Increasing the speed of parallel decoding of turbo codes.

Mustafa Taskaldiran
Richard C.S. Morling
Izzet Kale

School of Electronics and Computer Science

Increasing the Speed of Parallel Decoding of Turbo Codes

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— **Turbo codes experience a significant decoding delay because of the iterative nature of the decoding algorithms, the high number of metric computations and the complexity added by the (de)interleaver. The extrinsic information is exchanged sequentially between two Soft-Input Soft-Output (SISO) decoders. Instead of this sequential process, a received frame can be divided into smaller windows to be processed in parallel. In this paper, a novel parallel processing methodology is proposed based on the previous parallel decoding techniques. A novel Contention-Free (CF) interleaver is proposed as part of the decoding architecture which allows using extrinsic Log-Likelihood Ratios (LLRs) immediately as a-priori LLRs to start the second half of the iterative turbo decoding. The simulation case studies performed in this paper show that our parallel decoding method can provide %80 time saving compared to the standard decoding and %30 time saving compared to the previous parallel decoding methods at the expense of 0.3dB Bit Error Rate (BER) performance degradation.**

I. INTRODUCTION

Because of their near-Shannon Limit performance turbo codes [1] have been incorporated into many standards such as 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].

The turbo decoding process consists of two SISO processors which repeatedly perform symbol-by-symbol Maximum a Posteriori (MAP) decoding to compute soft decisions about the received data and exchange their decisions through a number of iterations [2]. This iterative process causes a major decoding delay which is critical in real-time wireless applications. The latency drawback of turbo codes can be solved by increasing the parallelism in both decoding algorithms and decoder architectures [3]-[5].

In this paper, a novel, reduced-latency parallel decoding scheme is proposed and it is compared with the conventional turbo decoding and existing parallel decoding methods. The proposed method is based on previously developed parallel decoding techniques as given in [4]. To further reduce the (de)interleaver delay we proposed a methodology to generate a CF interleaver allowing exchange of extrinsic information immediately after (de)interleaving.

The next section briefly describes the conventional and parallel turbo decoding. CF Interleaver requirements are explained in section II. Section III presents the proposed parallel decoding method and a new Fast-CF interleaver. Simulation results, performance comparison of the proposed decoding method and existing decoding methods are presented in section IV. Finally, section V concludes the paper with a discussion about pointers to future work.

II. TURBO CODES AND PARALLEL DECODING

The classical turbo code encodes an information sequence by using two Recursive Systematic Convolutional (RSC) encoders separated by an interleaver [1]. The UMTS standard requires the constituent turbo encoders start and end at a known state (all-zero-state) which is achieved by terminating both encoders by a certain tail-bit sequence. This known-state information is used at the decoder side for state metric computations [2].

A general turbo decoder consists of two SISO processors working iteratively on the received data sequence. These two processors produce LLRs for the encoded data sequence X transmitted over a noisy channel and received as Y. Each decoder calculates an LLR for the k^th transmitted data bit d_k as

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

The LLR computations can be performed by either MAP or Soft-Output Viterbi Algorithm (SOVA). In this paper, Log-MAP decoding algorithm is used as it provides better decoding complexity versus performance trade-off.

