Unveiling Turbo Codes: Some Results on Parallel Concatenated Coding Schemes

Sergio Benedetto, *Senior Member, IEEE*, and Guido Montorsi, *Member, IEEE*

***Abstract*— A parallel concatenated coding scheme consists of two simple constituent systematic encoders linked by an interleaver. The input bits to the first encoder are scrambled by the interleaver before entering the second encoder. The codeword of the parallel concatenated code consists of the input bits to the first encoder followed by the parity check bits of both encoders. This construction can be generalized to any number of constituent codes. Parallel concatenated schemes employing two convolutional codes as constituent codes, in connection with an iterative decoding algorithm of complexity comparable to that of the constituent codes, have been recently shown to yield remarkable coding gains close to theoretical limits. They have been named, and are known as, "turbo codes." We propose a method to evaluate an upper bound to the bit error probability of a parallel concatenated coding scheme averaged over all interleavers of a given length. The analytical bounding technique is then used to shed some light on some crucial questions which have been floating around in the communications community since the proposal of turbo codes.**

***Index Terms*— Turbo codes, concatenated codes, iterative decoding.**

I. Introduction And Motivations

KNOWLEDGE of the fact that increasing the codeword length n of block codes (or the constraint length of convolutional codes) leads to better performance dates back to Shannon theory. It is also well known that the complexity of maximum-likelihood (ML) decoding algorithms increases with n, up to a point where decoding becomes physically unrealizable.

Thus the research in coding theory has seen many proposals aiming at constructing powerful codes with large equivalent block lengths structured so as to permit breaking the ML decoding into simpler partial decoding steps. Iterated codes [1], product codes and their extension [2], concatenated codes [3] and their generalized version [4], and large constraint-length convolutional codes with suboptimal decoding strategies, like sequential decoding, are nonexhaustive examples of these attempts, and some of them have been successfully employed in applications where large coding gains are required, such as, for example, deep-space communication.

Manuscript received February 27, 1995; revised October 17, 1995. This work was supported by European Space Agency and by CNR under Progetto Finalizzato Trasporti, subproject "Prometheus." The material in this paper was presented in part at the IEEE International Symposium on Information Theory, Whistler, BC, Canada, September 17–22, 1995.

The authors are with the Dipartimento di Elettronica, Politecnico di Torino, 10129 Torino, Italy.

Publisher Item Identifier S 0018-9448(96)01685-9.

The most recent successful attempt consists of the so-called "turbo codes" [5], whose astonishing performance has given rise to a large interest in the coding community. They are *parallel concatenated codes* (PCC) (see Fig. 1 for the case of block PCC) whose encoder is formed by two (or more) *constituent* systematic encoders joined through an interleaver. The input information bits feed the first encoder and, after having been interleaved by the interleaver, enter the second encoder. The codeword of the parallel concatenated code consists of the input bits to the first encoder followed by the parity check bits of both encoders.

The suboptimal iterative decoder is modular, and consists of a number of equal component blocks formed by concatenating the decoders of the constituent codes (CC) separated by the same interleaver used in the encoder. Each decoder produces weighted soft decoding of the input sequence. By increasing the number of decoding modules, and thus the number of decoding iterations, bit error probabilities as low as 10^{-5} at $E_b/N_0 = 0.0$ dB have been shown by simulation [6]. A version of turbo codes employing two eight-state convolutional codes as constituent codes, an interleaver of 32×32 bits and an iterative decoder performing two and a half iterations with a complexity of the order of nine times the ML Viterbi decoding of each constituent code is presently available on a chip yielding a measured bit error probability of $9.0 \cdot 10^{-7}$ at $E_b/N_0 = 3$ dB [7].

Bandwidth-efficient versions of turbo codes, compared to trellis-coded modulation schemes have also been proposed [8], as well as turbo codes based on block (instead of convolutional) codes [9], [10].

A careful examination of the literature shows that, rather than being a sudden apparition, turbo codes are the result of a clever intuition building on several concepts already available. We can cite in general the literature on product and concatenated codes in relation with the idea of parallel concatenation, the pioneering work on symbol-by-symbol maximum *a posteriori* decoding of linear codes [11] and the proposals in [12]–[14] of the soft-decisions Viterbi algorithm in relation to the way of implementing the iterative decoder. Very close to turbo codes are also the separable "filters" described in [15] to iteratively decode multidimensional codes.

