Parallel Decoding of Turbo Codes Using Multi-point Trellis Termination and Collision-Free Interleavers

Mustafa Taskaldiran*, Richard C.S. Morling*, and Izzet Kale*
*Applied DSP and VLSI Research Group
Department of Electronic, Communication and Software Engineering
University of Westminster, London, W1W 6UW, United Kingdom
{m.taskaldiran, morling, kalei}@wmin.ac.uk

Abstract— **The UMTS turbo encoder is composed of parallel concatenation of two Recursive Systematic Convolutional (RSC) encoders which start and end at a known state. This trellis termination directly affects the performance of turbo codes. This paper presents performance analysis of multi-point trellis termination of turbo codes which is to terminate RSC encoders at more than one point of the current frame while keeping the interleaver length the same. For long interleaver lengths, this approach provides dividing a data frame into sub-frames which can be treated as independent blocks. A novel decoding architecture using multi-point trellis termination and collision-free interleavers is presented. Collision-free interleavers are used to solve memory collision problems encountered by parallel decoding of turbo codes. The proposed parallel decoding architecture reduces the decoding delay caused by the iterative nature and forward-backward metric computations of turbo decoding algorithms. Our simulations verified that this turbo encoding and decoding scheme shows Bit Error Rate (BER) performance very close to that of the UMTS turbo coding while providing almost %50 time saving for the 2-point termination and %80 time saving for the 5-point termination.**

I. INTRODUCTION

Turbo codes were first introduced in 1993 as a parallel concatenation of two RSC encoders [1]. Because of their excellent performance near the Shannon limit, they have found applications in the Consultative Committee for Space Data Systems (CCSDS), 3GPP/UMTS standard, Digital Video Broadcasting Return Channel Satellite and Terrestrial (DVB-RCS and DVB-RCT), 3GPP2/cdma2000 wireless communication systems, and IEEE.802.16 WiMAX standards [2].

A typical turbo encoder consists of two RSC encoders separated by an interleaver as shown in Figure 1. The UMTS turbo code standard requires the constituent turbo encoders start and end at a known state (all-zero-state). This known state information is used at the receiver side to start the decoding process. Terminating both of the RSC encoders at the all-zero-state can only be achieved by sending a certain tail-bit sequence for each encoder because of the feedback loop in the RSC encoders. The tail-bits for each RSC encoder depend on the generator polynomial and also the input data sequence. Therefore, the required tail-bits to terminate each of the RSC encoders will be different. The tail-bits information should also be send together with the turbo encoded code words.

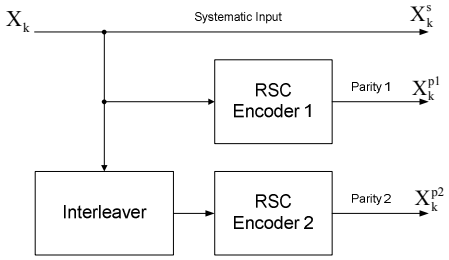

Figure 1. The parallel-concatenated turbo encoder.

There are a number of different trellis termination methods in the literature, such as terminating the constituent RSC encoders, terminating only one of the encoders or using no termination at all, and tail-biting turbo codes which determines the starting and final states by choosing the first state as a function of input sequence and trellis structure [3]. Two main problems about trellis termination are degrading decoding performance near the end of a data sequence and the effect of trellis termination on the distance spectrum of the code [3]. Another fact about turbo code trellis termination methods is that their performance is highly dependent on the interleaver employed by the turbo encoder. This dependency results from the so-called *interleaver edge effects* which deteriorates the distance spectrum of the code [3],[4].

In this paper, multi-point trellis termination of turbo codes is presented; its performance is compared with the UMTS turbo code standard, a novel decoding architecture based on this encoding scheme to reduce decoding delay is presented and further utilization of the multi-point trellis termination at the decoder side is discussed.

