponding to the encoded versions of a number of information bits, for each constituent convolutional code, equal to the constraint length ν of the constituent convolutional codes. Thus, the overall PCCC rate is reduced; for example, in the case of a rate 1/3 PCCC, the new rate is $N/3(N+\nu)$.

In [30] it was shown that the performance of the truncated encoder are significantly worse than those of the continuous one, whereas trellis termination is only slightly worse (see Fig. 3 in [30]). However, this is not the only difference. In fact, as it can be argued observing Fig. 2 in [30], in all cases the information sequence is split into blocks of N bits (N being the length of the interleaver used by the turbo code), that are encoded by the first constituent encoder and, after interleaving, by the second encoder. But, in the first operation mode, both constituent encoders work in a continuous fashion, whereas in the second, at the end of each block, both constituent encoders are simply reset. In the third mode, finally, the operation is similar to the second, but, instead of trellis truncation, at the end of each block a suitably chosen sequence of bits is appended to the information block in order to terminate the trellises of both constituent codes.

In this paper, as said above, trellis termination has been performed as in [3]. Given this choice, the binary process β_i modelling the block error process and described by (21), whose properties have been analysed applying the CTP algoritm described in Section III-B in order to investigate about blocks interdependence, may be expected to be describable through a memoryless model. However, its analysis has been anyway performed applying the CTP algorithm in order to give a general value to the proposed procedure, in the sense that for the second and third PCCC co-decoding configurations described above (namely, trellis truncation and trellis termination) the binary process β_i may be expected to be describable through a memoryless model, but it may be easily inferred that the binary process β_i will not have the same characteristics when the first PCCC co-decoding configuration (namely, continuous co-decoding) is assumed. For simplicity we assume that the turbo code interleaver is a random interleaver. We use the iterative Maximum A Posteriori (MAP) decoding algorithm described in [2]. The turbo-decoder is assumed to know the final states of the constituent encoders.

A rate-k/n turbo code may be obtained from this mother code with appropriate puncturing matrix. Consider, for instance, a rate-1/2 turbo code obtained from the above mentioned mother code with puncturing period 2. Assume an interleaver length $L = 1024$. The CTP confirms that error blocks are independent. Thus, the binary process of the block (of length L) successes and failures, defined by (21), can be modelled, as expected, by means of a memoryless model: this model gives always good statistical fit with the true data source.

Typically, bit-error rate charts of iterative decoding schemes can be divided into three regions (shown in Fig. 4, as far as the turbo codes considered in [33] are concerned) [19]:

1) The region of low E_b/N_0 with negligible iterative BER reduction, which can also be referred to as the pinch-off region, with the decoder transfer characteristics intersecting at low mutual information (corresponding to high BER) and the trajectory getting stuck.
2) The turbo cliff region (also known as waterfall region) with persistent iterative BER reduction over many iterations, which can also be referred to as the bottleneck region, with the decoding trajectory just managing to sneak through a narrow tunnel. Convergence toward low

