On Multiple Slice Turbo Codes[(1)(2)]

David Gnaedig[*], Emmanuel Boutillon[+], Michel Jézéquel[-], Vincent C. Gaudet[°] and P. Glenn Gulak[&]

[*] TurboConcept [+] LESTER. Université de Bretagne Sud [-] ENST Bretagne
[°] Dept. of ECE, University of Alberta [&] Dept. of ECE, University of Toronto
E-mail: david.gnaedig@univ-ubs.fr, emmanuel.boutillon@univ-ubs.fr

Abstract: *The main problem concerning the hardware implementation of turbo codes is the lack of parallelism in the MAP-based decoding algorithm. This paper proposes to overcome this problem with a new family of turbo codes, named Slice Turbo Codes. This family is based on two ideas: the encoding of each dimension with P independent tail-biting codes and a constrained interleaver structure that allows parallel decoding of the P independent codewords in each dimension. The optimization of the interleaver is described. A high degree of parallelism is obtained with equivalent or better performance than the best known turbo codes. The parallel architecture allows reduced complexity turbo decoder for very high throughput applications.*

Keywords: turbo code, interleaver, parallelism, tail-biting code, slice turbo code

1. INTRODUCTION

High-throughput turbo-decoder architectures generally use N symbol cycles to perform one decoding iteration on an N-symbol frame. I iterations then require the sequential use of I decoders. This scheme is relatively inefficient in terms of hardware complexity since the extrinsic information memories are duplicated I times [1]. Some authors propose to parallelize the decoding process of the convolutional code for each turbo code dimension. For that, they arbitrarily divide the frame to be decoded into P segments of equal size, in order to handle each segment by a MAP decoder. In these architectures, side effects at the ends of the segments are treated by the use of a learning period [2], or by the method of a pointer which gives the initial states of each segment between two iterations [3]. However, these papers perform the decoding process of one dimension without tackling the problem of memory conflicts that can arise from the interleaving while decoding the second dimension.

In this paper, we propose a new turbo code family where both constituent codes are constructed with P independent Circular Recursive Systematic Convolutional codes [4] (CRSC, also called tail-biting code), called "slices". The decoding process of P slices in parallel is made possible in the two code dimensions by using an adapted interleaver structure. It can be noted that another similar parallel interleaver has been independently studied by Dobkin *et al.,* but associated with a convolutional turbo code [5]. The interleaver structure is constructed in a similar way to that presented in this paper, but the equations are not described.

The paper is divided into four sections. In section 2, the global coding process of the Multiple Slice Turbo Codes is described. In section 3, the choice of the interleaver and its influence on the performance is discussed. Finally, the complexity reduction associated with the proposed scheme is presented in section 4 and the performance are given in section 5.

2. MULTIPLE SLICE TURBO CODES

Multiple Slice Turbo Codes are constructed as follows. An information frame of N m-binary symbols is divided into P blocks (called "slices") of M symbols, where $N = M \cdot P$. The resulting turbo code is denoted (N,M,P). As with a classical convolutional turbo code, the coding process is first performed in the natural order to produce the coded symbols of the first dimension. Each slice is encoded independently with a CRSC code. The information frame is then permuted by an N symbol interleaver. The permuted frame is again divided into P slices of size M and each of them is encoded independently with a CRSC code to produce the coded symbols of the second dimension. Puncturing is applied to generate the desired code rate.

The interleaver is constructed jointly with the memory organization to allow parallel decoding of the P slices. In other words, at each symbol cycle k, the interleaver structure allows the P decoders to read and write the P necessary data symbols from the P Memory Banks MB_0, MB_1,…, MB_{P-1} without conflict. Indeed, only one reading can be made at any given time from a single port memory. In order to be able to access P data symbols in parallel, the memory has to be split into P memory banks. With the solution described in the present paper, the degree of parallelism can be chosen according to the requirements of the application for avoiding memory duplication. Note that only the functional units are duplicated.