Author's of [5] proposed "frequent trellis termination" for adaptively changing the code rate and signal synchronization purposes. The upper constituent RSC encoder (see Figure 1) is frequently terminated while the second RSC encoder is terminated only at the end of the trellis. They claimed that better BER performance is observed comparing to the standard turbo coding. The idea of frequent trellis termination was reported in [5] before the authors of [6] presented it with a different name as "frame

4244-2589-1/09/$20.00 ©2009 IEEE

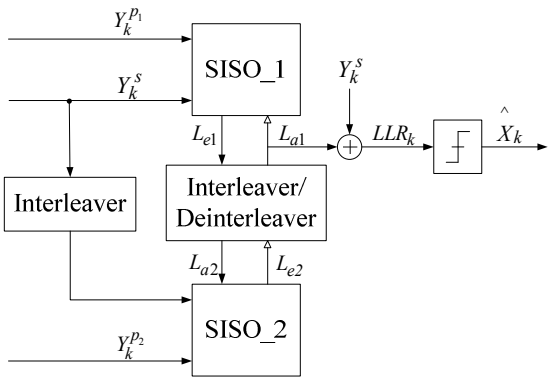

Figure 2. The generic turbo decoder.

splitting and trellis termination". In [6], turbo decoding delay was claimed to be reduced approximately on the order of 1/N where N is the number of sub-frames obtained by terminating both the RSC encoders N times. This reduction is obtained by parallel processing of N sub-blocks at the same time since their final states are known. This decoding approach requires a "combiner unit" to combine the extrinsic Log-Likelihood Ratios (LLRs) before (de)interleaving and a "segmenting unit" to decompose the whole frame into sub-blocks after (de)interleaving [6].

The next section gives a brief overview of the conventional and parallel turbo decoding methods. Section III presents multi-point trellis termination (which was called frequent trellis termination and frame splitting in previous papers). In section IV, simulation results are presented comparing the BER and Frame Error Rate (FER) performance of the multi-point trellis termination with the standard UMTS turbo coding and decoding. In section V, a novel turbo decoder architecture based on the multi-point trellis termination is presented.

II. TURBO DECODING

A general turbo decoder consists of two Soft-Input-Soft-Output (SISO) processors working iteratively on the received data sequence as shown in Figure 2. Each decoder computes an LLR for the k^{th} transmitted data bit d_k, as

$$L(d_k) = \log\left[\frac{\mathrm{P}(d_k = 1 \mid Y)}{\mathrm{P}(d_k = 0 \mid Y)}\right] \quad (1)$$

where Y is the received noisy sequence. The LLR computations can be performed by either Maximum a posteriori probability (MAP) algorithm or Soft-Output Viterbi Algorithm (SOVA). Three metric values are required to compute an LLR:

1. Branch metrics are calculated for each possible trellis transition as

$$\gamma_i(S_{k-1}, S_k) = A_k P(S_k \mid S_{k-1}) \exp\left[\frac{2}{N_o}\left(y_k^s x_k^s(i) + y_k^p x_k^p(i, S_{k-1}, S_k)\right)\right] \quad (2)$$

where $i = (0,1)$, A_k is a constant, S_k is the trellis state at trellis time k, x_k^s and x_k^p are the encoded systematic data bit and parity bit, y_k^s and y_k^p are the received noisy systematic data bit and parity bit respectively.

2. Recursive calculation of the forward state metrics is performed as

$$\alpha_k(S_k) = \sum_{j=0}^{1} \alpha_{k-1}(S_{k-1})\gamma_j(S_{k-1}, S_k) \quad (3)$$

where $\alpha_k(S_k)$ is the forward state metric of state S_k at trellis time k.

3. Similarly, the backward state metrics are calculated by a backward recursion from trellis time $k = N$ down to $k = 1$ as

$$\beta_k(S_k) = \sum_{j=0}^{1} \beta_{k+1}(S_{k+1})\gamma_j(S_k, S_{k+1}) \quad (4)$$

A major problem for practical applications of turbo codes in high-speed digital data communication is the high latency caused by the forward and backward state metric computations. The SISO decoder needs both the forward and the backward state metrics as well as the branch metrics to start computing LLRs. For a long interleaver length, this requirement causes high decoding delays. Therefore, turbo decoding latency should be reduced to meet the increasing demand for high throughputs by current wireless applications [7].

A. Parallel Turbo Decoding

The Log-MAP decoding of a size-N trellis can be completed in N (total frame length) clock cycles if one extrinsic LLR is computed at every clock cycle. The throughput of the turbo decoder can be increased approximately M-times by employing M SISO processors working in parallel. This will basically divide the size-N trellis into M size-W windows (N=WM). The problem of assigning boundary conditions arises here. The conventional decoding uses initial boundary conditions based on the known initial and final state information (all-zero state). For parallel decoding, neighbour windows can be overlapped to compute the boundary conditions for the state metrics [7]. However, this will bring extra computational load during the warm-up period and will also reduce the throughput. In [8], boundary conditions (state metrics) are initialised to 1/number_of_states for the MAP algorithm and updated with iterations by using the state metrics computed by the neighbour window. The decoding latency is reduced to W clock cycles from N clock cycles with almost no performance degradation [8].