As for the applications, it must be mentioned that PCC's, like all codes with very long codewords, suffer from one important drawback, namely the delay due to the interleaver and to the iterative decoding (as an example, the previously mentioned "chip" has a latency of 2318 bits). This prevents them from being used in applications where the combination

of decoding delay and data rate leads to intolerable delays, like digital telephony. A broad range of applications still remains, such as digital audio and video broadcasting, data packet transmission, and space applications. It is also worthwhile mentioning that the interleaver inherently present in the PCC can prove beneficial for transmission on fading channels [8].

Since the successful proposal of turbo codes, neither a good theoretical explanation of the codes behavior/performance nor an adequate comprehension of the role and relative importance of the PCC ingredients (constituent codes and interleaver) have appeared.

In terms of performance of PCC's, apart from the measurements on the chip [7], what is known is essentially due to simulation [5], [8], [15]–[18], which, in itself, is not at all a simple task, as it requires a huge amount of computer time to obtain reliable results down to bit error probabilities like 10^{-6}.

As a consequence, a number of basic questions are still unanswered:

1) What is the performance of the ML decoder ?
2) What are the relative contributions of the constituent codes and of the interleaver length in determining the PCC performance ?
3) For a given interleaver length, how sensitive is the performance to the interleaver choice ?
4) How crucial is the use of recursive (feedback) systematic convolutional codes (as opposed to nonrecursive ones) as constituent codes of the PCC scheme ?
5) How close are the proposed suboptimal iterative decoding algorithms to ML decoding?

Answering these questions is certainly important from a theoretical point of view. Some of them, however, have significant practical relevance as well. For example, questions 1 and 5 can encourage (or discourage) the search for improved decoding algorithms, and question 2 may lead to the optimization of the PCC for given system constraints such as delay or complexity. Question 3, in turn, is related to the importance of the interleaver optimization, a topic which has already received some "cut-and-try" attention [17]–[19]. Finally, question 4 has been discussed in [20] where the authors seem to believe that recursive convolutional codes have superior merits in themselves, rather than only when used as CC of a PCC.

Formidable complexity obstacles discourage classical theoretical analysis of the PCC's. As an example, the code implemented in VLSI in [7], when seen as a whole convolutional code, consists of an equivalent time-varying convolutional code with 2^{1030} states, thus preventing any analytical evaluation of the main performance parameters.

In this paper, we will try to shed some light on the theoretical comprehension of PCC's. We will propose answers to the previous questions, some of which may be only preliminary yet indicative of the right direction.

In particular, we will define and evaluate an upper bound to the *average performance* of the ML soft decoder for a PCC, stemming from characteristics of the CC's. Owing to its definition, the average performance, expressed in terms of bit error probability, turns out to be independent from the particular interleaver used, and helps in assessing what can be gained with given CC's and with an interleaver of a given length.

We will also present simulation results for PCC's with differently chosen interleavers of the same length and compare them with the proposed bound. The results show that "random" interleavers offer performance close to the average ones evaluated through the upper bound, independent, to a large extent, from the particular interleaver. Bad interleavers are very easy to avoid in practice.

Moreover, we will show that recursive convolutional codes, although providing almost the same performance as nonrecursive codes when used alone, are indeed crucial when embedded in a PCC as CC's.

Finally, by comparing our bound on ML performance with simulation results based on iterative decoding, we will give heuristic evidence that the suboptimal algorithms can come very close to the optimum.

To help the reader, we will accompany the description with frequent examples, and will start from the simpler case of PCC schemes using block codes as CC's (parallel concatenated block code (PCBC)) leaving for the final sections the more complicated case of parallel concatenated convolutional codes (PCCC).

II. Notations and Definitions

Notations and definitions will be introduced for the case of parallel concatenated block codes, the extension to convolutional codes being straightforward.

Given an (n, k) systematic block code C, its well-known weight enumerating function (WEF) is

$$B^C(H) \triangleq \sum_{i=0}^{n} B_i H^i$$

where B_i is the (integer) number of codewords with Hamming weight (number of ones) i and H is a dummy variable. The WEF of a code can be used to compute the exact expression of the probability of undetected errors and an upper bound to the word error probability [21].

We define the *input-redundancy weight enumerating function* (IRWEF) of the code as

$$A^C(W, Z) \triangleq \sum_{w,j} A_{w,j} W^w Z^j$$

where $A_{w,j}$ denotes the (integer) number of codewords generated by an input information word of Hamming weight w whose parity check bits have Hamming weight j, so that the overall Hamming weight is $w + j$.

The IRWEF makes explicit in each term of the WEF the separate contributions of the information and of the parity-check bits to the total Hamming weight of the codewords, and thus provides additional information on the (Hamming) weight profile of the code. It will prove crucial in the following when dealing with parallel concatenated codes (PCC), since the two input words to the constituent encoders, the second being obtained by interleaving the first, share the same Hamming weight, so that the redundant bits generated by the two

encoders derive from terms of the IRWEF with the same input weight w.

