Designing a Sensible Block Semi-Random Interleaver for Turbo Codes

Md. Nahidul Islam
Assistant Programmer
ICB Capital Management Limited

Md. Rakibul Islam
Assistant Programmer
ICB Capital Management Limited

Mahmudul Hasan
Assistant Professor
Dept. of CSE
Comilla University, Bangladesh

Ohidujjaman
Lecturer
Dept. of CSE, DIU.

ABSTRACT
It is highly known that an interleaver (a device that scrambles the order of a sequence of numbers) is a key component of a turbo encoder to guarantee excellent bit error rate and frame error rate performances. Turbo codes were initially proposed using a randomly constructed interleaver. Turbo codes are a rank of high-performance forward error correction (FEC) codes, which were the initial practical codes to closely approach the channel capability. We introduce here a method for generating a sequence of semi-random interleavers, projected to be optimally stored and employed in a turbo coding system that requires litheness of the input block (i.e., interleaver) size. By the arrangement of construction and random search based on a careful analysis of the low weight words and the distance properties of the component codes, it is possible to find interleavers for turbo coding with a high minimum distance. We have designed a block semi-random interleaver with permutations of each row, and found a combination of permutations where a tight upper bound to the minimum distance of the complete turbo scheme is 108. By using our designed technique it is easier to include restrictions which make the interleaver correctly-terminating or odd-even. While the block semi-random interleavers serves well for specifying interleaver spread, we think our method will achieve better performance in a more sophisticated designed criteria.

Keywords
Interleavers, Encoder, Iterative decoding, Bit error rate, Turbo Code.

1. INTRODUCTION
Interleaving is a key component of many digital communication systems involving forward error correction (FEC) coding. Interleaving the encoded symbols provides a form of time diversity to guard against localized corruption or bursts of errors. In the past the interleaving strategy was usually only weakly linked to the selected FEC scheme. Exceptions are concatenated FEC schemes such as concatenated convolutional and Reed-Solomon codes. In this case, the interleaving parameters are usually carefully selected to match the error correcting capabilities of the codes involved. Recently, interleavers have become an even more integral part of the code design itself. Such is the case for Turbo and Turbo-like codes. The problem of finding good interleavers for such codes is an on-going area of research. The use of pseudo random interleavers[1] in the turbo coding scheme has been a major obstacle in the analysis of distance properties, except for probabilistic analysis on the ensemble of all interleavers. Also, there are obvious reasons to believe that there exist better interleavers than the pseudo random ones. The advent of turbo codes [2] [3][4] motivated an extensive research aimed to explain a further improves in their extraordinary error performance. A codeword in turbo codes is produced by concatenation of the outputs of two convolutional encoders and the decoding is performed by an iterative procedure. Pure construction[5][6] based on the minimum weight words and distance properties of the component code do not seem to be a viable approach. Neither does a search among the pseudo random interleavers, due to extreme consummation of computer power since all the positions in the interleaver appear differently.

We have therefore used a combination of these two approaches. That is an interleaver construction which enables a random search and still has almost the same advantages of a random distribution as the pseudo random interleavers. We have been inspired by the interleaver construction in [7], and have constructed a block semi-random interleaver with permutations of each row. For these interleavers it is possible to search for a large number of the most critical patterns with regard to a specific component code. With this approach we have found an interleaver where we have a tight upper bound to the minimum distance of the complete turbo coding scheme of 108. This is supported by simulations showing a remarkable better performance at good signal-to-noise ratios.

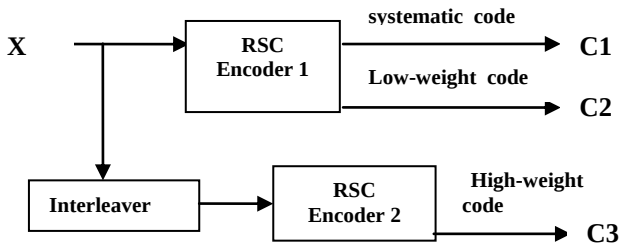


Fig 1: The interleaver increases the code weight for RSC Encoder 2 as compared to RSC Encoder 1.

2. INTERLEAVER RESPONSIBILITIES
Interleaving is a process of rearranging the ordering of a data sequence in a one to one deterministic format. The inverse of this process is called deinterleaving which restores the received sequence to its original order. Interleaving is a practical technique to enhance the error correcting capability of coding. In turbo coding, interleaving is used before the information data is encoded by the second component encoder. The block interleaver is the most commonly used interleaver in communication system. The semi-random interleaver[8] [9] is a compromise between a random interleaver[10] and a "designed" interleaver such as the block and circular-shifting interleavers[11]. The permutation algorithm for the semi-random interleaver is described below.