An important problem with the parallelism of the Log-MAP decoding is the so called memory collisions. Each sub-processing unit generates one extrinsic LLR to be written into one of the M extrinsic LLR memory units either at interleaved or deinterleaved address locations. During this process, one or more of the parallel processors might try to access to the same memory unit which will cause contentions in memory access [9].

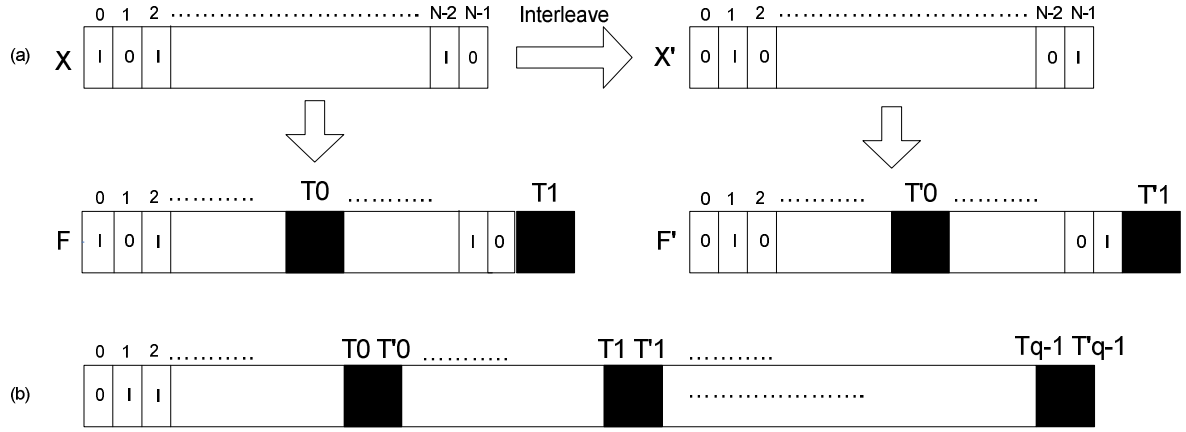

Figure 3. a) Two-point trellis termination of a data frame of size N. X is the data frame, $X^{'}$ is the interleaved frame. F and $F^{'}$ are the corresponding inputs to the RSC encoders with tail-bits inserted in the middle and at the end of the data frame. b) *q* point trellis termination.

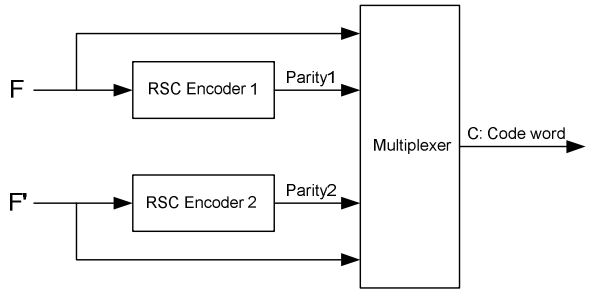

Figure 4. Turbo encoder with multi-point trellis termination

III. MULTI-POINT TERMINATED TURBO ENCODER

Multi-point trellis termination is obtained by forcing the constituent RSC encoders (see Figure 4) to go to a known state (all-zero-state) not only at the beginning and end of the trellis but also at a number of pre-determined trellis points. This approach requires transmitting additional tail-bits at those pre-determined trellis points. For the case of 2-point termination in Figure 3.a (at the middle and end of the frame) code word construction will be as follows:

Code word: $C_k = X_k P_k^1 P_k^2 \qquad k = 0,1,.....N-1$ (5)

Tail bits : $T_j = t_j^0 z_j^0 t_j^1 z_j^1 t_j^2 z_j^2 \qquad j = 0,1$ (6)

(T'_j for the interleaved frame)

Encoded frame: EF_2:$C_0C_1C_2$.....$T_0T_0^{'}$.....$C_{N-1}T_1T_1^{'}$ (7)

The turbo encoded frame terminated at *q* points is shown in Fig.3.b. For the non-punctured UMTS turbo encoder consisting of 8-state RSC encoders, trellis termination requires 12 additional tail-bits to be transmitted. Therefore, for this turbo encoder terminated at *q* points (as in Fig. 3.b), data rate will be

$$r = \frac{N}{3N+12q} = \frac{1}{3+12\frac{q}{N}} \qquad (8)$$

where r is the code rate, *q* is the frequency of trellis termination (number of trellis terminations per frame), and N is the frame length.