We mention also that the IRWEF characterizes the whole encoder, as it depends on both input information words and codewords, whereas the WEF only depends upon the code. As a consequence, the WEF is related to the word error probability of the code, whereas the IRWEF provides information on the bit error probability.

Obviously, the following relationship holds true:

$$B^C(H) = A^C(W = H, Z = H)$$

with

$$A^C(H,H) = \sum_{w,j} A_{w,j} H^{w+j} = \sum_k B_k H^k$$

where

$$B_k = \sum_{w+j=k} A_{w,j} \ .$$

Example 1: The $(7,4)$ Hamming code has the following WEF:

$$B^C(H) = 1 + 7H^3 + 7H^4 + H^7.$$

Splitting the contribution of the information and redundancy bits to the total codeword weight we obtain the IRWEF of the code

$$A^C(W,Z) = 1 + W(3Z^2 + Z^3) + W^2(3Z + 3Z^2) + W^3(1 + 3Z) + W^4 Z^3. \quad (1)$$

◇

Consider now the *conditional weight enumerating function* $A_w^C(Z)$ of the parity check bits generated by the code C corresponding to the input words of weight w. It can be obtained from the IRWEF as

$$A_w^C(Z) = \sum_j A_{w,j} Z^j = \frac{1}{w!} \cdot \frac{\partial^w A^C(W,Z)}{\partial W^w}\bigg|_{W=0}$$

so that we can also write the inverse relationship

$$A^C(W,Z) = \sum_w W^w A_w^C(Z). \quad (2)$$

Both IRWEF and the $A_w^C(Z)$ can be used with the union bound to compute an upper bound to the bit error probability for ML soft decoding of the code over a channel with additive white Gaussian noise in the form

$$\begin{aligned} P_b(e) &\le \frac{W}{k} \frac{\partial A^C(W,Z)}{\partial W}\bigg|_{W=Z=e^{-R_c E_b/N_0}} \\ &= \sum_{w=1}^{k} \frac{w}{k} W^w A_w^C(Z)\bigg|_{W=Z=e^{-R_c E_b/N_0}} \\ &= \sum_m D_m H^m\bigg|_{H=e^{-R_c E_b/N_0}} \end{aligned} \quad (3)$$

with

$$D_m \triangleq \sum_{j+w=m} \frac{w}{k} A_{w,j} \quad (4)$$

where R_c is the code rate.

The second and third line of (3) represent two equivalent expressions to bound the bit error probability. The first expression keeps distinct the contributions of information words with different weight w, whereas the second sums the contributions according to the overall weight m of the codeword through the coefficient D_m defined in (4).

A tighter bound can also be obtained from (3) [22] exploiting the inequality

$$\operatorname{erfc}(\sqrt{x+y}) \le \operatorname{erfc}(\sqrt{x}) e^{-y}.$$

It assumes the form

$$P_b(e) \le \frac{W}{2k} \operatorname{erfc}\left(\sqrt{\frac{d_{\min} R_c E_b}{N_0}}\right) \cdot e^{\frac{d_{\min} R_c E_b}{N_0}} \frac{\partial A^C(W,Z)}{\partial W}\bigg|_{W=Z=e^{-R_c E_b/N_0}} \quad (5)$$

which admits of course a further development like (3).

For parallel concatenated convolutional codes the information and code sequences are semi-infinite and, as a a consequence, the summation (explicit in (3) and implicit in (5)) must be truncated to a finite value. For the bound in the second line of (3) the truncation involves the computation of the complete conditional weight distributions up to a given information weight, whereas for the bound in the third line the truncation leads to the computation of the weight multiplicities of the unconditional weight distribution up to a given overall weight of the code sequences. Computing algorithms and a comparison of the two approximations are discussed in [23].

Using a finite number of terms in (5) transforms the upper bound into the approximation

$$P_b(e) \approx \frac{1}{2} \sum_m D_m \operatorname{erfc}\left(\sqrt{m \frac{R_c E_b}{N_0}}\right). \quad (6)$$

In the following, all the results in terms of bit error probability will be computed using (6).

