On the Theory and Performance of Trellis Termination Methods for Turbo Codes

Hokfelt, Johan; Edfors, Ove; Maseng, Torleiv

Published in:
IEEE Journal on Selected Areas in Communications

DOI:
10.1109/49.924868

2001

On the Theory and Performance of Trellis Termination Methods for Turbo Codes

Author	Affiliation
Johan Hokfelt	Fiberless Society AB, Lund, Sweden
Ove Edfors	Dept. of Electroscience, Lund University, Lund, Sweden
Torleiv Maseng	Norwegian Defence Research Establishment, Kjeller, Norway

***Abstract*—The performance of a Turbo code can be severely degraded if no trellis termination is employed. This paper investigates the implications of the choice of trellis termination method for Turbo codes, and explains the origin of the performance degradation often experienced without trellis termination. An efficient method to derive the distance spectrum of Turbo codes for different trellis termination methods is presented. Further, we present interleaver design rules that are tailored to each termination method. Using interleavers designed with these restrictions, we demostrate that the performance difference between various termination methods are very small, including no trellis termination at all. For example, we demonstrate a Turbo code with a 500-bit interleaver that exhibits no sign of an error floor for frame error rates as low as 10^{-8}, even though no trellis termination is employed.**

***Index Terms*—Turbo codes, Trellis termination, Distance spectra, Interleaver design**

I. Introduction

Turbo codes are in general implemented as two recursive convolutional encoders in parallel, where the input to the second encoder is an interleaved version of the original information sequence [1], [2]. At the beginning of each information block, the encoders are initialized to their zero-states. Similarly, it is desirable to force the encoders back to the zero-state at the end of the information block, an operation known as trellis termination. For feedforward convolutional encoders, this is readily achieved by appending zeros, known as tail bits, at the end of the encoder input sequence. However, the recursive property of the constituent encoders used in Turbo codes implies a state-dependency on these tail bits, and hence, individual tail sequences are required for each constituent encoder.

The problem of trellis termination for Turbo codes has been addressed by many authors. In the original Turbo code proposed by Berrou et al. [2], the trellis of the first encoder was terminated in the zero state while the second encoder was truncated in an unknown state. Since then, various techniques have been presented which allow both encoders to be terminated in their zero states. Examples of such techniques are: location of input positions that separately influence the final states of both encoders [3], interleaver restrictions that terminate the second encoder in the same state as the first encoder [4], [5], tail-biting Turbo codes [6], and post interleaver flushing [7]. Naturally, it is also an option to truncate both encoders without any trellis termination at all, as investigated for example in [8], [9]. Additional reports on trellis termination for Turbo codes are found in, among others, [10], [11].

The performance of trellis termination methods are, to various extents, dependent on the particular interleaver used in the Turbo encoder – a fact rarely acknowledged in the literature. This dependency is the result of *interleaver edge effects* [12]. These edge effects degrade the distance spectrum of Turbo codes in situations where at least one of the encoder trellises is truncated in an unknown state. The degree of distance spectrum degradation is highly dependent on the particular choice of interleaver. Thus, the performance of a trellis termination method is the result of the combination of the termination method *and* the edge effects present for the particular interleaver used.

This paper describes interleaver edge effects in detail, and demonstrates how different termination methods are unequally sensitive to the phenomenon. A method to calculate the distance spectrum of Turbo codes for different trellis termination methods is presented. Further, we describe how the interleaver edge effects can be avoided by means of sophisticated interleaver design, hereby significantly reducing the need for trellis termination. Using interleavers designed specifically for each termination method, we investigate the error correcting performances of different termination schemes. The investigated trellis termination methods are: *no termination at all*, *termination of the first encoder only*, termination of both encoders with *post interleaver flushing* [7], termination of both encoders using both *self-terminating interleavers* [4], [5] and *dual termination* [3].

A typical Turbo code encoder is depicted in Figure 1. The constituent encoders are recursive convolutional encoders with m memory elements and $M = 2^m$ states. The two encoders are linked by an interleaver of length N, so that every block of N information bits entering the second constituent encoder is an interleaved, or permuted, version of the original N-bit information block. Depending on the degree of puncturing performed on the two parity sequences, the overall code rate can be varied from 1/3 to 1/1.

Due to the interleaver, the overall state space of the encoder becomes prohibitively large for maximum likelihood decoding. However, reasonably low complexity decoding is obtained by iteratively decoding the codes generated by the first and second constituent encoders, respectively. The decoding of the con-

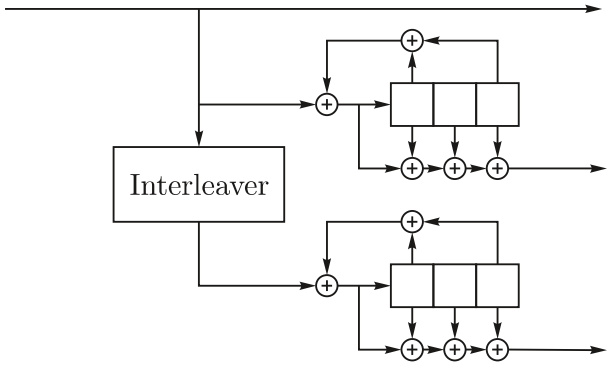

Fig. 1. Turbo encoder structure.

stituent codes requires a soft-in/soft-out decoding algorithm, providing *a posteriori* probabilities (APPs) of the symbols in the decoded sequence. One such algorithm is the BCJR algorithm [13], which provides optimal APPs on a symbol-by-symbol basis. Due to the comparably high complexity of the BCJR algorithm, soft-output algorithms based on the Viterbi algorithm (SOVA) were developed [14], [15]. In our simulations of Turbo codes, we use 15 decoding iterations employing the full BCJR algorithm, implemented in the logarithmic domain [16]. Further, all transmission is over the additive white Gaussian noise (AWGN) channel.

This paper is organized as follows. In Section II, we restate the theory required to obtain the distance spectrum of the ensemble of Turbo codes, and present an efficient method to include the effects of different trellis termination methods in these calculations. In Section III, we investigate the implications of using specific interleavers, whose performance deviates significantly from that of the ensemble of all interleavers. In Section IV, we discuss the issue of decoder initialization for the case when the trellis is truncated in an unknown state. Finally, we summarize our investigation in Section V, which is followed by an appendix describing interleaver design restrictions that are tailored for specific choices of trellis termination methods.

II. Influence of Trellis Termination on the Distance Spectrum

The standard method used for performance assessment of Turbo codes, as for conventional block- and convolutional codes, is to calculate the Hamming distance spectrum of the code. With the distance spectrum, or, equivalently, the weight distribution, lower and upper bounds on the error correcting performance can be derived. However, the calculation of the distance spectrum of a specific Turbo code involves taking the particular interleaver used into account, a task that becomes prohibitively complex for reasonably large interleaver sizes. A less computationally demanding method was introduced by Benedetto et al. in [17], where they presented a method of deriving the average distance spectrum for the ensemble of all interleavers of a certain length. In this section we summarize the method presented in [17], and present an efficient method to include the influences of different trellis termination methods.

