International Journal of Multimedia and Ubiquitous Engineering
Vol.10, No.8 (2015), pp.73-82
http://dx.doi.org/10.14257/ijmue.2015.10.8.08

Shortening the Turbo Codes Based on Unequal Error Protection

Shao Xia[1] and Zhang Weidang[2]

1. First author, Department of Information Engineering, North China University of Water Resources and Electric Power, Zhengzhou 450011, China. shaoxia@ncwu.edu.cn
2. Corresponding author, School of Information Engineering, Zhengzhou University, Zhengzhou, 450001, China, zhangweidang@zzu.edu.cn

Abstract

Shortened turbo product codes (TPCs) have already been adopted in many standards, especially in multimedia field. Generally, the realizations of shortening turbo codes are based on designing specific interleavers with variable interleaving span; or using shortened–extended Hamming codes as the component to obtain different encoding block sizes. In this paper, we propose a very simple method to realize the shortening. This method is based on a parallel concatenated convolutional code (PCCC) and their inherent characteristic of unequal error protection. Sample simulation results are presented confirming the efficiency of the proposed scheme.

Keywords: *turbo codes, variable length turbo codes, multimedia, unequal error protection (UEP)*

1. Introduction

Variable-length block coding is of significant interest in practice to support flexible and high code rate, especially in the multimedia communication. Since turbo codes are potential candidates for many applications due to their excellent performance near the capacity limit [1-2], shortened turbo codes have already been adopted in many standards. Since the encoding and decoding delay associated with turbo codes are essentially dominated by the block length (or the data frame length), it is of considerable practical interest to be able to construct variable-length turbo codes. There are some existed papers considered in this aspect. The paper [3] was focused on design of variable-length interleavers with application to turbo codes, starting from any generic interleaver of maximal length. Based on the approach presented in [4], it presented a pruning method suitable for practical implementation and applicable to any interleaver and showed the average optimality of the pruning strategy. The letter [5] given an in-depth analysis of a low-complexity method recently proposed by Guivarch *et al*. [6-7], where the redundancy left by a Huffman encoder is used at a bit level in the channel decoder to improve its performance. Several simulation results are presented, showing for two first-order Markov sources of different sizes that using a priori knowledge of the source statistics yields a significant improvement, either with a Viterbi channel decoder [8] or with a turbo decoder. Paper [9] studied the shortening strategy for turbo product codes (TPCs) [10-11]. In this paper, the shortened TPCs from two aspects, namely encoding design and decoding algorithm are studied. To obtain different encoding block sizes, shortened–extended Hamming codes [12] are used as the component codes of product codes in the IEEE 802.16 Standard. To design a good structure for the shortened TPC, it computed the undetected error probability of its corresponding component codes. The component codes are shortened–extended Hamming codes, and their optimal generator polynomials are selected in terms of their undetected error probability. For the decoding algorithm, it presented an efficient Chase decoding algorithm for shortened TPCs in flat fading

ISSN: 1975-0080 IJMUE

channels based on [13-14]. In the proposed scheme, the reliability factor used in Pyndiah's scheme is not needed; thus, the decoding complexity is greatly reduced by avoiding the normalization operation of the whole code word at each iteration. Simulation results are also presented to verify the performance of the proposed algorithm. In paper [15], the authors focused on pairs of source/VLC of low redundancy, *i.e*., when there is a good match between the source statistics and the length distribution of the VLC. It is a case not considered extensively in the literature so far and the classical concatenation of a VLC and a convolutional code is not satisfying. Through EXIT chart and interleaving gain analysis, it showed that the introduction of a repetition code between the VLC and the convolutional code considerably improves global performance. In particular, excellent symbol error rates are obtained with reversible VLCs which are used in recent source codes. In paper [16], the success of joint source-channel turbo techniques has been attested. It capitalized on previously developed performance upper bounds, and investigates whether the VLC can contribute to the interleaving gain of concatenated codes just as a convolutional code with noncatastrophic encoder would. To this end, the important concept of bounded VLC spectrum is introduced and is proved to be a sufficient condition for the VLC to contribute indeed to the interleaving gain. In paper [17], a parallel concatenated coding scheme for the VLC combined with a turbo code is presented. By merging a symbol-level VLC trellis with a convolutional trellis, a symbol-level joint trellis with compound states is constructed. Also, a solution of the symbol-by-symbol a posteriori probability (APP) decoding algorithm based on this joint trellis is derived, which leads to an iterative JSCD approach in the similar way to the classical turbo decoder. The simulation results show that our joint source-channel en/decoding system achieves some gains at the cost of increasing decoding complexity, when compared to the joint iterative decoding based on the bit-level super trellis for the separate coding system.