Here we consider a permutation algorithm:

Step1: Select a random index i, belongs to [0,N-1]

Step 2: Select a positive integer S<(N/2)^-1/2

Step 3: Compare i to previous S integers. For each of the S integers, compare i to see if it lies within +/-S. If i does lie within the range, then go back to Step 1.Otherwise, keep i.

Step 4: Go back to Step 1 until all N positions have been filled.

3. COMPONENT CODES PROPERTIES

For component code we will use (1, D^3 +D^2 +1/D^3 +D+1,D^3+D^2+D+1/D^3+D+1) with rate 1/3 and 8 states. We use the same code for both encoders but for the second one the information sequence is not transmitted. This gives an overall rate of 1/5.

The minimum weight word with information weight 2 has length 7, i.e. 10000001. In any case a sequence of 7 0's will bring the encoder through a sequence including all states, and the finale state will be identical to the initial one. Each of these cycles has output weight 8. Thus the length of the codewords with information weight 2 is 7×i+1, i=1,2,... The weight is 2+12+(i-1)×8. There are 6 classes of codewords with information weight 3. The minimum weight words in these classes are 110001, 1011, 1001000001, 1000101, 100001001, 100000100001. Again sequences of 7 0's can be inserted.

For codewords of weight 4 there are 36 classes of true weight 4 words plus the combination of two words with weight 2.

4. CONSTRUCTION OF THE INTERLEAVER

The block semi-random interleaver [12] is based on an iterative construction. We will construct the interleavers based on block and semi-random interleavers[13][14], i.e. write by row and read by column. But before reading we will make a permutation of the symbols in the rows. These permutations may be different for all the rows or identical for some of the rows. The permutations we will use are of the form i ->a×i mod rowsize. Where ‘a’ is some (prime) number. With these interleavers it is possible to search for a large number of the most critical patterns for the specific component code, and find a combination of ‘a’ values where the worst combinations are avoided.

We will divide the possible ”interleaver defects” in four different classes. These cases relate both to the effect of the “defects” and to the amount of time necessary for the search. The classes are: Low weight words appearing at any position in the interleaver, low weight words appearing at any position in a row, low weight words appearing at specific positions in a row and finally low weight words due to truncation.

For the first class the problem is rectangular patterns where the errors appear in rows with the same permutation (that may be the same row) as illustrated in Figure 2a and 2b. If we chose 11 permutations, the minimum weight is 132 and 120 respectively for the two cases. For the second class the main problem is two or three bit patterns (of different length) appearing in rows with different permutations but permuted to positions below each other, as illustrated in Figure 2a, 2b and 2c. This must be checked for all the ‘a’ values. Further the appearance of two or three bits in one row, as illustrated in Figure 2d and 2e, may appear as a codeword for the second code.

The third class includes bursts of errors distributed over two consecutive rows as illustrated in Figure 2f. Again the possibilities of low weight patterns must be checked for all a’s.

The fourth class of “interleaver defects” concern special problems due to truncation. We expect this to be a minor problem.

5. EXPERIMENTAL RESULTS

Here can consider multiple conditions for this block semi-random interleaver.

We have constructed an 11×907 (block size 9977) interleaver with 11 permutations. Further we write the rows in steps of 5, i.e. row 0, 5, 10, 4, 9 ... .With this interleaver we have searched for a number of critical patterns, that is:

Single 2-bit patterns (2d)

Two horizontal 2-bit patterns (2a)

Two vertical 2-bit patterns (2a)

2-bit patterns divided on two rows (2f)

Single 3-bit patterns

Two horizontal 3-bit patterns (2c)

Two vertical 3-bit patterns (2b)