Benedetto et al. introduced [17] the *input-redundancy weight enumerating function* (IRWEF)

$$A\left(W,Z\right) \triangleq \sum_{w=0}^{N} \sum_{j=0}^{n-N} A_{w,j} W^w Z^j, \tag{1}$$

for an (n,N)-code, where N is the length of the interleaver and n is the codeword length. $A_{w,j}$ is the number of codewords with input weight w and parity weight j, and W and Z are dummy variables. At this point, we restrict the coefficients $A_{w,j}$ to include trellis paths that start from the zero state and end in the zero state after N trellis transitions. The trellis paths that do not end in the zero state will be treated separately in Section II-A.

Since both constituent encoders in a Turbo code share the same input weight w, every codeword that belongs to a Turbo code is a combination of two constituent code codewords that both result from weight w input sequences. It is therefore convenient to define the *conditional weight enumerating function* (CWEF)

$$A_w\left(Z\right) \triangleq \sum_{j=0}^{n-N} A_{w,j} Z^j, \qquad w = 0, 1, \ldots, N,$$

which enumerates the number of codewords of various parity weights j, conditioned on a certain input-weight w. The CWEF of the first and second constituent encoders are denoted $A_w^{C_1}\left(Z\right)$ and $A_w^{C_2}\left(Z\right)$ respectively, and the CWEF of the overall Turbo code $A_w^{TC}\left(Z\right)$. Introducing a probabilistic interleaver construction called a *uniform interleaver*, for which all distinct mappings are equally probable, Benedetto et al. obtained the CWEF of the ensemble of all Turbo codes using interleavers with length N as

$$A_w^{TC}\left(Z\right) = \frac{A_w^{C_1}\left(Z\right) A_w^{C_2}\left(Z\right)}{\binom{N}{w}},$$

where $1 \Big/ \binom{N}{w}$ is the probability that a specific weight-w sequence is mapped to another, specific, weight-w sequence. Finally, the multiplicities a_d of codewords with Hamming weight d is achieved as

$$a_d = \sum_{w=1}^{N} A_{w,d-w}^{TC}, \tag{2}$$

where $A_{w,d-w}^{TC}$ are the coefficients in the Turbo code CWEF, i.e. $A_w^{TC}\left(Z\right) \triangleq \sum_{j=0}^{n-N} A_{w,j}^{TC} Z^j$. Note that, due to the averaging of the ensemble of distance spectra, each multiplicity a_d and $A_{w,j}^{TC}$ are not necessarily integers.

The distance spectrum obtained with (2) can be used to derive an upper bound on the error correcting performance. For AWGN channels and maximum likelihood decoding, the error rate performance is upper bounded by

$$P_{FER} \leq \sum_{d} a_d Q\left(\sqrt{d \frac{2RE_b}{N_0}}\right), \tag{3}$$

where P_{FER} is the frame or block error rate, $Q\left(\cdot\right)$ is the upper tail probability of a normalized Gaussian random variable, R is the code rate, E_b is the energy per information bit and N_0

is the single-sided noise spectral density. Note that the bound (3) is valid for the ensemble of all interleavers with a certain length; a single specific interleaver may perform both worse and better than the ensemble upper bound. For the bit error rate performance, the corresponding union bound is

$$P_{BER} \leq \sum_{w} \sum_{j} \frac{w}{N} A_{w,j}^{TC} Q\left(\sqrt{(w+j)\frac{2RE_b}{N_0}}\right). \quad (4)$$

When deriving the CWEF of the constituent codes of Turbo codes, it is common to take only the error events that end up in the zero state into account. This corresponds to taking only zero-terminating input sequences into consideration. Figure 2 depicts such a combination of an interleaver and an input sequence, for which a weight-3 input sequence is zero-terminating both before and after interleaving. Depending on the method of trellis termination, codewords might also exist that originate from trellis paths that do not end up in the zero state after N trellis transitions. Since these codewords can be thought of as being the result of the truncation imposed by the interleaver, they are referred to as *interleaver edge effects* [12]. In the next section we describe an efficient method for taking the interleaver edge effects into account, for various Turbo code trellis termination methods.

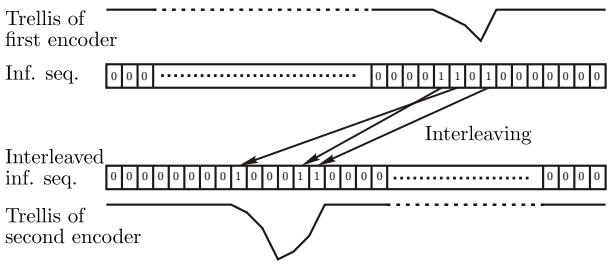

Fig. 2. Example of an interleaver and an information sequence producing two low weight parity sequences, and thus a low weight code word.

A. Interleaver edge effects

Interleaver edge effects refer to the implications on the distance spectrum resulting from the block partitioning of the input sequence, as the result of a limited-length interleaver. Due to this truncation, low weight parity words can be generated even though the encoder input sequences do not force the encoders back to the zero states. In terms of weight enumerating functions, this means that we require knowledge not only of the number of trellis paths that lead to the zero state after the last transition, but also of the number of paths that lead to the other states. This can be obtained by partitioning the IRWEF in (1) into a *state-dependent* counterpart, which enumerates the number of paths that lead to trellis state s, having input weight w and parity weight j. An efficient method to find the state-dependent IRWEF of a convolutional encoder valid after t trellis transitions is to extend the IRWEF of the same encoder obtained for $t-1$ transitions. Let $A_{t,s,w,j}$ denote the number of paths that lead to state s after t trellis transitions, having input weight w and parity weight j. Based on the encoder trellis, the coefficients of the state dependent IRWEF are recursively calculated as

$$A_{t,s,w,j} = \sum_{u=0}^{1} A_{t-1,S(s,u),w-u,j-P(S(s,u),u)},$$

where $S(s,u)$ is the state that leads to state s when the input symbol is u, and $P(S(s,u),u)$ is the parity weight of the corresponding trellis transition. The recursive procedure is initialized with $A_{0,s,w,j} = 0\ \forall s,w,j$, except for $A_{0,0,0,0} = 1$, which corresponds to an encoder initialized in the zero state.