The next section presents the interleaver construction, ensuring parallelism constraints while maintaining good performance.

[1] Patent N° 0204764 (France), April 16th 2002.

[2] This work has been partially funded by the region of Brittany (Virtual Turbo, grant PRIR n°A2C622).

3. INTERLEAVER CONSTRUCTION

The interleaver design is based on the one proposed in [6]: The interleaver structure is mapped onto a hardware architecture allowing a parallel decoding process.

3.1. Interleaver structure

Figure 1 presents the interleaver structure used to construct a slice turbo code constrained by decoding parallelism.

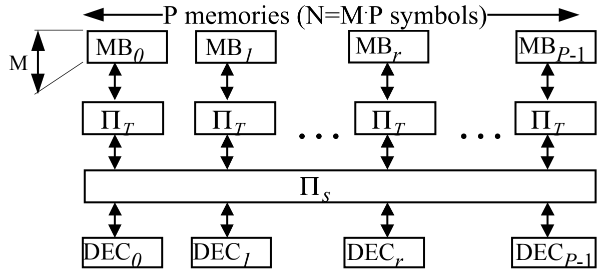

Figure 1: Interleaver structure for the (*N*,*M*,*P*) code

In the natural order, the coding process is performed on independent consecutive blocks of *M* symbols. Symbol index *j* is used in the slice *j/M* at temporal index time *j* mod *M*. Likewise, in the interleaved order, symbol index *k* is used in slice *r* = *k*/*M* at temporal index $t = k \bmod M$. We then have the relation $k = M \cdot r + t$, where $r \in \{0..P-1\}$ and $t \in \{0..M-1\}$. The permutation Π associates for each symbol at index *k* of the interleaved order, a corresponding symbol in the natural order at index $\Pi(k)$. The interleaver function can be split into two levels: a spatial permutation $\Pi_S(t,r)$ and a temporal permutation $\Pi_T(t,r)$, as defined in (1).

$$\Pi(k) = \Pi(t,r) = \Pi_S(t,r) \cdot M + \Pi_T(t,r) \qquad (1)$$

The symbol at index *k* in the interleaved order is read from the memory bank $\Pi_S(t,r)$ at address $\Pi_T(t,r)$. While decoding the first dimension of the code, the frame is processed in the natural order. The spatial and temporal permutations are then simply replaced by *Identity* functions.

The same temporal permutation is chosen for all memory banks, in order to simplify the hardware implementation. Thus, it depends only on the temporal index *t*. This solution has the advantage of requiring only one computation of the address to read *P* data symbols from *P* memory banks, that can, in fact, be merged into a single memory. The spatial permutation allows the *P* data read out to be transferred to the *P* decoders (named DEC in Figure 1). Decoder *r* receives at instant *t* the data from the memory bank $\Pi_S(t,s)$. For all fixed *t*, the function $\Pi_S(t,r)$, is then a bijection from the decoder index $r \in \{0..P-1\}$ to the memory banks $\{0..P-1\}$.

Furthermore, to maximize the shuffling between the natural and the interleaved order, we constrain the function $\Pi_T(t,r)$ such that, for each fixed *r*, *P* consecutive data symbols of slice *r* come from a distinct memory bank. Function $\Pi_S(t,r)$, for fixed *r*, is then a bijection from the temporal index $t \in \{0..P-1\}$ to the set $\{0..P-1\}$ of memory bank indices. The function is also *P*-periodic on the temporal index. This means that for $\forall t, \forall j$ satisfying $t + j \cdot P < M$, one obtains $\Pi_S(t + j \cdot P, r) = \Pi_S(t,r)$.

In the rest of the paper, *P*-periodic bijective functions are now considered for the spatial permutation. Optimization of the interleaver aims to fulfill two performance criteria: first, a good minimum distance for the asymptotic performance of the code at high signal to noise ratios (SNR); second, fast convergence, i.e. obtains most of the coding gain performance in few decoding iterations at low SNRs. As described in [7], the convergence is influenced by the correlation between the extrinsic information, caused by the presence of primary and secondary cycles. We study the influence of the spatial and temporal permutations on these two criteria.