978-1-4244-3732-0/09/$25.00 ©2009 IEEE

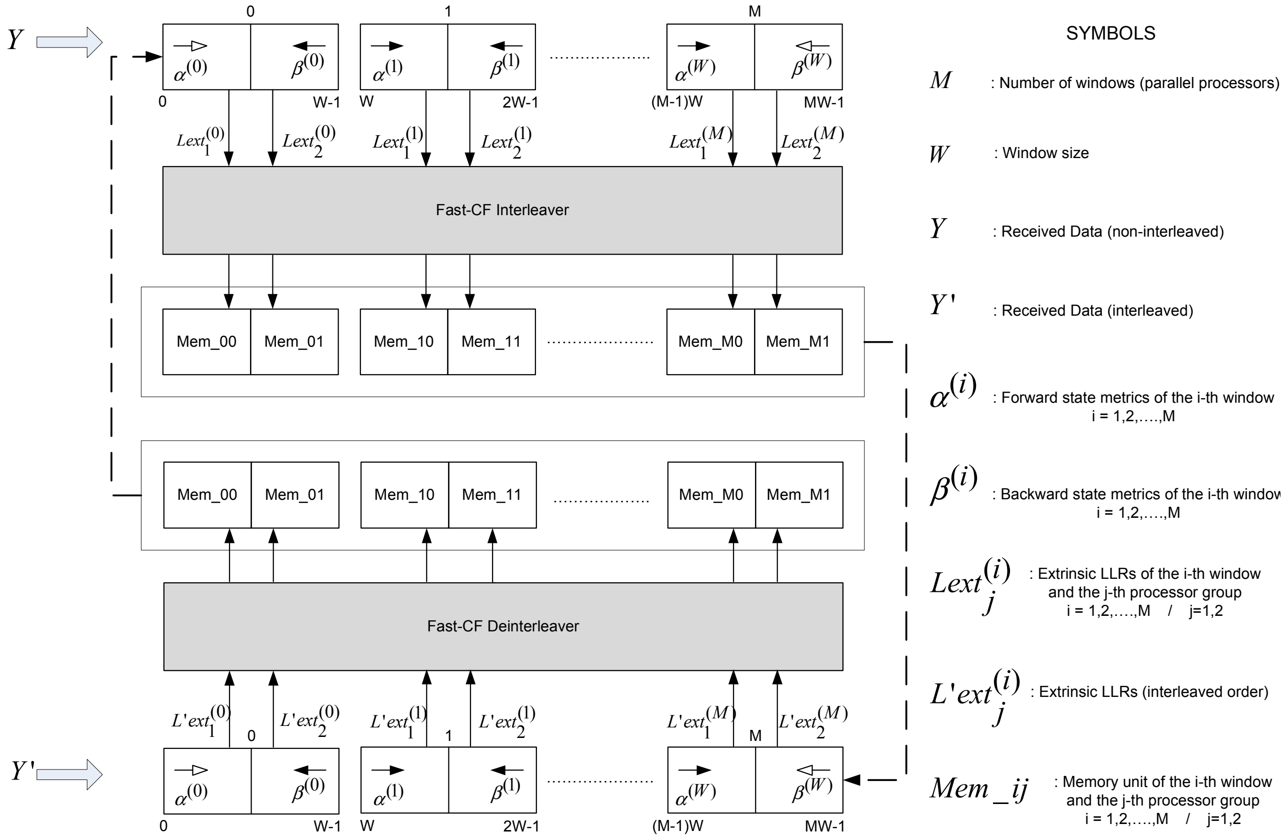

Y
Y'
0
1
M
α(0)
β(0)
α(1)
β(1)
α(W)
β(W)
0
W-1
W
2W-1
(M-1)W
MW-1
Lext(0) 1
Lext(0) 2
Lext(1) 1
Lext(1) 2
Lext(M) 1
Lext(M) 2
Fast-CF Interleaver
Mem_00
Mem_01
Mem_10
Mem_11
Mem_M0
Mem_M1
Fast-CF Deinterleaver
L'ext(0) 1
L'ext(0) 2
L'ext(1) 1
L'ext(1) 2
L'ext(M) 1
L'ext(M) 2
SYMBOLS
M : Number of windows (parallel processors)
W : Window size
Y : Received Data (non-interleaved)
Y' : Received Data (interleaved)
α(i) : Forward state metrics of the i-th window i = 1,2,….,M
β(i) : Backward state metrics of the i-th window i = 1,2,….,M
Lext(i) j : Extrinsic LLRs of the i-th window and the j-th processor group i = 1,2,….,M / j=1,2
L'ext(i) j : Extrinsic LLRs (interleaved order)
Mem _ij : Memory unit of the i-th window and the j-th processor group i = 1,2,….,M / j=1,2

Figure 1. The proposed parallel turbo decoding architecture.