Let $E_{w,j}^{C_1}$ and $E_{w,j}^{C_2}$ denote the multiplicities of codewords that correspond to trellis paths that do not end up in the zero state after encoding length-N input blocks, for constituent code C_1 and C_2 respectively. These multiplicities are dependent on the trellis termination method, and can be calculated using the coefficients of the state dependent IRWEF at time N, i.e. $A_{N,s,w,j}$. With these multiplicities, the overall CWEF of each constituent code, including both zero-terminating codewords and edge effect codewords, is obtained as $\tilde{A}_w^{C_l}(Z) = A_w^{C_l}(Z) + E_w^{C_l}(Z)$, $l = 1,2$, where $E_w^{C_l}(Z) = \sum_{j=0}^{n-N} E_{w,j}^{C_l} Z^j$. Finally, the overall Turbo code CWEF is

$$\tilde{A}_w^{TC}(Z) = \frac{\left(A_w^{C_1}(Z) + E_w^{C_1}(Z)\right)\left(A_w^{C_2}(Z) + E_w^{C_2}(Z)\right)}{\binom{N}{w}}. \quad (5)$$

When comparing different termination methods, it is illustrative to separate the part of the distance spectrum that originates from edge effects, i.e.

$$E_w^{TC}(Z) = \frac{A_w^{C_1}(Z) E_w^{C_2}(Z) + E_w^{C_1}(Z) A_w^{C_2}(Z) + E_w^{C_1}(Z) E_w^{C_2}(Z)}{\binom{N}{w}}, \quad (6)$$

so that $\tilde{A}_w^{TC}(Z) = A_w^{TC}(Z) + E_w^{TC}(Z)$. The difference in the $\tilde{A}_w^{TC}(Z)$'s for different trellis termination methods is in the way the $E_{w,j}^{C_l}$'s are calculated. This is treated in the subsequent paragraphs.

We separate the trellis termination methods into three principal classes:

1) No termination of either constituent encoder.
2) Termination of the first constituent encoder.
3) Termination of both constituent encoders.

Note that these classes refer to the trellis situations after encoding sequences that are N bits long, i.e. the length of the interleaver.

Class I. No trellis termination: With *no trellis termination* of either constituent encoder, the multiplicities of codewords that correspond to interleaver edge effects are calculated by summing the number of paths that end in non-zero states after N trellis transitions, i.e.

$$E_{w,j}^{C_1} = \sum_{s=1}^{2^{m_1}-1} A_{N,s,w,j}^{C_1}, \text{ and}$$

$$E_{w,j}^{C_2} = \sum_{s=1}^{2^{m_2}-1} A_{N,s,w,j}^{C_2},$$

where m_1 and m_2 are the number of memory elements in encoder 1 and 2, respectively. The overall distance spectrum

including edge effect codewords, $\tilde{A}_w^{TC}(Z)$, as well as the edge effect distance spectrum only, $E_w^{TC}(Z)$, is calculated using (5) and (6).

An important and frequently used subclass of no trellis termination is *post interleaver flushing*, proposed in [7]. With this method, both encoders are flushed independently of each other, after encoding their N-bit input sequences. The combination of the weight spectra of the constituent encoders is then similar to the case of no trellis termination, since the trellises are not terminated by the end of their length-N input sequences. However, extra codeword weight is added as a consequence of the encoder flushings. This is accounted for by adding the weight of the flush bits and the corresponding parity bits to the parity weight in the IRWEFs. More precisely,

$$E_{w,j}^{C_1} = \sum_{s=1}^{2^{m_1}-1} A_{N,s,w,j-F_1(s)}^{C_1}$$
$$E_{w,j}^{C_2} = \sum_{s=1}^{2^{m_2}-1} A_{N,s,w,j-F_2(s)}^{C_2},$$

where $F_l(s)$, $l = 1, 2$, is the sum of the weight of the flush bits and parity bits generated when forcing the encoder to the zero state from state s. Note that these flush- and parity bits results in a rate loss, corresponding to the extra $2m_1 + 2m_2$ bits transmitted.

Class II. Termination of the first encoder: By appending m_1 tail bits to the input sequence so that the first encoder is terminated in the zero state, the edge effect codewords are entirely removed from the first encoder. Note that the tail bits are included in the sequence that enters the interleaver, and that their Hamming weight is included in the input weight w. For the second encoder, the situation is identical to the case of no trellis termination. Hence,

$$E_{w,j}^{C_1} = 0$$
$$E_{w,j}^{C_2} = \sum_{s=1}^{2^{m_2}-1} A_{N,s,w,j}^{C_2}.$$

Class III. Termination of both encoders: It is also possible to terminate both constituent encoders in the zero states. Two different ways of achieving this are reported in the literature:

1) By imposing certain interleaver restrictions, the second encoder can be forced to end up in the same state as the first encoder [4], [5]. It is then sufficient to append tail bits according to termination class II, in order to terminate both encoders in the zero states. The necessary interleaver restrictions are

$$\pi(i) = i \pmod{L}, \qquad i = 1, 2, \ldots, N, \tag{7}$$

where $\pi(i)$ is the position of input bit i after interleaving, and L is the period of the impulse response of the constituent encoders (assumed to be identical). Following the terminology in [5], this method is referred to as using *self-terminating interleavers*.

2) By identifying specific, interleaver dependent, input positions it is possible to force the constituent encoders to their zero states independently of each other [3]. This is achieved without any restrictions on the choice of interleaver, but with a slight increase in the number of input bits dedicated for trellis termination (k termination bits are required, where $\max(m_1, m_2) \leq k \leq m_1+m_2$). Following the terminology in [3], this method is referred to as *dual termination*.

Even though the second method results in a slightly higher rate loss, it has the advantage of not imposing restrictions on the interleaver design.

With both encoders terminated in their zero states, all edge effect codewords are entirely removed. Consequently, $E_{w,j}^{C_1} = E_{w,j}^{C_2} = 0$, so that $E_w^{TC}(Z) = 0$ and $\tilde{A}_w^{TC}(Z) = A_w^{TC}(Z)$.

Figure 3(a) shows the lower parts of the edge effect distance spectra for the described termination methods, for a Turbo code with interleaver length 500 bits and constituent encoders with feedback and feedforward polynomials 15_8 and 17_8, respectively. As expected, "no termination" results in the worst distance spectrum, significantly improved by terminating the first encoder and post interleaver flushing. Naturally, since "termination of both encoders" entirely removes the interleaver edge effects, there exist no edge effect codewords. Figure 3(b) shows the corresponding total distance spectra, including both ordinary and edge effect codewords. Figure 4 shows the union bound on the frame error rates achieved with (3), corresponding to the distance spectra in Figure 3(b). As expected, the union bound diverges at a signal-to-noise ratio close to the cut-off rate, which is 2.03 dB for rate-1/3 coding. Also shown in Figure 4 are the simulated performances of randomly chosen interleavers, using the described trellis termination methods. For each transmitted frame, a new pseudo-random interleaver is chosen. The simulated performances rank in the same order as predicted by the derived distance spectra; however, the simulation results are actually *above* the derived union bounds. This is not an inconsistency, bearing in mind that the union bound is valid for maximum likelihood decoding, while the simulation results are obtained with suboptimal iterative decoding.