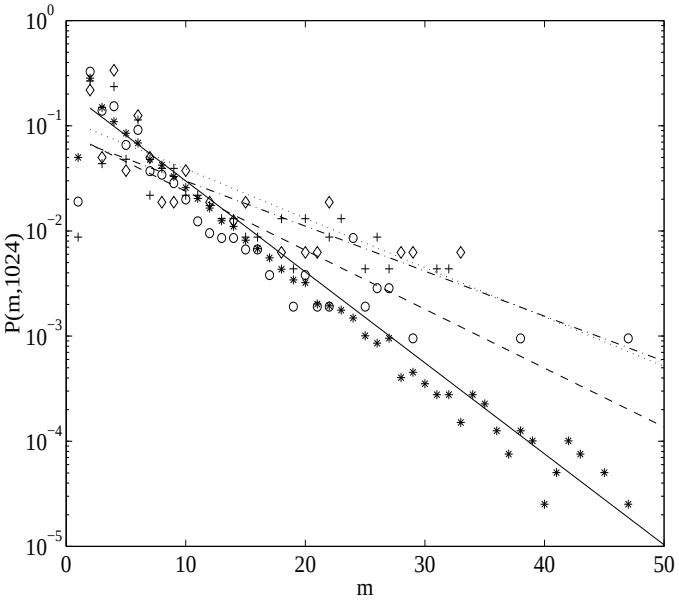

Fig. 3. Probability of having m errors in a wrong block of length $L = 1024$ at $E_b/N_0 = 2$ dB with $I = 2$ iterations ($*$), $I = 4$ iterations ($\circ$), $I = 7$ iterations ($+$) and $I = 10$ iterations ($\diamond$). The linear regression curve related to $I = 2$ is shown (solid line) together with those related to $I = 4$ (dashed line), to $I = 7$ (dash-dotted line), and to $I = 10$ (dotted line).

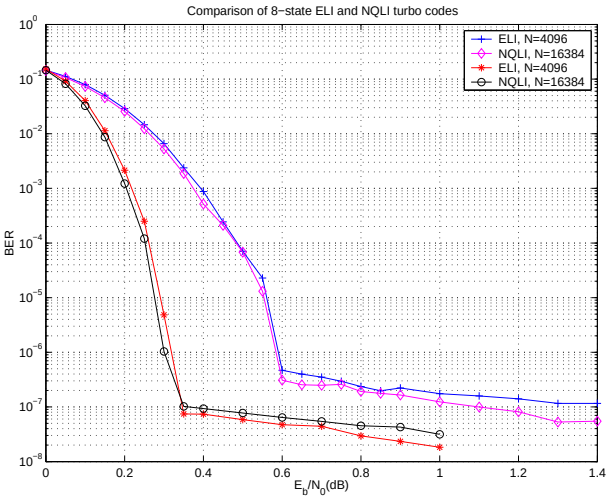

Fig. 4. Simulated BER performance for rate-1/3 turbo codes with 30 iterations and interleaver length N on the AWGN channel. The constituent codes are those listed in Table III(a) of [33].

BER is slow but possible, since both decoder transfer characteristics do not intersect anymore.

3) The BER floor region for moderate to high E_b/N_0, where a rather low BER can be reached after just a few iterations, which can also be referred to as the wide-open region with fast convergence.

Likewise, in [19] the acronym EXIT was explained as the opportunity to see at what E_b/N_0 the decoding trajectory succeeds in exiting the pinch-off region through the bottleneck to converge toward low BER. For rate-1/2 PCCCs, the pinch-off limit was shown in [19] to be at 0.69 dB, whereas we found the bottleneck limit to fall at 2 dB, as was also shown in [34]. Thus, since at $E_b/N_0 = 2$ dB falls the beginning of the BER error floor region for the rate-1/2 code considered, this E_b/N_0 was chosen in Fig. 3 to show the error number in each wrong block, with respect to the iterations number I.

TABLE I
BLOCK ERROR PROBABILITIES

I	a	b	Error probability
2	0.090	0.697	3.9705×10^{-1}
4	0.056	1.062	1.0510×10^{-2}
7	0.043	1.098	2.2900×10^{-3}
10	0.047	0.941	1.6000×10^{-3}

It can be observed that $P(m, L)$, i.e., the probability of having m errors in a wrong block of length $L = 1024$, depends on I and is approximately given by:

$$P(m, L) = 10^{-(am+b)} \tag{24}$$

i.e., it is linear in logarithmic coordinates. The approximate coefficients a and b have been determined by means of linear regression curves for each iterations number I and are shown in Table I.

As shown in Fig. 3, the probability of having m errors in a wrong block of length L depends on I, but the linear regression curves obtained for the different I values intersect each other or are very close to each other for low values of the number of errors within a wrong block m, i.e., approximately for $0 < m < 15$. Outside this interval, i.e., for $m \geq 15$, the linear regression curves obtained for $I > 2$ are all lower bounded by the linear regression curve obtained for $I = 2$. Thus, we can keep the latter as a good approximation of $P(m, L)$, $\forall I$, for low values of m (less than about 15), and as a lower bound on $P(m, L)$, $\forall I$, for higher m values. This fitted model will work perfectly for $I = 2$, and approximately well for $I > 2$ and low values of m. For higher m values, the linear regression curve obtained for $I = 2$ will give a lower bound on the bit error performance obtained for $I > 2$.