3.2. Definitions

For an *m*-binary convolutional code, a Return To Zero (RTZ) sequence is defined as an ordered list of symbols that makes the encoder diverge from state 0 and then reconverge to state 0 [8]. The number of symbols in the list defines the length of the sequence. These RTZ sequences represent the low weight error patterns of a convolutional code. As a first approximation, we will consider that their Hamming weight grows linearly with their length. A Primary Error Pattern (PEP) is defined as a primary cycle [7] whose symbols form an RTZ sequence in both code dimensions. Note that, by construction of the constituent codes, symbols of a PEP are in the same slice in both code dimensions and they will be defined as co-cyclic symbols. Let us consider 2 symbols of a slice. Because of the tail-biting technique, two RTZ sequences are possible (see figure 2). But the one with the smallest length penalizes the minimum distance the most (smallest Hamming weight) and the good convergence ability of the code. We then define the arc length $A(t_1, t_2)$ between symbols t_1 and t_2 of a slice by the length of the smallest path $\min(|t_1 - t_2|, M - |t_1 - t_2|) + 1$ between t_1 and t_2, as presented on figure 2. The cyclic girth of two co-cyclic symbols is defined as the sum of the arc lengths in both dimensions. Let ρ be the division of the number of symbols in a slice by the number *P* of slices: $\rho = M / P$. In what follows we suppose without loss of generality that ρ is an integer.

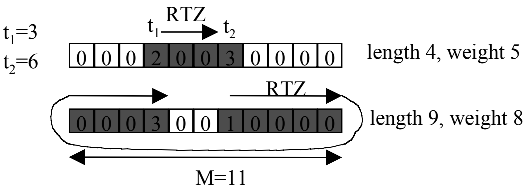

Figure 2: Example RTZ sequences for 2 given positions with the tail-biting duo-binary code defined in [9].

3.3. Temporal permutation

For the sake of simplicity, Let us define the temporal permutation $\Pi_T(t)$ as a regular one, where α and M are mutually prime. Then, $\Pi_T(t) = \alpha \cdot t \bmod M$.

Let us consider a bijective and P-periodic spatial permutation. In order to choose the parameter α of the temporal permutation, we try to maximize the cyclic girths. This maximises the weight of RTZ sequences, while also favouring the decorrelation of the extrinsic information and therefore the convergence of the turbo code [7]. As the spatial permutation is P-periodic, every couple $(t, t + j \cdot P)$,where $j = 1..\rho - 1$, forms a co-cyclic couple. Then, the cyclic girth between two co-cyclic symbols is then $A_j = A(t, t + j \cdot P) + A(\alpha \cdot t, \alpha(t + j \cdot P))$, where the first (second) term is the arc length in the interleaved (natural, respectively) order. We define $A = \{A_j, j = 1..\rho - 1\}$ as the set of cyclic girths of all co-cyclic symbols of a slice. The optimal value of α is the one that maximizes the minimal cyclic girth. Since $\alpha < M$, an exhaustive search can be performed.

The influence of the spatial permutation on more complex error patterns is now analysed.

3.4. Spatial permutation

A Secondary Error Pattern (SEP) is a secondary cycle whose symbols give two distinct RTZ sequences in both natural and interleaved order as shown on figure 3.b.