III. Interleaver edge effects with non-uniform interleavers

Even though calculating the average distance spectrum for the different termination methods gives a good insight to the issues of trellis termination, it is important to keep in mind what the method does – i.e. the averaging over the entire ensemble of interleavers having a certain length. As an example of when the averaging method coincides poorly with individual interleaver performance, we study in this section ordinary block interleavers. With such interleavers, in which the input sequence are written row-wise and read column-wise, the very last input position remains at the last position also after interleaving. Hence, with no trellis termination, the minimum distance of the code is 3 regardless of the interleaver length and the choice of component encoders. If post interleaver flushing is employed, the minimum distance is increased to 13 (for feedback and feedforward polynomials 15_8 and 17_8, respectively), again regardless of the interleaver length. Even though 13 is considerably larger than 3, it is

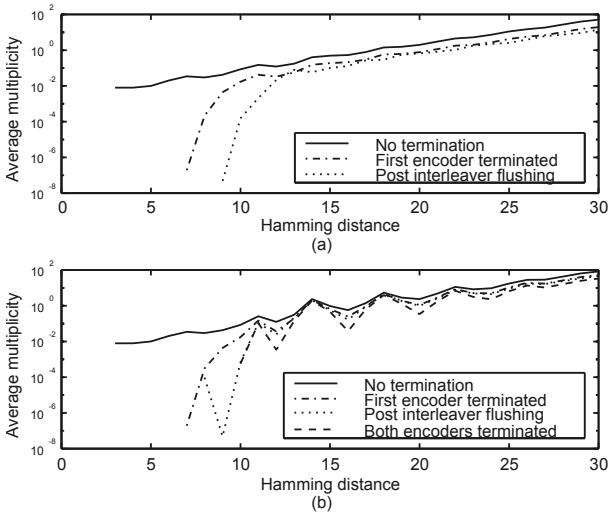

Fig. 3. Average distance spectra of Turbo codes using the ensemble of all 500-bit interleavers, for four different trellis termination methods. The constituent encoders use feedback and feedforward polynomials 15_8 and 17_8, respectively. (a) The part of the distance spectra that result from interleaver edge effects only. (b) The entire distance spectra, including both edge effect codewords and ordinary codewords for which both encoders end in the zero state.

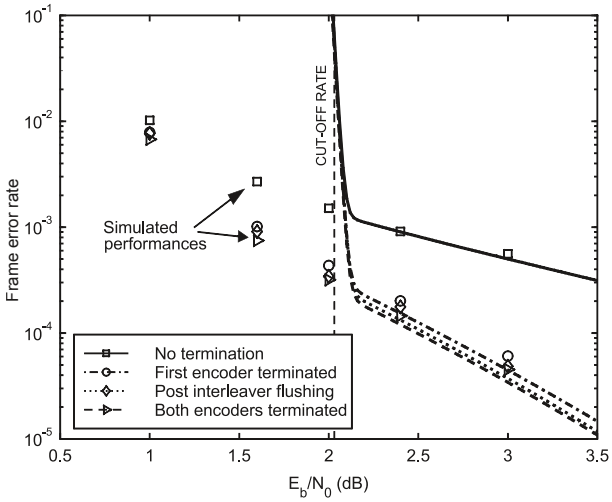

Fig. 4. Union bound of frame error rate performance achieved with the different trellis termination methods, for 500-bit interleavers. The constituent encoders use feedback and generator polynomials 15_8 anf 17_8, respectively. Also shown are simulated performances, achieved by simulating a large number of randomly chosen interleavers.

still a poor minimum distance for interleavers of reasonable lengths, say above 100 bits. As a comparison, the minimum distances when *terminating the first encoder* or *terminating both encoders* is 28 (with multiplicity 473), when using a 500-bit block interleaver with 20 rows and 25 columns. Figure 5 shows simulated frame error rate performances for these codes, along with their respective free distance asymptotes [18]. The poor performance for no trellis termination follows, as described, from the severe edge effect that an ordinary block interleaver results in.

The particular influence from the edge effect that arises when using ordinary block interleavers can be singled out by

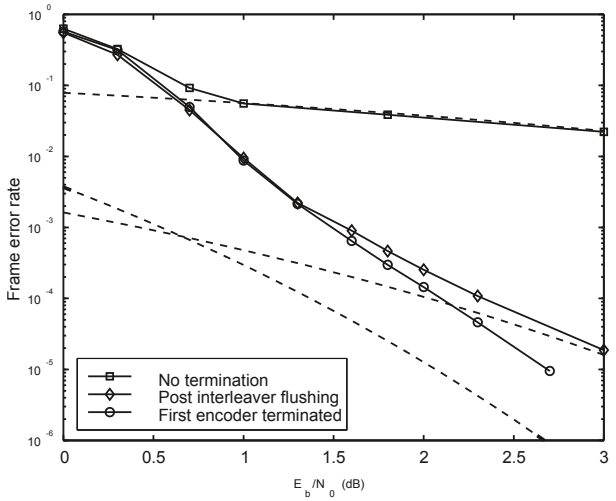

Fig. 5. Simulated frame error performance for Turbo codes using 500-bit block interleavers (20 rows by 25 columns) and feedback and parity polynomials 15_8 and 17_8. Three termination methods are shown: no termination, post interleaver flushing, and termination of the first encoder.

comparing with the performance achieved with reversed block interleavers [19]. When using reversed block interleavers, the interleaved sequence is read in reversed order, starting from the last column and the last row. This procedure effectively removes the interleaver edge effects, thereby significantly improving the performance of no termination and post interleaver flushing. The performance of a 500-bit reversed block interleaver, using the different termination methods, are shown in the upper part of Figure 6. As can be seen, there is essentially no difference in the performance of the different termination methods.

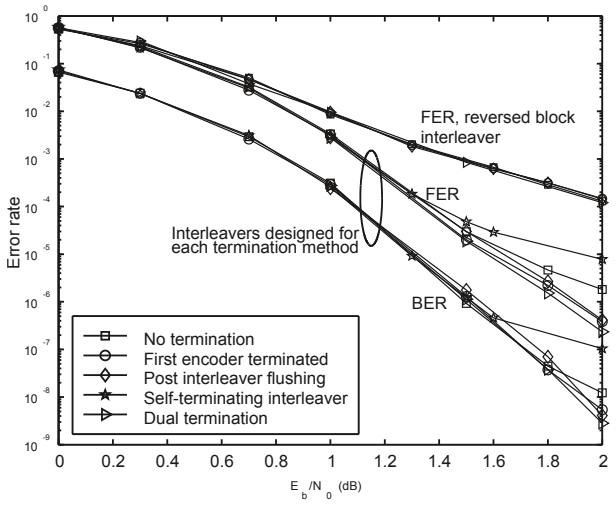

Fig. 6. Error rate performance of Turbo codes using 500-bit interleavers that do not result in interleaver edge effects.The constituent encoders use feedback and feedforward polynomials 15_8 and 17_8, respectively.