Keeping the focus on the error floor region, since at $E_b/N_0 = 2$ dB falls its beginning, with fast convergence to rather low BER just after a few iterations, we found that at $E_b/N_0 > 2$ dB the error number in each wrong block, with respect to the iterations number I, presents the same distribution shape as that found at $E_b/N_0 = 2$ dB, but vertically shifted towards lower $P(m, L)$ values.

Therefore, in conclusion, the error statistics of iteratively decoded turbo codes may be approximated by using a generator that adopts a two level model:

1) a memoryless generator, modelling the binary process of the length-L blocks of successes and failures;
2) an error generator uniformly distributing m errors within each length-L block (with m chosen according to (24)).

The relative frequency of blocks in error decreases with the iterations number I, as shown in Table I, but within the wrong blocks the error distribution is always approximated (for lower

m values) or lower bounded (for higher m values) by (24) with the values of a and b selected for $I = 2$.

This model highlights that errors appear in bursts, but the error densities in the bursts are very low, as may be simply seen looking at the results reported in Fig. 3. It is a fact that the burst characterization would require, in principle, a much deeper analysis aimed to investigate the errors correlation properties, but the linear regression curves we found already provide some information in this sense. In fact, it should be noticed that there is a close relationship between the correlation coefficient and the slope of a regression line, since, when using the ordinary least squares method, as done in this case, the correlation coefficient r is the slope of the regression line of the standardized data points, and represents a measure of the amount of agreement between the variable m and $P(m, L)$. When r is positive, the correlation is positive, which means that high values of one variable correspond to high values of the other. Conversely, if r is negative, as in this case, then the correlation is negative: low values of one variable correspond to high values of the other. An important property of r is that $-1 \leq r \leq 1$: the values $r = \pm 1$ correspond to a perfect correlation, the value $r = 0$ to the absence of correlation. As may be seen from Fig. 3 and Table I, the absolute values of the slopes of the linear regression curves are all very small and decrease, in absolute value, with the iterations number I. Thus, the strongest, but however weak, correlation is encountered with $I = 2$ and the weakest with $I = 10$. This suggests that, for AWGN channels, a Bose-Chaudhuri-Hocquenghem (BCH) code, correcting a number of independent bits, is sufficient to lower the BER floor typical of turbo codes performance, instead of using a Reed-Solomon code as in traditional concatenated schemes.

V. Performance Analysis of a BCH-turbo Code Serial Concatenation

On the basis of the statistical feature given by (24), a serial concatenation scheme, consisting of a BCH outer code and a turbo inner code, can be designed.

The BCH codes form a class of cyclic error-correcting codes that are constructed using polynomials over a finite field (also called Galois field). One of the key features of BCH codes is that, during code design, there is a precise control over the number of symbol errors correctable by the code. In particular, it is possible to design binary BCH codes that can correct multiple bit errors. Another advantage of BCH codes is the ease with which they can be decoded, namely, via an algebraic method known as syndrome decoding. This simplifies the design of the decoder for these codes, using small low-power electronic hardware.

Although the exact expression of the residual BER of the concatenated scheme is very difficult to obtain, there is a simple upper bound valid for a t-error-correcting BCH code of length n [35]. The bound is given by the following expression:

$$P_b(e) \leq \sum_{j=t+1}^{n} \frac{\min(j+t, n)}{n} P(j, n) \qquad (25)$$

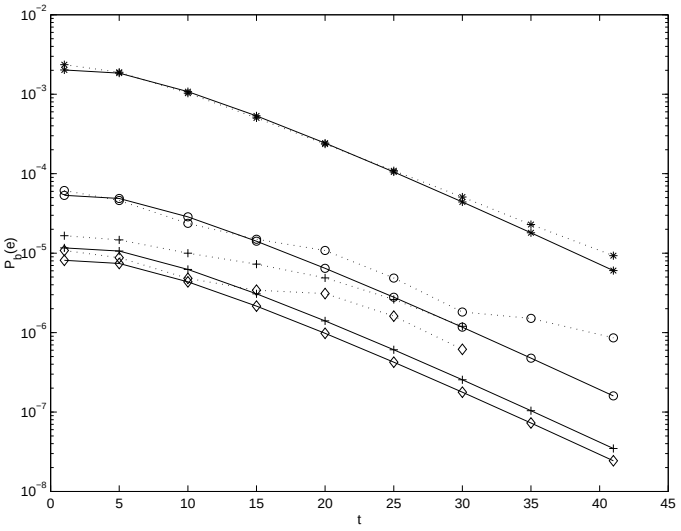

Fig. 5. Residual bit error probability after decoding of the concatenated coding scheme versus the error correction capability t of the BCH(1023,k) outer code, with $I = 2$ iterations ($*$), $I = 4$ iterations ($\circ$), $I = 7$ iterations ($+$) and $I = 10$ iterations ($\diamond$). The solid curves are related to the performance of the fitted model, whereas the dotted curves show simulation results.

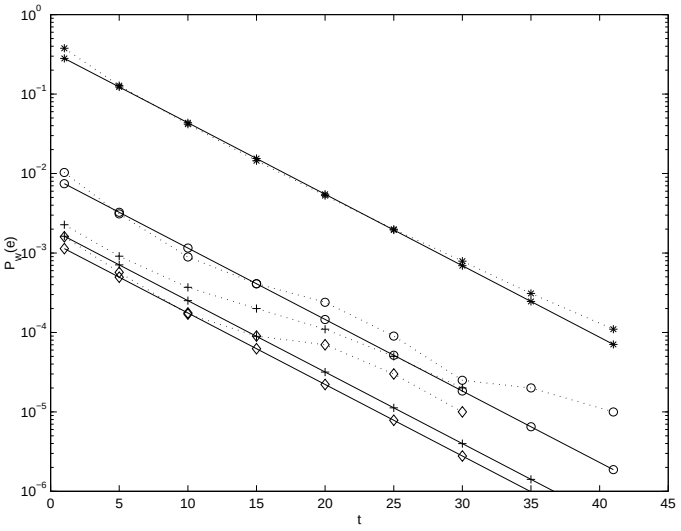

Fig. 6. Residual frame error probability after decoding of the concatenated coding scheme versus the error correction capability t of the BCH(1023,k) outer code, with $I = 2$ iterations ($*$), $I = 4$ iterations ($\circ$), $I = 7$ iterations ($+$) and $I = 10$ iterations ($\diamond$). The solid curves are related to the performance of the fitted model, whereas the dotted curves show simulation results.

where $P(j, n)$ denotes the probability that j bits out of n are affected by error. The BCH outer code should be chosen from the BCH codes set of length n multiple of the interleaver length L [26] with a certain error correction capability t. Here a set of BCH(1023, k) codes has been selected, with length n extended to $n = 1024$ by adding an overall parity check bit [36], with t ranging from 1 to 41 (and k from 648 to 1013, correspondingly).

To validate the model, a custom software based on [37] was employed to simulate the performances of the above described serial concatenation scheme over an AWGN channel, assuming a BPSK (Binary Phase Shift Keying) modulator.

Figs. 5 and 6 show, respectively, the residual bit error probability and the residual frame error probability after decoding of the concatenated coding scheme versus the error correction capability t of the BCH(1024, k) outer code, with respect to the iterations number I, at $E_b/N_0 = 2$ dB. The solid curves are related to the performance of the fitted model, whereas the dotted curves show simulation results.

Since, as said in the previous section, the linear regression curve obtained for $I = 2$ is a good approximation of $P(m, L)$, $\forall I$, for low values of m (less than about 15), and a lower bound on $P(m, L)$, $\forall I$, for higher m values, this model will work perfectly for $I = 2$, and approximately well for $I > 2$ for lower m values. In fact, for $I = 2$, the figures highlight a very good agreement between the true data source performance (dotted curve) and the performance of the fitted model (solid curve). For $I > 2$, the agreement between the true data source performance and the performance of the fitted model is good for low values of the error correction capability t, up to about 15. For $t > 15$, the figures highlight the fact that the true data source performance (dotted curves) are always lower bounded by the performance of the fitted model (solid curves).