IV. SIMULATION RESULTS

The multi-point terminated turbo encoder and the corresponding turbo decoder are simulated according to the UMTS turbo code standard. For simulations, rate-1/3 turbo encoder composed of RSC encoders using generator polynomial (13, 15)oct are used. The decoding algorithm used by Soft-In Soft-Out (SISO) decoders is Log-MAP algorithm. The only difference from the UMTS standard is the extra tail-bits used to terminate two constituent RSC encoders. Fig. 5 and Fig. 6 show the BER and FER performance comparison of the UMTS standard and the multi-point terminated turbo decoding. The interleaver length is kept constant as 1000, the number of iterations is 6 and 2000 frames are transmitted over an Additive White Gaussian Noise (AWGN) channel. One of the multi-point terminated codes is terminated at 2 points (every 500 data bits) while the second one is terminated at 5-points (every 200 bits).

Simulation results show that the BER performance of 2 and 5-point terminated turbo codes are almost the same as the UMTS turbo coding at low Signal-to-Noise Ratio (SNR) region. However, at the high SNRs, BER performance degradation is observed. The BER and FER performance degradation at high SNR can be explained as the increased number of non-turbo coded bits introduced by multi-point termination. From Table 1, it can be observed that a considerable decoding time reduction can be obtained by parallel decoding based on multi-point trellis termination.

Table 1. Decoding delay and performance comparison of standard and multi-point trellis termination.

SISO length	**Time saving**	**BER (at 0.75 dB)**	**FER (at 0.75 dB)**
Regular : 1003	-	$0.77\text{x}10^{-3}$	$3.7\text{x}10^{-2}$
2-point term: 503	%49.85	$0.88\text{x}10^{-3}$	$2.4\text{x}10^{-2}$
5-point term: 203	%79.76	$1.5\text{x}10^{-3}$	$17.9\text{x}10^{-2}$

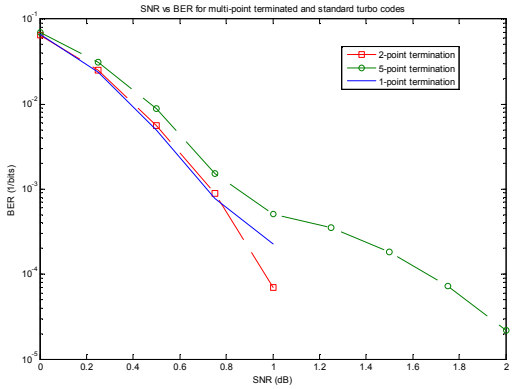

Figure 5. BER performance comparison for different number of trellis terminations. Frame length is 1000 and the number of iterations is 6.

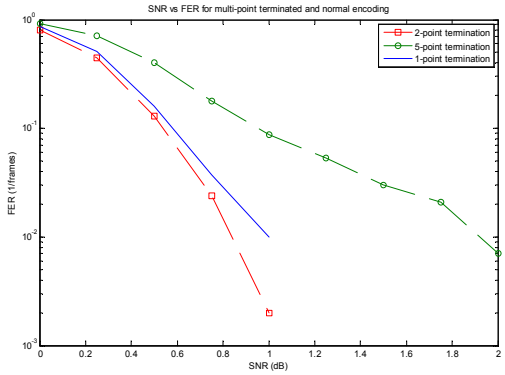

Figure 6. FER performance comparison for different number of trellis terminations. Frame length is 1000 and the number of iterations is 6.