It is well known that it is possible to design interleavers that perform better than both ordinary block interleavers and reversed block interleavers. Most interleaver design methods described in the literature are based on design criteria that strive to avoid interleaver mappings that correspond to

low weight codewords produced by zero-terminating input sequences, as exemplified in Figure 2. However, interleavers designed without consideration to the edge effects may result in low weight edge effect codewords. As a consequence, comparisons between different trellis termination methods may be corrupted due to the stochastic presence of the edge effects.

In order to evaluate the performance of the trellis termination methods when using interleavers with good performance, we have designed interleavers with specific criteria that target the interleaver edge effects. The edge effect criteria are used as a supplement to an interleaver design algorithm that use both ordinary distance spectrum criteria (such as in [20]), as well as a criterion based on the correlation properties of the extrinsic information [21]. The interleaver design restrictions that avoid interleaver edge effects are described in Appendix A. In essence, the vector that represents the interleaver is assigned element by element. For each new assignment, the positions that result in codewords with highest weight are located (in this process, all positions that results in codeword weights larger than d_{design} are included in the "highest weight" set). Among those positions, the one with the best correlation properties are chosen as assignment of the interleaver element in question. Thus, the distance spectrum criterion has precedence over the correlation criterion. A flow chart diagram of the design algorithm is shown in Figure 7; further information on the algorithm, as well as performance comparisons with other algorithms, can be found in [22].

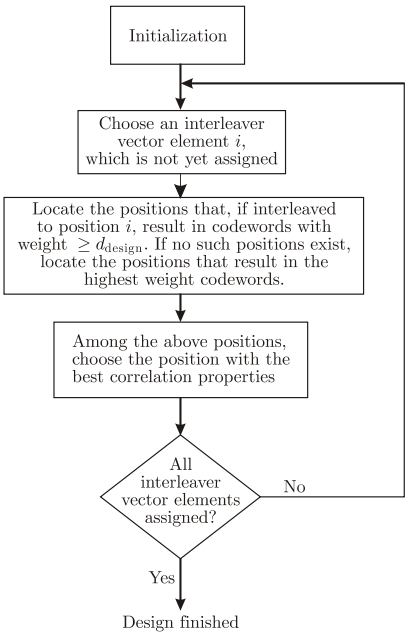

Fig. 7. Flow-chart diagram of the interleaver design algorithm.

The performance of 500-bit interleavers designed specifically for each termination method are shown in Figure 6, by the two lower sets of curves. All interleavers were designed using $d_{\text{design}} = 35$. The best performances are achieved with dual termination, termination of the first encoder, and post interleaver flushing. The error floors of these codes are not reached even for bit error rates as low as 10^{-9}. The slight degradation in performance achieved with no termination and self-terminating interleavers are both due to implications regarding the interleaver design. For the case of no termination, the number of edge effects codewords to avoid in the interleaver design becomes very large, resulting in a high computational complexity. On the other hand, when designing self-terminating interleavers, the interleaver restrictions (7) severely limit the design freedom. Therefore, it becomes difficult to design self-terminating interleavers that perform as well as interleavers designed without these restrictions.

Disregarding self-termination, for reasons mentioned above, the performance differences between the remaining codes in Figure 6 are very small, even though they use different termination methods. This, in combination with the fact that the dual termination method does not suffer from edge effects, leads us to conclude that the edge effect codewords have been successfully removed by the respective interleavers. Had this not been the case, error floors due to edge effects would have led to significant performance degradations, especially when using no termination.

An important parameter regarding the computational complexity of interleaver design is the length of the period of the encoder impulse responses, i.e. L. The larger the length of the period L, the smaller the computational complexity required to design an interleaver with a specified minimum distance. Further, the multiplicities of codewords with weights just above the minimum distance will in general be larger for a code with lower L. Figure 8 shows the performance of two Turbo codes that use constituent encoders with 3 and 4 memory elements respectively, both using interleavers designed for no termination with $d_{\text{design}} = 35$. The periods of the encoder impulse responses are 7 and 15, respectively. This indicates that we should expect a better asymptotic performance for the Turbo code with 4 memory elements, which is supported by the performances shown in Figure 8. Note that there in no indication of an error floor for this Turbo code, even though no trellis termination is employed. Further, note that the Turbo code with fewer memory elements, and thus lower decoding complexity, perform better for a rather large and important part of the SNR range. This behavior has been observed in [23], and analysed in for example [24], [25].

IV. Discussion on Decoding Initialization

One of the issues of trellis termination for Turbo codes is the initialization of the backward recursion in the BCJR algorithm, see for example [26]–[28]. In essence, the *a posteriori* probabilities produced by the BCJR algorithm for a certain trellis transition is influenced by three quantities used in decoding, usually denoted α, β and γ. The α- and β-values can be interpreted as state probabilities, while the γ-values represent branch transition probabilities in the trellis. The γ-values at a given time t are based solely on the channel statistics received for symbol t. In contrast, the state probabilities α at time t summarizes the received channel statistics from time 0 up to t. Similarly, the β-values are based on the channel statistics from time $t+1$ till the end of the block, i.e. to time N. The α- and β-values are computed recursively, a procedure referred to as the forward and the backward recursion, respectively.

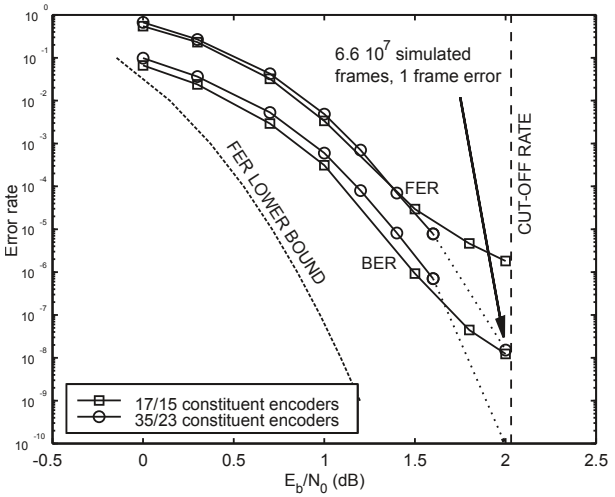

Fig. 8. Performance of Turbo codes without trellis termination, for constitutent encoders with 3 and 4 memory elements. Both encoders use 500-bit interleavers designed for no trellis termination.