VI. Conclusions and Open Problems

In this paper we proposed a model for the generation of error sequences with the same statistical characteristics as those output from a turbo decoder. This model is useful to avoid the extensive use of simulation needed to generate the error patterns at the output of a turbo decoder and to investigate the performance of concatenated schemes having a turbo code as inner code.

An example of the construction of this model was given, considering a particular turbo code with given rate, states number and interleaver length L, at a fixed value of E_b/N_0 in dB.

It was shown that the error statistics of iteratively decoded turbo codes may be approximated by using an overall two level model generator:

1) a memoryless generator, modelling the binary process of the length-L blocks of successes and failures,
2) an error generator uniformly distributing m errors within each length-L block (with m chosen according to (24)).

The results show that $P(m, L)$, i.e., the probability of having m errors in a wrong block of length L, given by (24), depends on I, but the linear regression curves obtained for the different I values intersect each other or are very close to each other for low values of m and, for higher m values, the linear regression curves obtained for $I > 2$ are all lower bounded by the linear regression curve obtained for $I = 2$. Thus, we have kept the latter as a good approximation of $P(m, L)$, $\forall I$, for low values of m (less than about 15), and as a lower bound on $P(m, L)$, $\forall I$, for higher m values. This fitted model is expected to work perfectly for $I = 2$, and approximately well for $I > 2$ and for low values of m. For higher m values, the linear regression curve obtained for $I = 2$ is expected to give a lower bound on the bit error performance obtained for $I > 2$.

To show that this model is well suited to represent the error patterns at the output of a turbo decoder, the performance analysis of a concatenated scheme, having the turbo-code considered in the example as inner code, was provided. As expected, this analysis confirms a very good agreement between the performance of the fitted model and the simulation results for $I = 2$. For $I > 2$, the agreement between the true data source performance and the performance of the fitted model was shown to be better for low values of the error correction capability t (up to about 15). On the other hand, for $t > 15$, the true data source performance (dotted curves) were shown to be always lower bounded by the performance of the fitted model (solid curves).

This work can be considered preparatory to the definition of a model for the generation of error patterns at the output of a turbo decoder (of given rate and interleaver length L) with parameters given by the iterations number I, the states number ν of the turbo code itself, and E_b/N_0 in dB.

Moreover, the definition of a model for the generation of error patterns at the output of a LDPC code could be of interest, too, since recently LDPC codes were also proposed as component codes of product code structures [38] for the next generation digital terrestrial broadcasting transmission system [39]. Since the main concern in these communication systems is the need of adaptive and flexible communication techniques, the results of [40] and [41] (based on the results obtained in [42] and [43]), where we recently addressed the design of rate-compatible punctured LDPC codes, may be combined with the results of this paper to design adaptive HCC structures. This design is useful for many practical applications and may take advantage from the method herein described to avoid heavy simulations and get faster results.

References

[1] F. Babich and F. Vatta, "On the error statistics of turbo decoding for hybrid concatenated codes design", *Proc. of the 2017 International Conference on Software, Telecommunications and Computer Networks*, Split, Croatia, September 21-23, 2017, pp. 1-5. DOI: 10.23919/SOFTCOM.2017.8115567.

[2] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: turbo codes", *Proc. of the 1993 IEEE International Conference on Communications*, Geneva, Switzerland, May 1993, pp. 1064-1070. DOI: 10.1109/ICC.1993.397441.

[3] D. Divsalar and F. Pollara, "Turbo codes for PCS applications", *Proc. of the 1995 IEEE International Conference on Communications*, Seattle, WA, USA, June 1995, pp. 54-59. DOI: 10.1109/ICC.1995.525138.

[4] D. J. Costello, J. Hagenauer, H. Imai, and S. B. Wicker, "Applications of error-control coding", *IEEE Transactions on Information Theory*, Vol. 44, No. 6, October 1998, pp. 2531-2560. DOI: 10.1109/18.720548.

[5] N. Kahale and R. Urbanke, "On the minimum distance of parallel and serially concatenated codes", *Proc. of the IEEE International Symposium on Information Theory*, Cambridge, MA, USA, August 1998, p. 31. DOI: 10.1109/ISIT.1998.708611.