Example 2: The conditional WEF's of the Hamming code $(7,4)$ considered in the previous example are

$$\begin{aligned} A_0^C(Z) &= 1 \\ A_1^C(Z) &= 3Z^2 + Z^3 \\ A_2^C(Z) &= 3Z + 3Z^2 \\ A_3^C(Z) &= 1 + 3Z \\ A_4^C(Z) &= Z^3 \end{aligned}$$

so that the upper bound on the bit error probability computed through (6) and (4) becomes

$$P_b(e) \le \frac{3}{2} \operatorname{erfc}\left(\sqrt{\frac{3R_c E_b}{N_0}}\right) + 2 \operatorname{erfc}\left(\sqrt{\frac{4R_c E_b}{N_0}}\right) + \frac{1}{2} \operatorname{erfc}\left(\sqrt{\frac{7R_c E_b}{N_0}}\right).$$

◇

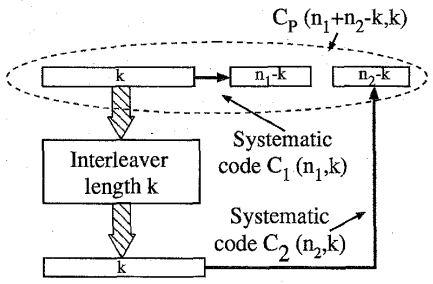

Fig. 1. Parallel concatenated block code.

III. Parallel Concatenated Block Codes

Consider now a parallel concatenated block code (PCBC) obtained as in Fig. 1. Two linear systematic block codes C_1 with parameters (n_1, k) and C_2 with parameters (n_2, k), the constituent codes (CC), having in common the length k of the input information bits, are linked through an interleaver so that the information part of the second codeword is just a permuted version of the first one. The PCBC codeword is then formed by adding to the input bits the parity-check bits generated by the first and second encoder. The PCBC, that we denote as C_P, is then a $(n_1 + n_2 - k, k)$ linear code as the interleaver performs a linear operation on the input bits.

If w is the (Hamming) weight of the input word, and z_1 and z_2 the weights of the parity check bits introduced by the first and second encoders, respectively, the weight of the corresponding codeword of C_P will be $w + z_1 + z_2$.

We want now to obtain the IRWEF $A^{C_P}(W, Z)$ of C_P starting from the knowledge of those of the constituent codes. For a given interleaver, this operation is exceedingly complicated, as the redundant bits generated by the second encoder will not only depend on the weight of the input word, but also on how its bits have been permuted by the interleaver. The only viable solution, in theory, would be an exhaustive enumeration of all possible cases; in practice, this is an impossible achievement for large k, and this was precisely the reason for lengthy computer simulations.

To overcome this difficulty, we introduce an abstract interleaver called *uniform interleaver*, defined as follows.

Definition 1: A *uniform interleaver* of length k is a probabilistic device which maps a given input word of weight w into all distinct $\binom{k}{w}$ permutations of it with equal probability $1/\binom{k}{w}$. △

From the definition, it is apparent that the conditional weight enumerating function $A_w^{C_2}(Z)$ of the second code becomes independent from that of the first code thanks to the uniform randomization produced by the interleaver.

As a nice consequence of this, we can easily evaluate the conditional weight enumerating function of the PCBC which uses the uniform interleaver as the product, suitably normalized, of the two conditional weight enumerating functions of the constituent codes

$$A_w^{C_P}(Z) = \frac{A_w^{C_1}(Z) \cdot A_w^{C_2}(Z)}{\binom{k}{w}}. \tag{7}$$

Also, from (2) we obtain the IRWEF of the code C_P as

$$A^{C_P}(W, Z) = \sum_{w=1}^{k} W^w A_w^{C_P}(Z). \tag{8}$$

Example 3: The IRWEF of the PCBC constructed using as constituent codes two identical $(7, 4)$ Hamming codes can be obtained plugging the conditional WEF obtained in the previous example into (7) and applying (8)

$$\begin{aligned} A^{C_P}(W, Z) = 1 &+ W(2.25Z^4 + 1.5Z^5 + 0.25Z^6) + \\ &+ W^2(1.5Z^2 + 3Z^3 + 1.5Z^4) + \\ &+ W^3(0.25 + 1.5Z + 2.25Z^2) + W^4Z^6. \end{aligned} \tag{9}$$

Notice in (9) the presence of fractional coefficients representing the multiplicity of the various terms. They are a direct consequence of the use of the uniform interleaver. ◇

The introduction of the uniform interleaver permits an easy derivation of the weight enumerating functions of the PCBC. However, in practice, one is confronted with deterministic interleavers, which give rise to one particular permutation of the input bits. So, what is the significance of the preceding definitions and equations ?

To answer this question, we prove now the main property of a PCBC which uses the uniform interleaver.