A detailed description of the turbo encoding and decoding can be found in [2].

Turbo decoding latency should be reduced to meet the increasing demand for high throughputs by current wireless applications. 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 valid 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 state metrics [6]. However, this will bring extra computational load during the warm-up period and will also reduce the throughput. In [7], 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 [7]. 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].

To avoid memory collisions CF interleavers have been proposed [9]. CF interleavers should carefully be developed to prevent memory collisions while preserving the decoding performance. Our novel CF interleaver used in the proposed parallel decoding method is described in the next section. A detailed mathematical description of the CF interleaver requirements can be found in [9].

III. THE PROPOSEED PARALLEL DECODING METHOD

The previous parallel decoding methods use CF interleavers to immediately write extrinsic LLRs at (de)interleaved address locations. This requires computing all extrinsic LLRs before using them as a-priori LLRs at the next decoding stage. To eliminate this latency between two half iterations, at each clock cycle, parallel SISO processors working on the *non-interleaved data-parity* pair should compute the exact a-priori information required to start processing the *interleaved data-parity* pair. Parallel SISO processors working on the interleaved data-parity pair should do the same when it comes to computing its extrinsic LLRs. Our parallel decoding method uses this novel approach to eliminate the waiting time between half iterations.

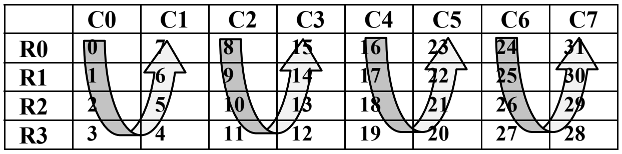
The proposed parallel turbo decoding method shown in Fig. 1 divides a received message (Y) of length N into M windows of size W as was done in previous parallel decoding methods. Forward (α) and backward (β) state metric computations start from the opposite ends of a window at the same trellis time as shown in Fig. 1. When the midpoint of the window is reached two extrinsic LLRs from each window are computed and written into the (de)interleaved memories at each clock cycle. To prevent memory collisions, each window writes into two distinct memory units depending on the (de)interleaving. Furthermore, the CF (de)interleaver is chosen in such a way that the computed extrinsic LLRs correspond to the required a-priori information of the other set of parallel processors. Therefore, the latency requirement between the non-interleaved and interleaved data processing is eliminated. This new CF interleaver will be called as *Fast-CF interleaver*.

In this paper, the window size is assumed to be a power of 2. Furthermore, the boundary conditions (state metrics) are initialised to 1/number_of_states and updated at the end of iterations as suggested in [7].

A. *Fast-CF Interleaver*

The proposed Fast-CF interleaver of length N=MW first writes the addresses from 0 to N-1 into a (W/2 x 2M) matrix. This matrix is filled as graphically shown in Fig. 2. First column is filled in downwards (0 to W/2-1) and the second column is filled in upwards (W/2 to W-1). This procedure is repeated until all addresses from 0 to N-1 are written into the matrix. At the end, W/2 row-matrices (of size 2M) are shuffled by using an appropriate interleaver. In our simulations we use the Takeshita-Costello interleaver [8] for row shuffling. Each shuffle should be independent from each other to obtain good interleaving. We achieve this by changing the parameters of the Takeshita-Costello interleaver for each row permutation. Finally, the interleaved addresses are read through the matrix starting from the first column upwards then second columns downwards, etc. As an example, the CF interleaver of size N=32, window size W=8 and number of windows M=4 is constructed as follows:

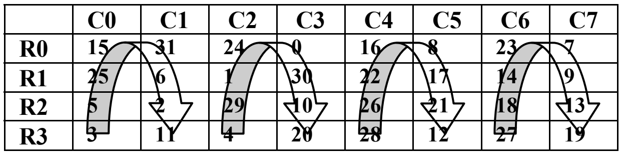
1. Fill in the 4x8 matrix as shown in Fig. 2.