[6] M. Breiling, "A logarithmic upper bound on the minimum distance of turbo codes", *IEEE Transactions on Information Theory*, Vol. 50, No. 8, August 2004, pp. 1692-1710. DOI: 10.1109/TIT.2004.831763.

[7] L. C. Perez, J. Seghers, and D. J. Costello, Jr., "A distance spectrum interpretation of turbo codes", *IEEE Transactions on Information Theory*, Vol. 42, No. 6, November 1996, pp. 1698-1709. DOI: 10.1109/18.556666.

[8] S. Benedetto and G. Montorsi, "Unveiling turbo codes: some results on parallel concatenated coding schemes", *IEEE Transactions on Information Theory*, Vol. 42, No. 2, March 1996, pp. 409-428. DOI: 10.1109/18.485713.

[9] L. Bazzi, M. Mahdian, and D. A. Spielman, "The minimum distance of turbo-like codes", *IEEE Transactions on Information Theory*, Vol. 55, No. 1, January 2009, pp. 6-15. DOI: 10.1109/TIT.2008.2008114.

[10] F. Fagnani and C. Ravazzi, "Spectra and minimum distances of repeat multiple accumulate codes", *IEEE Transactions on Information Theory*, Vol. 55, No. 11, November 2009, pp. 4905-4924. DOI: 10.1109/TIT.2009.2030459.

[11] A. Brown, M. Luby, and A. Shokrollahi, "Repeat-accumulate codes that approach the Gilbert-Varshamov bound", *Proc. of the IEEE International Symposium on Information Theory*, Adelaide, Australia, September 2005, pp. 169-173. DOI: 10.1109/ISIT.2005.1523316.

[12] D. Divsalar and F. Pollara, "Serial and hybrid concatenated codes with applications", *Proc. of the 1st International Symposium on Turbo Codes and Related Topics*, Brest, France, September 1997, pp. 80-87. DOI: 10.1.1.81.5866.

[13] A. Perotti, G. Montorsi, and S. Benedetto, "Performance analysis and optimization of concatenated block-turbo coding schemes", *Proc. of the 2004 IEEE International Conference on Communications*, Paris, France, June 20-24, 2004, pp. 1-5. DOI: 10.1109/ICC.2004.1312505.

[14] K. R. Narayanan and G. L. Stüber, "Selective serial concatenation of turbo codes", *IEEE Communications Letters*, Vol. 1, No. 5, September 1997, pp. 136-139. DOI: 10.1109/4234.625037.

[15] F. Babich and F. Vatta, "Turbo codes construction for robust hybrid multitransmission schemes", *Journal of Communication Software and Systems (JCOMSS)*, Vol. 7, No. 4, December 2011, pp. 128-135. DOI: 10.24138/jcomss.v7i4.174.

[16] C. Koller, A. Graell i Amat, J. Kliewer, F. Vatta, K. Sh. Zigangirov, and D. J. Costello, Jr., "Analysis and design of tuned turbo codes", *IEEE Transactions on Information Theory*, Vol. 58, No. 7, July 2012, pp. 4796-4813. DOI:10.1109/TIT.2012.2195711.

[17] M. Fu, "Stochastic analysis of turbo decoding", *IEEE Transactions on Information Theory*, Vol. 51, No. 1, January 2005, pp. 81-100. DOI: 10.1109/TIT.2004.839494.

[18] A. Fedoul and M. Benjillali, "LLR-based coded performance analysis of cognitive dual-hop BICM systems", *EURASIP Journal on Wireless Communications and Networking*, Vol. 2017, No. 195, 2017, pp. 1-8. DOI: 10.1186/s13638-017-0974-4.

[19] S. ten Brink, "Convergence behavior of iteratively decoded parallel concatenated codes", *IEEE Transactions on Communications*, Vol. 49, No. 10, October 2001, pp. 1727-1737. DOI: 10.1109/26.957394.

[20] A. A. Markov, "Theory of Algorithms", *Journal of Symbolic Logic*, Vol. 22, No. 1, 1957, pp. 77-79. DOI: 10.2307/2964063.