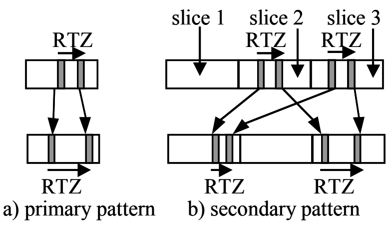

Figure 3: Primary and secondary error patterns

We have built a tool that, for a given permutation, determines all SEPs and their Hamming weight. This tool exhibits SEPs with low Hamming weight with the regular interleaver $\Pi_T(t, r) = rot_P(t \bmod P)$, where $rot_P(t)$ is the rotation of amplitude $\{0..P-1\}$, which at the index slice r associates the slice of index $(r + t) \bmod P$. In order to reduce the number of these error patterns, we introduce irregularity in the spatial permutation, with a rotation whose amplitude does not depend linearly on the temporal index. To do so, the spatial rotation is generated by a function $S(t)$, P-periodic and bijective from $\{0..P-1\}$ to $\{0..P-1\}$, and is given by

$$\Pi_T(t, r) = rot_P(S(t)) \qquad (2)$$

Using (2) with an appropriate irregular function S reduces the number of short secondary cycle and low weight SEPs and therefore improves the convergence of the code.

3.5. Optimization of the minimum distance

With increasing frame size, the study of the PEPs and SEPs, in order to choose the optimal parameters α and S is not sufficient to improve the minimum distance of the code. Indeed, other error patterns appear, penalizing the minimum distance. In practice, the analysis and the exhaustive counting of these new patterns are too complex to be performed. Thus, as in the DVB-RCS standard [9], another degree of freedom in introduced by adding four coefficients, $\beta(\mathrm{i})_{i=0..3}$ multiples of 4, in the temporal permutation:

$$\Pi_T(t) = \alpha \cdot t + \beta(t \bmod 4) \bmod M \qquad (3)$$

Even with irregularity in the temporal permutation, the irregularity in the spatial permutation is still compulsory because all symbols distant of 4 are not affected by the irregularity introduced by (3). The choice of the β parameters, which maximize the minimum distance, is determined by the error impulse method proposed by Berrou *et al.* [10] for the evaluation of the minimum distance.

4. HARDWARE IMPLEMENTATION

For a high-throughput turbo decoder, a sequential architecture, having no parallelism, requires for each iteration a duplication of the memories and of the functional units. Conversely, in a parallel architecture, only the functional units have to be duplicated. In this latter case, the same hardware (memory and SISOs) are reused twice for each iteration in order to perform decoding in both code dimensions. The memories can account for up to 90% of the total area of a sequential decoder using 10 iterations. Then with only one memory for extrinsic information and two for channel values, the total area reduction for the parallel decoder can be up to 75% for a 150Mbits/s turbo decoder, compared to a classical sequential solution. Moreover interleaving equations are simple and can be implemented in hardware with low complexity.

5. RESULTS

Table 1 presents the minimum distance optimization by applying the different methods developed in section 3. A duo-binary code is constructed with the following parameters (2048,256,8). An intra-symbol permutation [9] is also applied to increase minimum distance. When optimizing the parameter α, the minimum distance of 14 comes from SEPs and not from PEPs for which the minimal weight is above 30. By optimizing the spatial permutation through S, low weight PEPs and SEPs are eliminated and the minimum distance increases to 18. When introducing irregularity in the temporal permutation the minimum distance is raised to 21.

α	S	β	d_{min}
23 or 27	$S = Id$	{0,0,0,0}	14
27	S={0,3,2,7,4,6,5,1}	{0,0,0,0}	18
27	S={0,3,2,7,4,6,5,1}	{0,4,36,48}	21

Table 1: Minimum distance optimization for (2048,256,8) duo-binary codes

Figure 4 compares codes (2048,256,8) of Table 1 with a (2048,2048,1) code for a rate of 1/2 and 8 decoding iterations of the Log-MAP algorithm [11]. The code with one slice is constructed using the same equation (3) as the DVB-RCS code with parameters α = 45 and β(i)={0,20,12,20} and has a minimum distance of 19.