3-bit patterns divided on to rows

Two horizontal true 4-bit patterns

Two vertical true 4-bit patterns

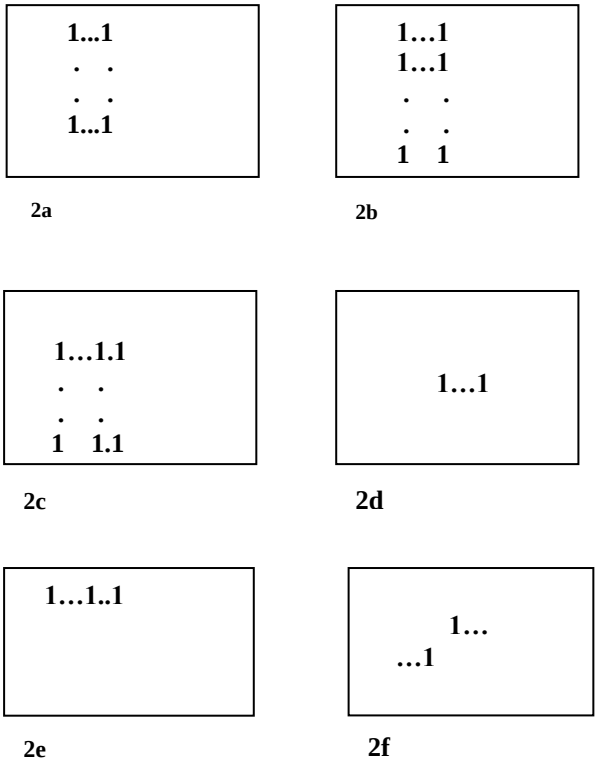
Some patterns related to truncation

With ‘a’ values 7, 11, 13, 17, 37, 43, 47, 53, 59, 61, 67. The minimum weight word found for these cases have weight 108. This word appears as two 3-bit patterns in rows with different permutations, as shown in Figure 2c. This pattern (i.e. 907 words with weight 108) corresponds to an expected bit error rate of 1x10^ -12 at E /N = 2 dB. As a comparison we must expect two words of weight 32 with a semi-random interleaver of N size. These words occur when a two bit word is interleaved to a similar pattern.

We have tried to verify this upper bound on the minimum distance by simulation. In Figure 3 these results are compared with simulation results with a semi-random interleaver of approximately the same size (10,000). In both cases we show the results after 9 iterations and after 32 iterations. While the system with the pseudo random interleaver suffers from an error floor at a BER about (1.0×10)^-12, no error floor is seen for the system with the row permuted block semi-random interleaver.

This means that the error floor at least has been lowered several decades. At Eb /No = 2 dB we have simulated 1e8 frames with errors 100 after 32 iterations. What might look as the beginning of an error floor for the curve representing 9 iterations, is more probable just due to uncertainty on the simulation results since only one erroneous frame was observed at 2 dB.

For the low signal-to-noise ratios the performance is not dominated by the minimum weight words, but by lack of convergence in the decoding process. There seem to be a small advantage (< 1 dB) for the pseudo random interleaver after a large number of iterations in this region.



2a 2b

2c 2d

2e 2f

Fig 2: Different Critical patterns

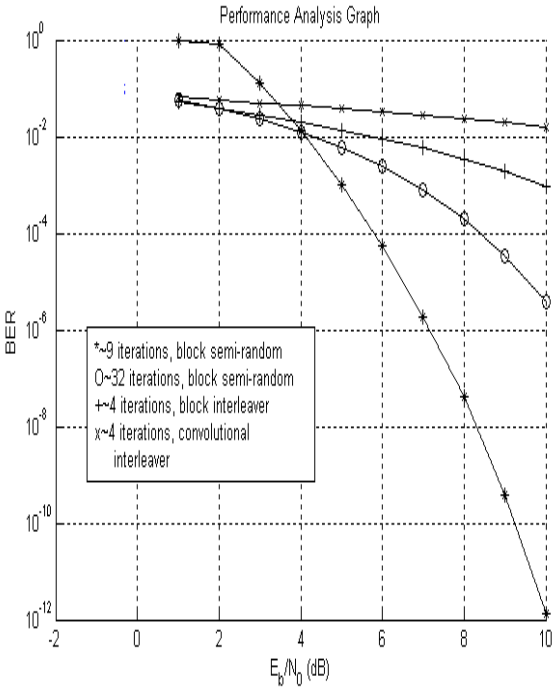


Fig 3: Simulation result with convolutional and block interleaver with Block semi-random interleaver.