[21] M. I. Weinberger, J. J. Rissanen, and M. Feder, "A universal finite memory source", *IEEE Transactions on Information Theory*, Vol. 41, No. 3, May 1995, pp. 643-652. DOI: 10.1109/18.382011.

[22] F. M. J. Willems, Y. M. Shtarkov, and T. J. Tjalkens, "The context-tree weighting method: basic properties", *IEEE Transactions on Information Theory*, Vol. 41, No. 3, May 1995, pp. 653-664. DOI: 10.1109/18.382012.

[23] O. E. Kelly, "Intersymbol interference equalization by universal likelihood", Ph.D. dissertation, Rice University, Houston, TX, USA, 1996. Available at: https://hdl.handle.net/1911/19173.

[24] F. Babich, O. E. Kelly, and G. Lombardi, "A context-tree based model for quantized fading", *IEEE Communications Letters*, Vol. 3, No. 2, February 1999, pp. 46-48. DOI: 10.1109/4234.749358.

[25] D. J. Costello and G. Meyerhans, "Concatenated turbo codes", *Proc. of the 1996 International Symposium on Information Theory and Applications*, Victoria B. C., Canada, September 1996, pp. 571-574.

[26] J. D. Anderson, "Turbo codes extended with outer BCH codes", *IEE Electronics Letters*, Vol. 32, No. 22, 24th October 1996, pp. 2059-2060. DOI: 10.1049/el:19961383.

[27] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, John Wiley & Sons, Inc., New York, NY, USA, 1991. DOI: 10.1002/047174882X.

[28] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate", *IEEE Transactions on Information Theory*, Vol. 20, No. 2, March 1974, pp. 284-287. DOI: 10.1109/TIT.1974.1055186.

[29] A. Matache, S. Dolinar, and F. Pollara, "Stopping rules for turbo decoders", *TMO Progress Report 42-142*, Jet Propulsion Lab., Pasadena, CA, USA, August 15, 2000, pp. 1-22. DOI: 10.1.1.22.983.

[30] S. Benedetto and G. Montorsi, "Performance of continuous and block-wise decoded turbo codes", *IEEE Communication Letters*, Vol. 1, No. 3, May 1997, pp. 77-79. DOI: 10.1109/4234.585802.

[31] A. S. Barbulescu and S. S. Pietrobon, "Interleaver design for turbo codes", *Electronics Letters*, Vol. 30, No. 25, December 1994, pp. 2107-2108. DOI: 10.1049/el:19941434.

[32] A. S. Barbulescu and S. S. Pietrobon, "Interleaver design for three dimensional turbo-codes", *Proc. of the 1995 IEEE International Symposium on Information Theory*, Whistler BC, Canada, September 17-22, 1995, p. 37. DOI: 10.1109/ISIT.1995.531139.

[33] A. Banerjee, F. Vatta, B. Scanavino, and D. J. Costello, "Nonsystematic turbo codes", *IEEE Transactions on Communications*, Vol. 53, No. 11, November 2005, pp. 1841-1849. DOI: 10.1109/TCOMM.2005.858672.

[34] I. Chatzigeorgiou, M. R. D. Rodrigues, I. J. Wassell, and R. A. Carrasco, "Analysis and design of punctured rate-1/2 turbo codes exhibiting low error floors", *IEEE Journal on Selected Areas in Communications*, Vol. 27, No. 6, August 2009, pp. 944-953. DOI: 10.1109/JSAC.2009.090812.

[35] G. C. Clark, Jr. and J. B. Cain, *Error-correction coding for digital communications*, Plenum Press, New York, 1981. DOI: 10.1007/978-1-4899-2174-1_6.

[36] W. W. Peterson and E. J. Weldon, Jr., *Error-correcting codes*, The MIT Press, Cambridge, MA, USA, 1972.

[37] A. Boscolo, F. Vatta, F. Armani, E. Viviani, and D. Salvalaggio,"Physical AWGN channel emulator for Bit Error Rate test of digital baseband communication", *Applied Mechanics and Materials*, Vols. 241-244, 2013, pp. 2491-2495. DOI: 10.4028/www.scientific.net/AMM.241-244.2491.