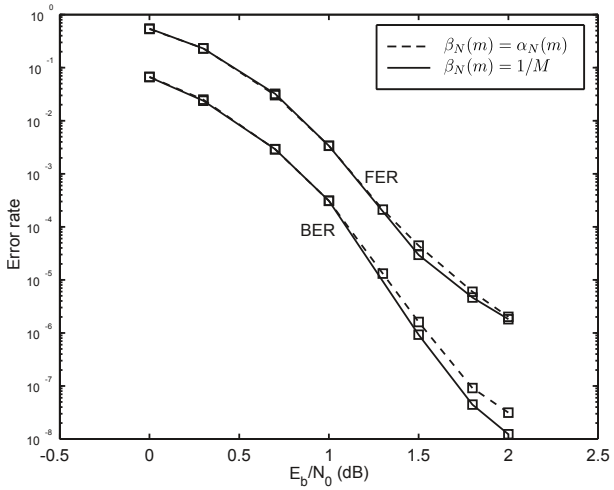

Fig. 9. Comparison of the frame- and bit error rate performance for two different initialization methods for the backward recursion in the BCJR algorithm, when no trellis termination is employed. Interleaver length: 500 bits, encoder polynomials: $(17/15)_8$.

Since the constituent encoders are initialized in their zero states before each information block is encoded, it is straightforward to initialize the α-values for time 0: state zero is assigned probability 1, while all the other states are assigned probability 0. If the constituent encoders are terminated in their zero states, the β-values are initialized in the same manner. However, when using no trellis termination, or termination of the first encoder only, there is at least one trellis for which the final state is unknown.

At least two methods to deal with the state-uncertainty were proposed in the literature. In the original Turbo code paper [1], Berrou et al. proposed to initialize the β-values with equal probabilities, i.e. $1/M$, where M is the number of states in the trellis. Another possibility is to use the final α-values obtained from the forward recursion, as evaluated for example in [26]. However, this method is in violation with the derived expression for the β-values, which states that they should depend only on channel statistics from the present time up to the end of the block. By using the final α-values for initialization, the β-values become dependent on the channel statistics received during the entire block.

Figure 9 shows simulated error rate performance of the previously used Turbo code using a 500-bit interleaver with no trellis termination and constituent encoders with polynomials 15_8 (feedback) and 17_8 (feedforward). The initialization for the backward recursion, i.e. the β-values, are performed with the aforementioned methods. For low signal-to-noise ratios, there is no detectable difference in the error correcting performance. However, as previously observed in [27], at medium and high SNRs the decoder that uses equiprobable initial states perform better than the decoder that uses the final α-values for initialization, especially in terms of bit error rate. These simulations are in agreement with the above reasoning, claiming that the initial β-values should not depend on the α-values from the forward recursion.

V. Conclusions

In this paper, we have discussed central aspects of trellis termination methods for Turbo codes. An efficient method to calculate Turbo code distance spectra for the ensemble of interleavers, including the effects of the choice of termination method, has been presented. This method illustrates the result of *interleaver edge effects*, thus allowing for basic understanding of the properties that govern the performance of a specific termination method. For the ensemble of interleavers, it is evident that it is essential for the code performance that some kind of trellis termination is used.

The performance differences between commonly used termination methods, such as *termination of the first encoder only, post interleaver flushing,* and *dual termination*, is in general small. However, examples are shown for which the differences are substantial, explained by the fact that the different methods are not equally sensitive to interleaver edge effects. Hence, we stress the importance of the combination of the choice of trellis termination method and the choice of interleaver. In particular, we present interleaver design guidelines that are tailored to the chosen termination method. It is demonstrated that for interleavers designed with such design criteria, the performance of the different termination methods are nearly indistinguishable, also when using no trellis termination at all. In fact, we demonstrate Turbo codes using no trellis termination and 500–bit interleavers that show no sign of an error floor, even at frame error rates as low as 10^{-8}.

Appendix A

In this appendix we present interleaver design restrictions that aim to avoid interleaver edge effects. The restrictions on the interleaver design depends on the chosen method of trellis termination. To start with, since termination of both encoders completely removes all edge effects, this method requires no interleaver design restrictions. For the other two classes of

trellis termination, the interleaver design includes restrictions based on the particular choice of termination method.

For the purpose of describing the interleaver design restrictions, we distinguish between three principal types of interleaver edge effects: (Type I) truncation where the first encoder is in a non-zero state, (Type II) truncation where the second encoder is in a non-zero state, and (Type III) truncation where both encoders are in non-zero states. The three types of edge effects are illustrated in Figure 10.

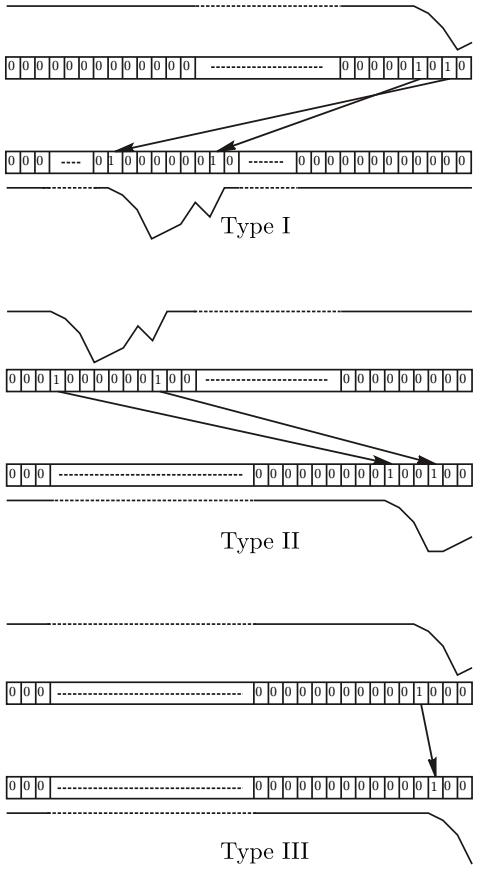

Fig. 10. Examples of three types of interleaver edge effects: truncation of the first trellis in a non-zero state (Type I), truncation of the second trellis in a non-zero state (Type II), and truncation of both trellises in non-zero states (Type III).

Let the component encoders be represented by the generators $G_1(D) = \frac{G_{1p}(D)}{G_{1f}(D)}$ and $G_2(D) = \frac{G_{2p}(D)}{G_{2f}(D)}$, respectively, and their input sequences by $X(D)$ and $X'(D)$ ($X'(D)$ is the interleaved version of $X(D)$). All zero-terminating input sequences are divisible by the feedback polynomial, i.e. $X(D) = Y(D) G_f(D)$, where $\deg Y(D) \leq \deg X(D)$. Thus, the parity sequence is $X(D) G_1(D) = Y(D) G_{1p}(D)$. Further, let $i_1, i_2, i_3, \ldots$ and $i'_1, i'_2, i'_3, \ldots$ denote the positions of the first, second, third, and so on, input ones before and after interleaving, so that $X(D) = \sum_{k=0}^{N-1} D^{i_k}$ and $X'(D) = \sum_{k=0}^{N-1} D^{i'_k}$. Finally, the Hamming weight of a sequence, or the corresponding polynomial representation, is denoted $w_H(\cdot)$.