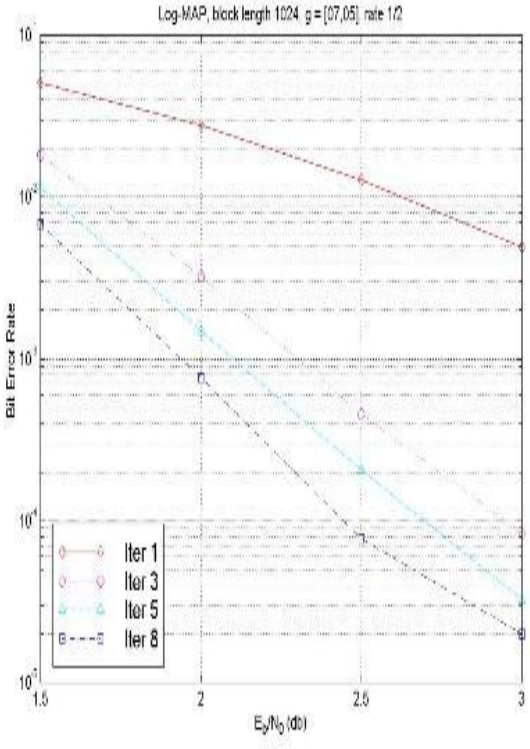


Fig 4: Log-MAP, Bit-Length 1024

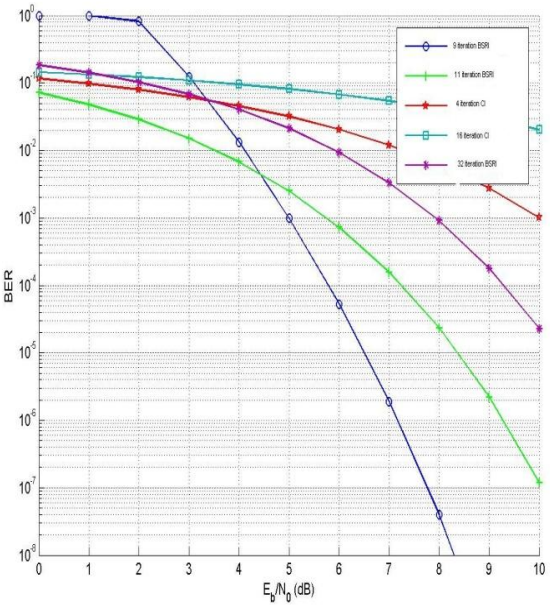


Fig 5: Another performance analysis with convolutional interleaver(CI) and block semi-random interleaver (BSRI).

From these above figures we clearly see that our designed block semi-random interleaves provide more sophisticated result compared with other traditional interleavers.

6. CONCLUSION

The technique proposed in this paper to design block semi-random interleaver allows the designer to satisfy the requirement of prunability without sacrificing performance.

To substantiate this statement, results were given in terms of spreading parameters, obtainable free distance of the actual codes, and simulated frame error probabilities. We have proposed here a new interleaver design technique and show that it performs better than other random or partly semi-randomized designs.

With this new interleaver construction we have been able to remove the error floor problem. We do not claim that this is the optimal interleaver/coding scheme in anysense, but just that it represents the first step towards a structured interleaver design and weight analysis. The number of critical patterns in the search list can of cause be augmented or adjusted. In future we will overcome our limitations regarding the simulation and other threshold calculation techniques.

7. REFERENCES

[1] Mahmudul Hasan, Iqbal Izaz Khan "Block Semi Random Interleaver Design for Turbo Codes", in ICECC, 27-29 June-2007, Rajshahi University,Rajshahi.

[2] J. D. Andersen,"Turbo Codes Extended with Outer BCH code", Electronics Letters, Vol. 32, No. 22, October 1996.

[3] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in Proc. IEEE Int. Commun. Conference (ICC), 1993, pp. 1064–1070.

[4] S. Dolinar and D. Divsalar, Weight distribution for turbo codes using random and nonrandom permutations, in TDA Progress Rep. 42-122, Aug. 1995.

[5] J. Hokfelt, O. Edfors, and T. Maseng, "Assessing interleaver suitability for turbo codes," in Nordic Radio Symp., Saltsjobaden, Sweden, Oct. 1998.

[6] S. Crozier, "New high-spread high-distance interleavers for turbo-codes," in Proc. 20th Biennial Symp. on Communications. Kingston, ON, Canada, May 2000, pp. 3–7.

[7] S. Benedetto and G. Montorsi, "Design of parallel concatenated convolutional codes," IEEE Trans. Commun., vol. 44, no. 5, May 1996.

[8] L.Dinoi and S. Benedetto, "Design of prunable S-random interleavers," in Proc. Int. Symp. on Turbo Codes, Brest, France, 2003, pp. 279–282.

[9] M. Ferrari, F. Scalise, and S. Bellini, "Prunable S-random interleavers," in Proc. IEEE Int. Commun. Conf. (ICC), vol. 3, 2002, pp. 1711–1715.

[10] Petar Popovski, Ljupco Kocarev and Aleksandar Risteski, "Design of Flexible-Length S-Random Interleaver for Turbo Codes" in Proc. IEEE communication letters vol. 8, no. 7, July 2004.

[11] Fu-hua Huang, "Turbo Code" in Blacksburg, Varginia, May 29, 1997.

[12] Berrou, Claude; Glavieux, Alain; Thitimajshima, Punya, *Near Shannon Limit Error - Correcting*, retrieved 11 February 2010

[13] C. Berrou, A. Glavieux, and P. Thitimajshaima, "Near Shannon limit error-correcting coding and decoding: turbo-codes," in *Proc. IEEE Int. Conf. Commun.*, May 1993, pp. 1064–1070.

[14] C. Berrou, A. Glavieux ,"Near optimum error correcting coding and decoding: Turbo codes," *IEEE Trans. Communications*, vol. 44,No 10 pp. 1261–1271,Oct. 1996.

[15] Robert Garello,Paola Pierleoni and Sergio Benedetto," Computing the free distance of turbo codes and serially concatenated codes with interleavers : Algorithms and Applications," *IEEE Journal on selected areas in communications*,Vol.19 No.5,May 2001