In this paper, we only focus on constructing variable-length turbo codes, such as the references [3] and [9] did. We try to present a very simple and practical method to achieve this destination. The pruning strategy is based on a generic turbo code; we call it mother turbo code. And the encoding and decoding algorithms will not be changed almost. The rest of the paper is organized as follows. In Section II, we briefly introduce the inherent characteristic of unequal error protection of turbo codes which takes a very important role in the proposed strategy. In Section III, we present the proposed pruning strategy. Some simulation results are also showed in this Section very the efficiency of the strategy. In Section IV, more simulation results are given and the results are compared with those in references. In Section V, we draw some conclusions.

2. Inherent Feature of Unequal Error Protection of Turbo Code

Generally, the Bit Error Rate (BER) at various positions within the data block is not uniform. This phenomenon has been discussed deeply in reference [18], so we do not mention it again. Here we only give an example that will be used in the next Section. In Figure 1 we show the bit error rate at various positions within a 64-bit data block for a turbo code with generator polynomials (1,1101/1011) and an 64 random interleaver. There was as no puncturing. The code rate is 1/3. To obtain the BER results, we simulated the transmission of 1010 data bits (cut up into 64-bit blocks) over an Additive White Gaussian Noise (AWGN) channel with a Signal-to- Noise Ratio (SNR) of 4 dB. Turbo decoding was carried out using 5 iterations of the Buhl–Cocke–Jelinek–Raviv (BCJR) algorithm.

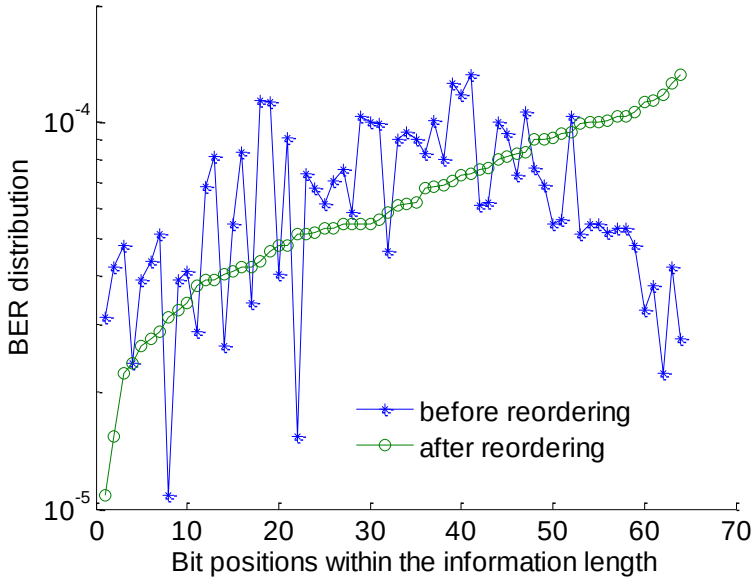


Figure 1. BER Distributions within a Data Block of 64 bits both in the Original Order and in Reordering Order at SNR=4dB

There are two curves in Figure 1. One is the original BER distribution and another is the ascending order of the BER. Figure 1 shows that BER is actually not uniform and varies by an order of magnitude over the 64 bit positions.

If the original index sequence for the bit position is (0,1,2,…,k-1), after reordering, the sequence of the index becomes (i_0,i_1,i_2,…,i_{k-1}). This indicates that the i_0th bit in the original sequence has the lowest error rate, so it is placed in the first position in the reordered sequence; i_1th bit has the second lowest error rate, it was placed in the second position in the reordered sequence; so on. In the following Section we will use this inherent UEP property for shortening the turbo codes.

3. Designing Procedures of the Shortening Strategy

In this Section, we will directly present the designing procedures of the shortening strategy firstly and then give simulation results to show the efficiency of the scheme proposed.