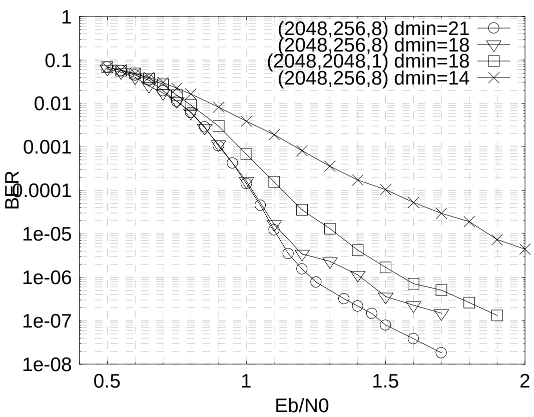

Figure 4: Performance of the (2048,256,8) duo-binary codes and (2048,2048,1) code for 8 iterations

Simulation results are in accordance with the minimum distances of the codes. The (2048,256,8) code performs 0.3 dB better at BER of 10^{-6} than the DVB-RCS. The codes proposed in this standard are between 48 and 1728 bits long and the interleaver has not been designed for longer frames. For shorter frames, Slice Turbo Codes have the same performance than the DVB-RCS codes, in terms of convergence and minimal distance.

6. CONCLUSION

A new family of convolutional turbo codes is proposed. A study of an interleaver construction has been made, ensuring the decoding parallelism, simple hardware implementation and good performance. The interleaver is split into two levels: first, the spatial permutation handles the parallelism and its irregularity improves the convergence of the code and the minimum distance of the code; second, the temporal permutation further optimizes the minimum distance. The proposed scheme allows a hardware complexity reduction of 75 % for a 150Mbits/s turbo decoder. The performance simulations show that the parallelism constraint in the interleaver construction introduces no degradation in performance, and can even improve it.

REFERENCES

[1] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes", *Proc. ICC'93*, Geneva, Switzerland, pp. 1064-1070, May 1993.

[2] Z. Wang, Z. Chi and K.K. Parhi, "Area-Efficient High Speed Decoding Schemes for Turbo/MAP Decoders", *IEEE Trans. on VLSI Systems*, 10(12), Dec. 2002.

[3] A. Worm, H.Lamm and N Wehn, "VLSI architectures for high speed MAP decoders", in *Proc 14th Int Conf VLSI Design*, pp 446-453, 2001.

[4] C. Berrou, C. Douillard and M. Jézéquel, " Multiple parallel concatenation of circular recursive systematic codes ", *Annales des Télécommunications*, tome 54, n°3-4, pp 166-172, 1999.

[5] R. Dobkin, M Peleg and R. Ginosar, "Parallel VLSI Architecture for MAP Turbo Decoder", *13th IEEE Int. Symp. on Personal, Indoor and Mobile Radio Communications* (PIMRC 2002), Lisboa, Portugal, Sep. 2002.

[6] E. Boutillon, J. Castura and F.R. Kschischang, "Decoder-first code design", in *Proceedings of the 2nd International Symposium on Turbo Codes and Related Topics*, pp 459-462, Sept 2000.

[7] J. Hokfelt, O.Edfors and T. Maseng, "Interleaver Design fo Turbo Codes Based on the Performance of Iterative Decoding", *ICC '99. IEEE International Conference on* , Vol 1 , 1999.

[8] R. Podemski, W. Holubowicz, C. Berrou and G. Battail, "Hamming distance spectra of turbo-codes", *Annales des Télécommunications*, tome 50, n° 9-10, pp. 790-797, Sept-Oct. 1995.

[9] DVB-RCS Standard, "Interaction channel for satellite distribution systems", *ETSI EN 301 790*, V1.2.2, pp. 21-24, Dec 2000.

[10] C.Berrou, S.Vaton, M.Jezequel and C.Douillard, "Computing the minimum distance of linear codes by the error impulse method", *GLOBECOM 2002*, Taipei, Taiwan, Nov 2002.

[11] P. Robertson, E. Villebrun and P. Hoeher, "A Comparison of Optimal and Sub-optimal Decoding Algorithm in the Log Domain", *Proc ICC*, Seattle, WA, pp. 1009-1013., June 1995