Theorem 1: Let $A^{C_{P_k}}(W, Z)$ be the IRWEF of the code C_{P_k} obtained using the particular interleaver I_k. Then

$$\mathrm{E}_k\left[A^{C_{P_k}}(W, Z)\right] = A^{C_P}(W, Z) \tag{10}$$

where E_k means expectation with respect to the whole class of interleavers. ▽

Proof of Theorem 1: The proof makes use of (2) through the following equality chain:

$$\mathrm{E}_k\left[A^{C_{P_k}}(W, Z)\right] = \mathrm{E}_k\left[\sum_w W^w A_w^{C_{P_k}}(Z)\right] \tag{11}$$

$$= \sum_w W^w \mathrm{E}_k\left[A_w^{C_{P_k}}(Z)\right] \tag{12}$$

$$= \sum_w W^w A_w^{C_P}(Z) = A^{C_P}(W, Z). \tag{13}$$

where the third equality comes from the definition of the uniform interleaver. **QED**

A second result, which comes as a corollary of the previous one from the linear dependency of (6) with respect to the conditional weight enumerating function, is the following.

Corollary 1: The upper bound computed using the IRWEF $A^{C_P}(W, Z)$ coincides with the average of the upper bounds obtainable with the whole class of deterministic interleavers. ▽

The corollary guarantees that, for each value of the signal-to-noise ratio, the performance obtained with the uniform interleaver are achievable by at least one deterministic interleaver.

Example 4: We can check the result (10) by computing, for the simple example of Hamming code previously examined, the IRWEF's of the PCBC's constructed using all the inter-

TABLE I
IRWEF OF THE PARALLEL CONCATENATED CODES BASED ON THE $(7,4)$ HAMMING CODE FOR ALL POSSIBLE INTERLEAVERS

Perm.	$A^{C_{P_k}}(W,Z)$
0123 0321 1023 1320 3021 3120	$1+W(3Z^4+Z^6)+W^2(3Z^2+3Z^4)+W^3(1+3Z^2)+W^4Z^6$
0132 0213 0231 0312 1032 1203 1230 1302 2013 2031 2103 2130 2301 2310 3012 3102 3201 3210	$1+W(2Z^4+2Z^5)+W^2(Z^2+4Z^3+Z^4)+W^3(2Z+2Z^2)+W^4Z^6$

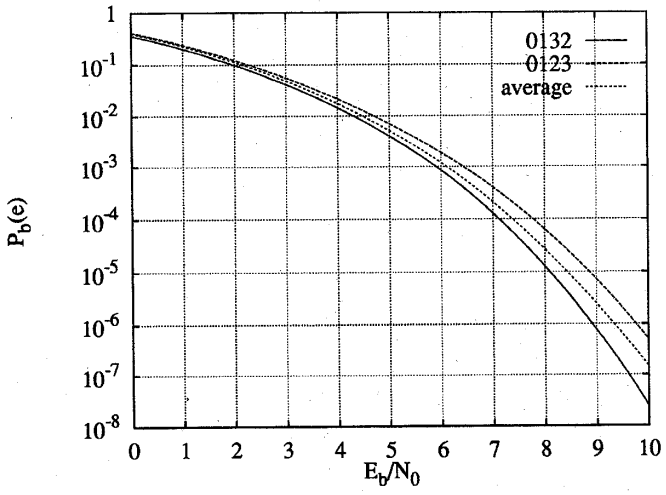

Fig. 2. Upper bounds for Example 4.

leavers originating from the $24 = 4!$ permutations of the input bits. The computed IRWEF's are reported in Table I.

From the table, it is apparent that, for this scheme, only two types of IRWEF are possible:

$$A^{C_{P_1}}(W,Z) = 1 + W(2Z^4 + 2Z^5) + W^2(Z^2 + 4Z^3 + Z^4) + W^3(2Z + 2Z^2) + W^4Z^6 \quad (14)$$

which derives from 18 different permutations and

$$A^{C_{P_2}}(W,Z) = 1 + W(3Z^4 + Z^6) + W^2(3Z^2 + 3Z^4) + W^3(1 + 3Z^2) + W^4Z^6 \quad (15)$$

which appears six times. The average computed over all possible interleavers yields

$$A^{C_P}(W,Z) = \frac{18}{24}A^{C_{P_1}}(W,Z) + \frac{6}{24}A^{C_{P_2}}(W,Z)$$

which coincides with (9).

The upper bounds obtained substituting (14), (15), and (9) into (6) are plotted in Fig. 2. ◇

Let $n, k, t)$ denote the parameters of a t-error correcting (n,k) code. We have also analyzed the performance achieved by a PCBC using as CC the $(63,57,3)$ and the $(63,51,5)$ BCH codes. The interleavers have lengths $k = 57$ and 51,

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.