The total codeword weight of a Turbo code codeword is

$$d = w + w_H\left(\langle X(D) G_1(D)\rangle_N\right) + w_H\left(\langle X'(D) G_2(D)\rangle_N\right),$$

where w is the input weight, and $\langle\cdot\rangle_N$ denotes truncation of all terms in a polynomial for which the D-exponent is larger than or equal to N. The complexity required to check (and avoid) all the necessary interleaver mappings grows rapidly with increasing input weight w. Fortunately, the probability of edge effects resulting in low weight codewords decreases with increased input weight. This is especially true when designing interleavers with some sort of spreading criterion, such as the S-random design [12] or the correlation criterion [21]. With such criteria, we have found it sufficient to check for mappings of input sequences with weight $w = 2, 3$ and 4. Further, the spreading criterion relieve the number of input sequences that need to be checked for $w = 3$ and 4. Consequently, the interleaver design restrictions we present in the following relies on a spreading criterion being used in the interleaver design.

Figure 11 shows the principal appearance of Type I edge effect interleaver mappings that a spreading criterion fails to avoid. For all such input sequences, the interleaver design should ensure that

$$w_H\left(\langle X(D) G_1(D)\rangle_N\right) + w_H\left(X'(D) G_2(D)\right) < d_{\text{design}} - w,$$

where d_{design} is the targeted minimum distance of the overall Turbo Code. The input sequences $X(D)$ and $X'(D)$ are constrained to be of the forms given in the upper part of Table I, which summarizes the characteristics of Type I edge effect input sequences. For weight-2 and weight-4 input sequences, the sequence characterization is based solely on the periods of the encoder impulse responses, i.e. L_1 and L_2. For weight-3 inputs, the zero-terminating sequences are instead characterized by a set of fundamental zero-terminating weight-3 sequences. All the existing weight-3 zero-terminating input sequences can be expressed using one of these fundamental sequences. These are in turn represented by the sets of constants $\phi_{l1}$and $\phi_{l2}, l = 1, 2, \ldots, l_{\max}$where $l_{\max}$ is the number of such fundamental sequences that exist for a certain feedback polynomial. More precisely, every zero-terminating weight-3 input can be written as $D^{k_0}\left(1 + D^{\phi_{l1}+k_1 L} + D^{\phi_{l2}+k_2 L}\right)$, where k_0, k_1 and k_2 are integers. Values of ϕ_{l1}, ϕ_{l2}, for the most common encoders with 2–4 memory elements, are given in Table II. For example, an encoder with feedback polynomial $1+D^3+D^4$ (23 in octal representation), every zero-terminating weight-3 input sequence can be written on one of the forms $D^{k_0}\left(1 + D^{1+7k_1} + D^{12+7k_2}\right)$, $D^{k_0}\left(1 + D^{2+7k_1} + D^{9+7k_2}\right)$ or $D^{k_0}\left(1 + D^{5+7k_1} + D^{10+7k_2}\right)$.

Type II and Type III edge effects are treated the same way as Type I edge effects. The characterization of the input sequences that correspond to Type II and Type III edge effects is given in the middle and lower part of Table I.

The interleaver design constraints presented here can be implemented with sufficiently low computational complexity to allow for design of large interleavers. The most demanding constraints arise when *no trellis termination* is used, since all the types of edge effects must be avoided. When terminating the first encoder, both Type I and Type III edge effects are removed, thus reducing the interleaver design complexity considerably.

	w	Characteristics of $X(D)$	Characteristics of $X'(D)$
Type I	2	$i_2 - i_1 \bmod L_1 \neq 0$	$i'_2 - i'_1 \bmod L_2 = 0$
	3	$i_2 - i_1 \bmod L_1 = 0$	$D^{i'_1} + D^{i'_2} + D^{i'_3} = D^{k_0}\left(1 + D^{\phi_{l1}+k_1 L_2} + D^{\phi_{l2}+k_2 L_2}\right)$
	4	$i_2 - i_1 \bmod L_1 = 0$ $i_4 - i_3 \bmod L_1 \neq 0$	$i'_2 - i'_1 \bmod L_2 = 0$ $i'_4 - i'_3 \bmod L_2 = 0$
Type II	2	$i_2 - i_1 \bmod L_1 = 0$	$i'_2 - i'_1 \bmod L_2 \neq 0$
	3	$D^{i_1} + D^{i_2} + D^{i_3} = D^{k_0}\left(1 + D^{\phi_{l1}+k_1 L_1} + D^{\phi_{l2}+k_2 L_1}\right)$	$i'_2 - i'_1 \bmod L_2 = 0$
	4	$i_2 - i_1 \bmod L_1 = 0$ $i_4 - i_3 \bmod L_1 = 0$	$i'_2 - i'_1 \bmod L_2 = 0$ $i'_4 - i'_3 \bmod L_2 \neq 0$
Type III	2	$i_2 - i_1 \bmod L_1 \neq 0$	$i'_2 - i'_1 \bmod L_2 \neq 0$
	3	$i_2 - i_1 \bmod L_1 = 0$	$i'_2 - i'_1 \bmod L_2 = 0$
	4	$i_2 - i_1 \bmod L_1 = 0$ $i_4 - i_3 \bmod L_1 \neq 0$	$i'_2 - i'_1 \bmod L_2 = 0$ $i'_4 - i'_3 \bmod L_2 \neq 0$

TABLE I

CHARACTERIZATION OF LOW WEIGHT INPUT SEQUENCES THAT CAN CAUSE LOW WEIGHT EDGE EFFECT CODEWORDS, NOT AVOIDED BY USING A SPREADING CRITERION IN THE INTERLEAVER DESIGN.