[38] V. Sidorenko, M. Bossert, and F. Vatta, "Properties and encoding aspects of direct product convolutional codes", *Proc. of the 2012 IEEE International Symposium on Information Theory*, Boston, MA, USA, July 1-6, 2012, pp. 2351-2355. DOI: 10.1109/ISIT.2012.6283934.

[39] B. Liu, Y. Li, B. Rong, L. Gui, and Y. Wu, "LDPC-RS product codes for digital terrestrial broadcasting transmission system", *IEEE Trans. on Broadcasting*, Vol. 60, No. 1, March 2014, pp. 38-49. DOI: 10.1109/TBC.2013.2291359.

[40] F. Babich, M. Noschese, and F. Vatta, "Analysis and Design of Rate Compatible LDPC Codes", *Proc. of the 27th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC'16*, Valencia, Spain, September 4-8, 2016, pp. 1-6. DOI: 10.1109/PIMRC.2016.7794684.

[41] F. Babich, M. Noschese, A. Soranzo, and F. Vatta, "Low complexity rate compatible puncturing patterns design for LDPC codes", *Proc. of the 2017 International Conference on Software, Telecommunications and Computer Networks*, Split, Croatia, September 21-23, 2017, pp. 1-5. DOI: 10.23919/SOFTCOM.2017.8115558.

[42] F. Babich, A. Soranzo, and F. Vatta, "Useful mathematical tools for capacity approaching codes design", *IEEE Communications Letters*, Vol. 21, No. 9, Sept. 2017, pp. 1949-1952. DOI: 10.1109/LCOMM.2017.2714684.

[43] F. Vatta, A. Soranzo and F. Babich, "Low-complexity bound on irregular LDPC belief-propagation decoding thresholds using a Gaussian approximation", *Electronics Letters*, Vol. 54, No. 17, August 23rd, 2018, pp. 1038-1040. DOI: 10.1049/el.2018.0478.

Fulvio Babich received the doctoral degree, (Laurea), cum laude, in Electrical Engineering, at the University of Trieste, on July 1984. After graduation he was with Telettra at the Research and Development Laboratories, where he was engaged in optical fiber communications. Then he joined Zeltron, where he was a communication system engineer, responsible of the activities within the ESPRIT program. In 1992 he joined the Department of Electrical Engineering (DEEI) of the University of Trieste, where he is Professor of Signals and Systems and Wireless Networks. Fulvio Babich is vice director of Department of Engineering and Architecture. He is the coordinator of the Ph.D. in Industrial and Information Engineering of the University of Trieste. His current research interests are in the field of wireless networks and personal communications. He is involved in channel modeling, multiple access techniques, channel encoding, error control techniques and cross-layer design. Fulvio Babich serves as reviewer for many international journals, has served as co-chair for the Communication Theory Symposium, ICC 2005, Seul, ICC 2014, Sydney, and ICC 2017, Paris, for the Wireless Communication Symposium, ICC 2011, Kyoto, and for the Wireless Communication Symposium, WCSP 2012, Huangshan, China. He has served as General chair of the 13th IEEE IFIP Annual Mediterranean Ad Hoc Networking Workshop, Med-Hoc-Net 2014 June 2-4, 2014 Piran, Slovenia. Fulvio Babich is Senior Member of IEEE.

Francesca Vatta received the M.S. Degree in Electronic Engineering and the Ph.D. Degree in Telecommunications from the University of Trieste, Trieste, Italy, in 1992 and 1998, respectively. She joined the Department of Engineering and Architecture (DIA) of the University of Trieste in 1999, where she is Assistant Professor of Information Theory and Error-Control Coding. Starting in 2002, she spent several months as visiting scholar at the University of Notre Dame, Notre Dame, IN, U.S.A., cooperating with the Coding Theory Research Group under the guidance of Prof. D. J. Costello, Jr. Starting in 2005, she spent several months as visiting scholar at the University of Ulm, Germany, cooperating with the Telecommunications and Applied Information Theory Research Group under the guidance of Prof. M. Bossert. She is an author of more than 80 papers published on international journals and conference proceedings. Her current research interests are in the area of channel coding concerning, in particular, the analysis and design of capacity achieving coding schemes.