Assume BPSK is used and the code is binary and decoding algorithm is iterative.

a) Select a systematic (n, k) mother code based on which the shortening will be processed. Some more detailed examples for selection will be given in next Section.

b) Obtain a BER distribution for the information bits at a certain SNR by simulation based on the mother code. This distribution will be used for reordering the shortened input bits.

c) Shorten the information sequence by setting the last s bits to zero. Let **I**=(a_0,a_1,a_2,…,a_{k-1}) be the information sequence, after shortening, it becomes **I`**=(a_0,a_1,…,a_{k-s-1}, 0,…,0). The shortened information sequence is **I``**= (a_0,a_1,…,a_{k-s-1}).

d) Reorder the k-s bits so that they are located at the positions where the bit error rates are lowest. Let **J** = (b_0,b_1,…,b_{k-1}) be the information sequence after reordering. b_0 is the position index in the original sequence where the bit error rate is lowest; b_1 is the position index in the original sequence where the bit error rate is second lowest; and so on. The mapping rule is

$$\begin{cases} b_{\pi(i)} = a_i & 0 \le i \le k-1-s \\ b_{\pi(i)} = 0 & k-s \le i \le k-1 \end{cases} \quad (1)$$

e) Encode the reordered sequence **J** to get the n-k parity check bits.

f) Transmit the shortened input sequence **I``** (not the reordered input bits) and the parity check sequence.

At the receive part, after demodulation, let **J``**= ($d_0$,$d_1$,…,$d_{k-1-s}$) be the sequence related to **I``** retrieved from the received sequence. Add s 0s to the **J``**, **J``** becomes **J`**= ($d_0$,$d_1$,…,$d_{k-s-1}$,0,…,0). Then do step d) to reorder **J`**. If the modulation form is 0→-1 and 1→+1 and iterative decoding algorithm is adopted, the s 0s are known to the decoder. So they should be replaced by a large minus value (for example, a value of -1000 is used in the following examples) before decoding. After decoding, de-reordering the output sequence, we get the decoded information bits in the original order.

To see the improvements after reordering, we now give the simulation results by selecting the code mentioned above as the mother code.

Figure2 shows the average BER before and after reordering for several shortened lengths which are 8, 16, 32 and 48 separately. For every length, systematic bits in the original order and reordering order are simulated separately. The improvements are very significant. For example, the improvements are nearly 1 order of the amplitude for group 2 and group 3 after SNR>=4 dB.

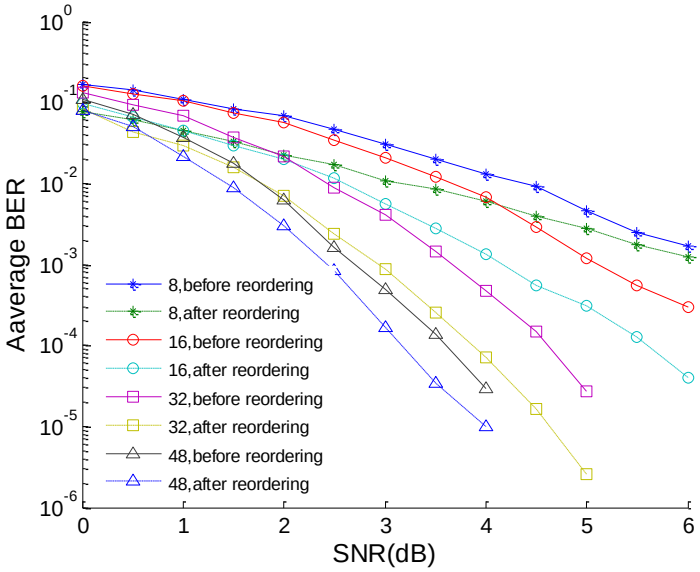


Figure 2. Average BER Before and After Reordering for Several Shortened Lengths that are 8, 16, 32 and 48 Separately

Table 1 shows the minimum weights for different shortened codes before reordering and after reordering. The minimum weight of the mother code is 10. From the Table we can see that the minimum weights are really increased after reordering.