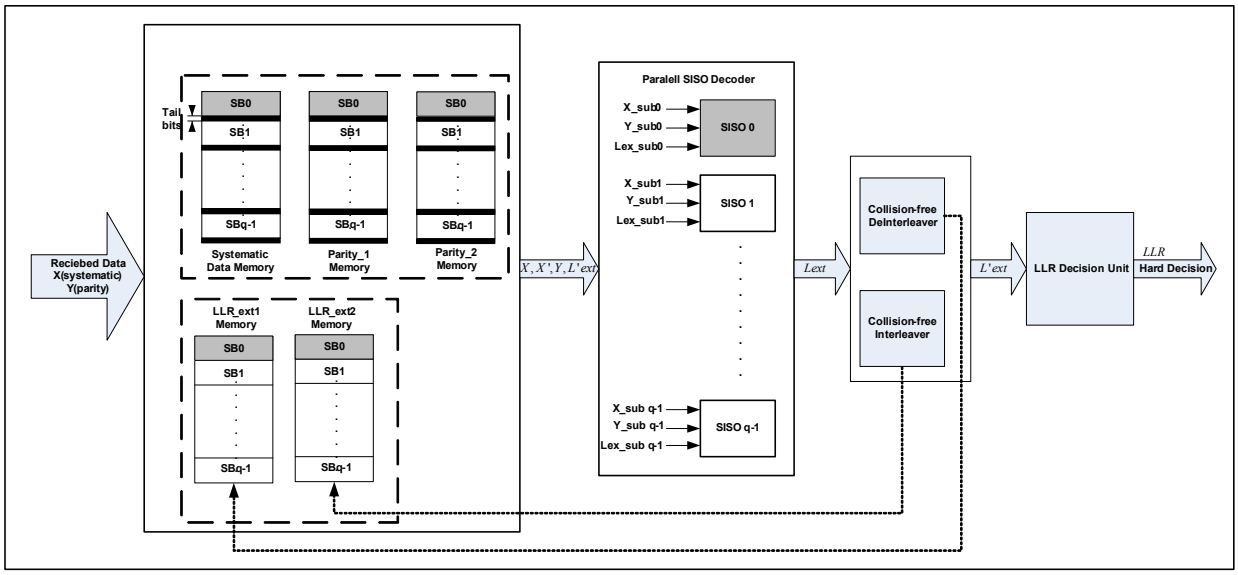

Figure 7. Turbo decoder architecture for the multi-point termination.

V. TURBO DECODER ARCHITECTURE

A novel turbo decoder architecture is presented based on the multi-point trellis termination which can be exploited for low-power and/or high-speed turbo decoding. To increase the decoding speed, each sub-frame terminated with tail bits $T_j T'_j$ (see Fig. 3) can be treated as an independent received block. If trellis termination is applied at q points, q parallel SISO decoders can process q sub-blocks simultaneously as shown in Fig. 7. This parallel processing will increase the decoding operation speed approximately q times as stated in [6]. However, this parallel processing has a major problem of memory collisions caused by the interleaving stage. These memory-collisions happen when more than one extrinsic LLR should be stored in the same memory block [10]. In [6], memory-collisions problem is implicitly solved by the combiner and segmenting blocks which require extra processing before passing the computed extrinsic information to the other SISO processors. A more efficient solution to the memory-collisions problem is to use of collision-free interleavers. These interleavers show a similar BER performance to the UMTS interleaver as reported in [10].

The decoder architecture shown in Fig. 7 employs a collision-free (de)interlevear instead of using combining and segmenting blocks used in [6]. This modification eliminates the requirement for combining and segmenting the extrinsic information. The memory-collisions at the received data and parity memory access, which is not mentioned in [6], can also be eliminated by the use of collision-free interleavers. Each memory unit in Fig. 5 consists of q separate memory blocks for each sub-block (SB) and black regions represent stored tail bits. Each SB can be treated as a separate block to be processed in parallel by q dedicated SISO decoders. The extrinsic LLRs are easily stored in either interleaved or deinterleaved order thanks to the collision-free (de)interleaver used. Therefore each SB's SISO processor can linearly read through their corresponding extrinsic LLRs coming from the previous half- iteration. After a fixed number of iterations, a hard decision is produced as shown in Fig. 7.