	C0	C1	C2	C3	C4	C5	C6	C7
R0	0	7	8	15	16	23	24	31
R1	1	6	9	14	17	22	25	30
R2	2	5	10	13	18	21	26	29
R3	3	4	11	12	19	20	27	28

Figure 2. Fast-CF interleaver address-matrix write procedure

2. Shuffle 4 row-matrices by using the Takeshita-Costello interleaver.



	C0	C1	C2	C3	C4	C5	C6	C7
R0	15	31	24	0	16	8	23	7
R1	25	6	1	30	22	17	14	9
R2	5	2	29	10	26	21	18	13
R3	3	11	4	20	28	12	27	19

Figure 3. Fast-CF interleaver address matrix shuffle and read procedure

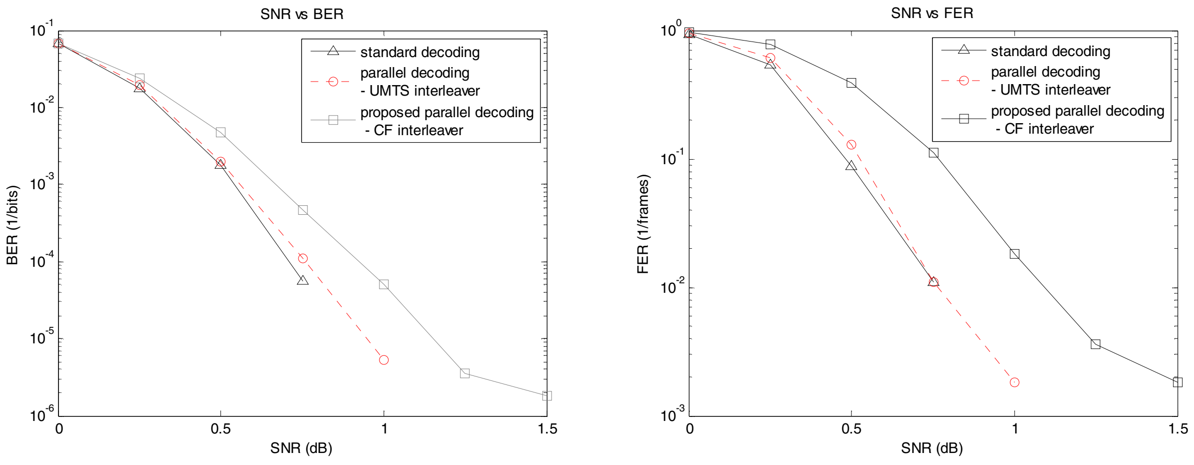
3. Read the final interleaver addresses into a row-matrix through the columns of 4x8 matrix as shown in Fig. 3: Read C0 upwards from R3 to R0, C1 downwards then R0 to R3, then C2 upwards R3 to R0,and so on. The interleaver addresses are stored in a row matrix as (3, 5, 25, 15, 31, 6, 2, 11, 4, 29, 1, 24, 0, 30, 10, 20, 28, 26, 22, 16, 8, 17, 21, 12, 27, 18, 14, 23, 7, 9, 13, 19).

IV. SIMULATION RESULTS

Our simulation framework uses the UMTS standard, rate 1/3 turbo encoder consisting of 8-state component codes with a generator polynomial of $G(D) = \frac{1 + D + D^3}{1 + D^2 + D^3}$. BPSK modulation is used to send the turbo encoded information over an Additive Gaussian White Noise (AGWN) channel. Log-MAP algorithm is used for turbo decoding. Simulations compare the BER and Frame Error Rate (FER) performance of our proposed parallel decoding method with the standard turbo decoding and parallel decoding reported in [7]. The frame length is 2048 and the number of decoding iterations is 6. For our reduced latency parallel decoder the Fast-CF interleaver explained in section III is used. For the standard decoding and the other parallel decoding method, the UMTS turbo interleaver is used. The window size for the parallel decoding methods is taken as 128-bit.