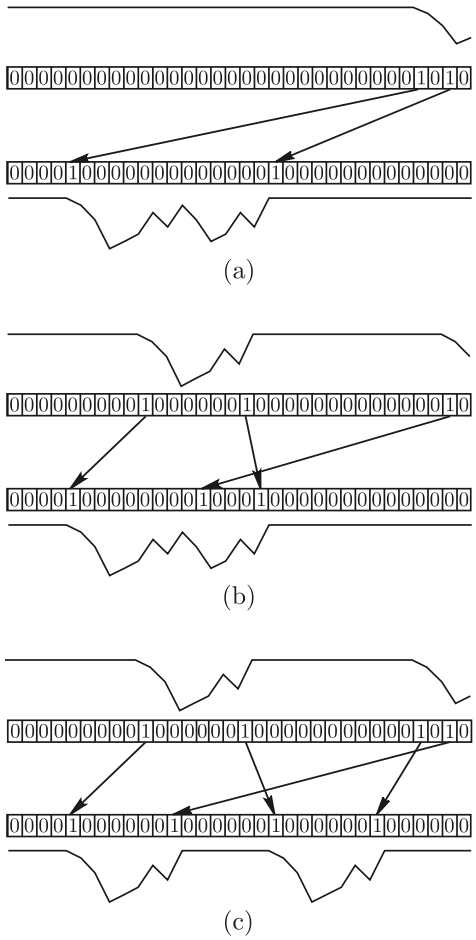

Fig. 11. Examples of Type I edge effect interleaver mappings not avoided by a spreading interleaver design criterion, for (a) weight-2, (b) weight-3, and (c) weight-4 input sequences.

m	G_f	L	l	ϕ_{l1}	ϕ_{l2}
2	5	2	-	-	-
	7	3	1	1	2
3	11	3	-	-	-
	13	7	1	1	5
	15	7	1	1	3
	17	4	-	-	-
4	21	4	-	-	-
	23	15	1	1	12
			2	2	9
			3	5	10

m	G_f	L	l	ϕ_{l1}	ϕ_{l2}
4	25	6	1	2	4
	27	7	-	-	-
	31	15	1	1	4
			2	2	8
			3	5	10
	33	6	-	-	-
	35	7	-	-	-
	37	5	-	-	-

TABLE II

FUNDAMENTAL ZERO-TERMINATING WEIGHT-3 SEQUENCES FOR RECURSIVE ENCODERS WITH FEEDBACK POLYNOMIAL G_f (IN OCTAL REPRESENTATION).

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo Codes," in *Proc. 1993 IEEE International Conference on Communication (ICC)*, pp. 1064–1070, Geneva, Switzerland, May 1993.

[2] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-Codes," *IEEE Transactions on Communications*, vol. 44, pp. 1261–1271, October 1996.

[3] P. Guinand and J. Lodge, "Trellis termination for turbo encoders," in *17th Biennial Symp. On Communications*, pp. 389–392, Kingston, Canada, May 1994.

[4] A. S. Barbulescu and S. S. Pietrobon, "Terminating the trellis of turbo-codes in the same state," *Electronics Letters*, vol. 31, pp. 22–23, January 1995.

[5] M. Hattori, J. Murayama, and R. J. McEliece, "Pseudo-random and self-terminating interleavers for turbo codes," in *Winter 1998 Information Theory Workshop*, pp. 9–10, San Diego, USA, February 1998.

[6] S. Crozier, P. Guinand, J. Lodge, and A. Hunt, "Construction and performance of new tail-biting turbo codes," in *6-th International Workshop on Digital Signal Processing Techniques for Space Applications (DSP '98)*, Noordwijk, The Netherlands, September 1998.

[7] D. Divsalar and F. Pollara, "Turbo codes for PCS applications," in *IEEE International Conference on Communications*, New York, USA, 1995.

[8] M. C. Reed and S. S. Pietrobon, "Turbo-code termination schemes and a novel alternative for short frames," in *Seventh IEEE International Symposium on Personal, Indoor and Mobile Communications*, New York, USA, 1996.

[9] J. Hokfelt, O. Edfors, and T. Maseng, "A survey on trellis termination alternatives for turbo codes," in *IEEE Vehicular Technology Conference*, pp. 2225–2229, Houston, Texas, USA, May 1999.

[10] W. J. Blackert, E. K. Hall, and S. G. Wilson, "Turbo code termination and interleaver conditions," *Electronics Letters*, vol. 31, pp. 2082–2084, November 1995.

[11] O. Joerssen and H. Meyr, "Terminating the trellis of turbo codes," *Electronics Letters*, vol. 30, pp. 1285–1286, August 1994.

[12] S. Dolinar and D. Divsalar, "Weight distributions for turbo codes using random and nonrandom permutations." TDA progress report 42-122, Jet propulsion Lab., Pasadena, CA, August 1995.

[13] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of

linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, pp. 284–286, March 1974.

[14] J. Hagenauer and P. Hoeher, "A Viterbi algorithm with soft-decision outputs and its applications," in *Globecom 1989*, pp. 1680–1686, Dallas, Texas, USA, November 1989.

[15] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes," *IEEE Transactions on Information Theory*, vol. 42, pp. 429–445, March 1996.

[16] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *IEEE International Conference on Communications*, pp. 1009–1013, Seattle, USA, 1995.

[17] S. Benedetto and G. Montorsi, "Unveiling turbo codes: Some results on parallel concatenated coding schemes," *IEEE Transactions on Information Theory*, vol. 42, pp. 409–428, March 1996.

[18] L. Perez, J. Seghers, and D. J. Costello, Jr., "A distance spectrum interpretation of turbo codes," *IEEE Transactions on Information Theory*, vol. 42, pp. 1698–1709, November 1996.

[19] H. Herzberg, "Multilevel turbo coding with short interleavers," *IEEE Journal on Selected Areas in Communications*, pp. 303–309, February 1998.

[20] J. Hokfelt and T. Maseng, "Methodical interleaver design for turbo codes," in *International Symposium on Turbo Codes & Related Topics*, pp. 212–215, Brest, France, September 1997.

[21] J. Hokfelt, O. Edfors, and T. Maseng, "A turbo code interleaver design criterion based on the performance of iterative decoding," *IEEE Communications Letters*, In press.

[22] J. Hokfelt, *On the Design of Turbo Codes*. PhD thesis, Lund University, Lund, Sweden, August 2000. ISBN 91-7874-061-4 (http://www.tde.lth.se/home/jht/publications.html).

[23] O. Y. Takeshita, O. M. Collins, P. C. Massey, and D. J. Costello, Jr., "A note on asymmetric Turbo-codes," *Communications Letters*, vol. 3, pp. 2082–2084, March 1999.

[24] S. ten Brink, "Iterative decoding trajectories of parallel concatenated codes," in *IEEE/ITG Conference on Source and Channel Coding*, Munich, Germany, January 2000.

[25] H. E. Gamal and A. R. Hammons, Jr., "Analyzing the turbo decoder using the gaussian approximation," in *IEEE International Symposium on Information Theory*, p. 319, Sorrento, Italy, June 2000.

[26] P. Robertson, "Illuminating the structure of code and decoder of parallel concatenated recursive systematic (turbo) codes," in *Proceedings Globecom '94*, pp. 1298–1303, Dec. 1994.

[27] M. C. Reed and S. S. Pietrobon, "Turbo-code termination schemes and a novel alternative for short frames," in *Proc. International Symposium on Personal, Indoor and Mobile Radio Communications 1996*, Taipei, Taiwan, October 1996.

[28] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Soft-output decoding algorithms for continuous decoding of parallel concatenated convolutional codes," in *Proceedings of International Conference on Communications, ICC 1996*, pp. 112–117, 1999.