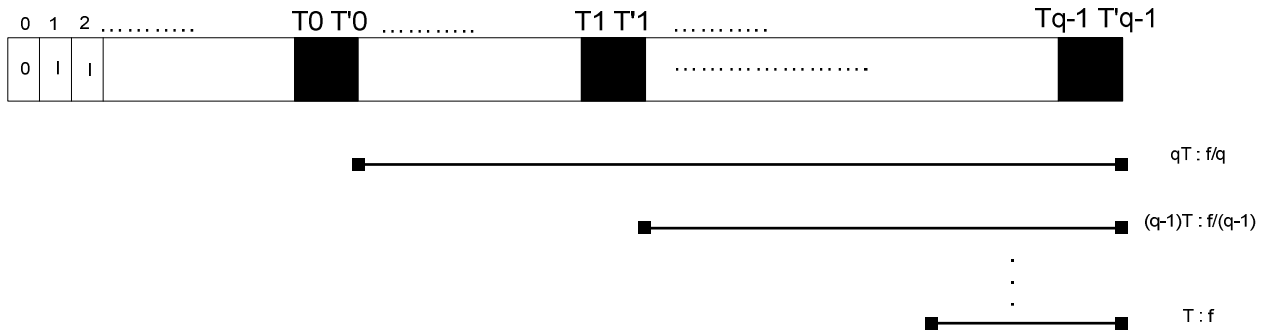

Figure 8. q-point trellis termination turbo decoding timing diagram.

Fig. 8 shows use of multi-point termination for on-the-fly decoding. This approach will eliminate the requirement of receiving the whole data frame before starting the decoding process which saves time for long frame lengths. In addition, to reduce power consumption it is possible to adaptively change the decoding frequency as shown in Fig. 8. While receiving data symbols, the extrinsic LLR information is only required for the next half-iteration. Therefore, first sub-block SB0 can be decoded with frequency *f/q* while the next sub-block can be decoded at *f/(q-1)*, and the final SB can be decoded at the highest frequency *f*.

VI. CONCLUSION

The multi-point trellis termination of turbo decoders is simulated and compared with the standard turbo coding. Simulation results show that in the low-SNR region the BER and FER performance of multi-point termination is almost the same as the standard turbo decoding. However, at high SNRs, the BER and FER performance of the multi-point terminated turbo code degrades as the number of termination points increases. This degradation can be explained by the addition of non-turbo coded extra tail bits.

The multi-point trellis termination can be utilized for high speed and low-power turbo decoding. A turbo decoder architecture based on multi-point trellis termination and collision-free interleavers is presented in the paper. Also, adapting on-the-fly decoding of turbo coded frames with multi-point trellis termination is explained which will provide further reduction in the decoding latency for long interleaver lengths.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, *Near Shannon limit error-correcting coding and decoding: Turbo-codes.* in Proc. ICC'93, 1993. **2**: p. 1064 - 1070.

[2] E. Boutillon, C. Douillard, and G. Montorsi, *Iterative Decoding of Concatenated Convolutional Codes: Implementation Issues*. Proceedings of the IEEE Custom Integrated Circuits Conference (CICC), 2007. **95**(6): p. 1201 - 1227.

[3] J. Hokfelt, O. Edfors, and T. Maseng, *A survey on trellis termination alternatives for turbo codes.* IEEE 49th Vehicular Technology Conference, 1999. **3**: p. 2225 - 2229.

[4] J. Hokfelt, O. Edfors, and T. Maseng, *On the theory and performance of trellis termination methods for turbo codes.* IEEE Journal on Selected Areas in Communications, 2001. **19**(5): p. 838 - 847.

[5] B. Mielczarek and A. Svensson, *Joint adaptive rate turbo decoding and synchronization on Rayleigh fading channels.* Vehicular Technology Conference, 2001. VTC 2001 Spring. IEEE VTS 53rd, 2001. **2**: p. 1342 - 1346.

[6] K. Wan, Q. Chen, and P. Fan. *A novel parallel turbo coding technique based on frame split and trellis terminating*. in *Proceedings of the Fourth International Conference on Parallel and Distributed Computing, Applications and Technologies, PDCAT'2003*. 2003.

[7] Z. He, P. Fortier, and S. Roy, *Highly-Parallel Decoding Architectures for Convolutional Turbo Codes.* IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2006: p. 1147 - 1151.

[8] Y. Liu, M. Fossorier, and S. Lin. *MAP algorithm for decoding linear block codes based on sectionalized trellis diagrams*. in *Proceedings of the GlobeCom'98*. 1998. Sydney, Australia.

[9] S. Yoon and Y. Bar-Ness, *A parallel MAP algorithm for low latency turbo decoding.* IEEE Communications Letters, 2002. **6**(7): p. 288 - 290.

[10] A. Nimbalker, et al., *Contention-Free Interleavers for High-Throughput Turbo Decoding.* IEEE Transactions on Communications, 2008: p. 1258 - 1267.