Fig. 4 shows the BER and FER performances for three decoding methods. The standard and normal parallel decoding show almost the same performance. The performance of the proposed parallel decoding method is 0.3dB worse than the standard decoding at the BER of 10^{-5}. This performance degradation is caused by the constraints used to generate the Fast-CF interleaver to allow the second constituent decoder to start decoding as soon as one extrinsic LLR from the other constituent decoder is computed.

Decoding time saving and BER/FER performances at 0.75dB are given in Table 1. The proposed parallel decoding



a)

b)

Figure 4. a) BER and b) FER performance of the standard decoding, parallel decoding and proposed parallel decoding. The block length is 2048, the number of decoding iterations is 6. For both parallel decoding methods the window size is 128-bit.

method provides approximately %80 time-saving compared to the standard decoding and %30 (≈%79.76 - %49.85) time-saving compared to the normal parallel decoding method at the expense of 0.3dB performance degradation.

TABLE I. DECODING TIME AND PERFORMANCE COMPARISON

		At 0.75 dB	
	Time saving	**BER**	**FER**
Standard decoding	-	5.59x10^{-5}	3.7x10^{-2}
Standard parallel decoding	%49.85	11.01 x10^{-5}	2.4x10^{-2}
Proposed parallel decoding	%79.76	46.43x10^{-5}	17.9x10^{-2}

V. CONCLUSIONS AND FUTURE WORK

This paper presents a novel parallel decoding method using a new Fast-CF interleaver designed to increase the parallelism in turbo decoding. This method eliminates the time delay caused by (de)interleaver and provides continuous decoding. The Fast-CF interleaver is designed by writing addresses (of a certain frame length) into a rectangular matrix column-wise (the number of columns depends on the window size), shuffling the rows by using the Takeshita-Costello interleaver, and reading out final interleaved addresses column-wise in a certain manner. In our simulations a performance degradation of 0.3dB is observed. On the other hand decoding speed is increased by almost %80 compared to the standard decoding.

The constraints imposed on the interleaver are the major cause of the performance degradation of the proposed parallel processing method. These constraints can be examined in more detail as well as improving the CF interleaver performance through better row-shuffling. There is a potentially high gain to be made if relevant procedures to avoid low-weight multiplicities of the code words (at the encoder side) outlined in [9] are deployed for our method.

VI. REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in Proceedings of 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] Y.C. Lu, T.C. Chen, and E.H. Lu, "Low-Latency Turbo Decoder Design by Concurrent Decoding of Component Codes," 3rd International Conference on Innovative Computing Information and Control,ICICIC '08. 2008:p.533.

[4] Y.C. Lu and E.H. Lu, "A Parallel Decoder Design for Low Latency Turbo Decoding," Second International Conference on Innovative Computing, Information and Control. ICICIC '07., 2007: p. 386.

[5] Y. Zhang and K.K. Parhi, "Parallel Turbo decoding," Proceedings of the 2004 International Symposium on Circuits and Systems, ISCAS '04., 2004. 2: p. 509-512.

[6] Y. Lin, S. Lin, and M. Fossorier, "MAP algorithm for decoding linear block codes based on sectionalized trellis diagrams," Proceedings of the GlobeCom'98, Sydney, Australia, Nov. 1998, pp. 562–566.

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

[8] O.Y. Takeshita, and D.J.,Jr. Costello, "New Classes of Algebraic Interleavers for Turbo-Codes," Proceedings of 1998 IEEE International Symposium on Information Theory, Boston, Aug. 1998. p. 419.

[9] A. Nimbalker, T.K. Blankenship, T.K., B. Classon, T.E. Fuja, and D.J. Costello, "Contention-Free Interleavers for High-Throughput Turbo Decoding," IEEE Transactions on Communications, 2008. P. 1258 – 1267.