Table 1. Minimum Weight for Different Shortened Codes Before Reordering and After Reordering

Shortened lengths	8	16	32	48
Before reordering	15	15	14	13
After reordering	19	18	16	15

In Figure 2, the reordering in based on the BER distribution obtained at SNR=4dB as in Figure 1. Which SNR should we choose so that the reordering would produce the highest improvements? We have compared the improvements after reordering the bits based on four different BER distributions produced at four different SNRs for the code

used in Figure 2. The four SNRs are -2dB, 0dB 2dB and 4 dB separately. The results are showed in Figure3. There are four groups of the curves in Figure 3. From Figure 3 we can see that at low and moderate SNRs, the difference among the curves in each group are not visible no matter what lengths are. At high SNR, the BERs according to the BER distributions produced at high SNR are lowest than others. Therefore, the BER distribution produced at high SNR is preferred. But as the code length becomes large, getting the BER distribution at high SNR by simulation will be very time consuming. In this case, a moderate SNR should be selected at the cost of little degradation of BER at high SNR.

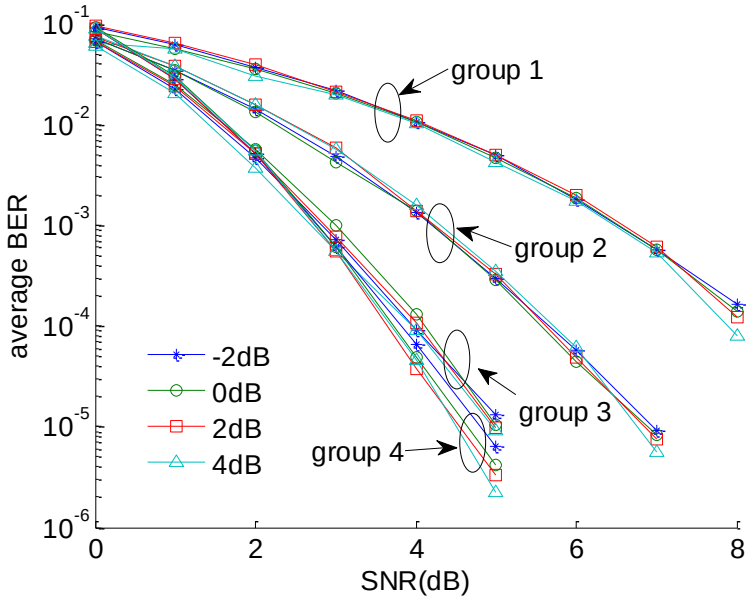


Figure 3. Comparisons of the Improvements after Reordering the Bits Based on Four Different BER Distributions Produced at Four Different SNRs

4. Simulation Results and Comparison with Other References

In this Section we will give several examples referred to the references [3] and [9] to verify the effectiveness of the proposed scheme. The first two examples are the comparisons of our method with those proposed in [3] where interleaver pruning is used for construction of variable-length turbo codes. The last example is the comparison of our method with those proposed in [9] where the shortened– extended Hamming codes are used to get different encoding block sizes. Monte Carlo simulations are performed. AWGN is assumed in the simulations.

Example 1. For consistency, we choose the same Turbo code and interleaver as in [3]. The interleaver length is 160. The BER distribution for reordering is produced by simulation at SNR=1.5dB. Figure 4 shows the BER as a function of the shortened length at a SNR of 4 dB. The length step is 20. The solid curve is produced by the method proposed in this paper. The dash line is produced by the method proposed in [3]. From the Figure we can see that, although there are some differences between the two curves at different shorten lengths, they have nearly the same performance.

Example 2. In this example, we will give our simulation results to compare with those in Figure 11 in [3]. The RSC used in the turbo code construction and the UMTS interleaver are specified in 3GPP TS 25.212 v3.11.0 standard specification. The UMTS turbo code of length=320 is chosen as the mother code. The BER distribution for reordering is produced by simulation at SNR=1.2dB. BCJR decoding algorithm is used

and the number of iteration is 10. In [3], 12 iterations of the sliding-window BCJR algorithm are used. The shortened input lengths are 80; 160 and 320(without shorten).

The simulation results are displayed in Figure 5. For comparison, we have copied the curves from Figure 11 of [3] to Figure 5. From the Figure we can see that below 1dB for length=80 and 2dB for length=160, the curves produced by the method proposed in this paper are lower than those produced in the [3]. After 1 and 2dB, the curves are little higher than those produced in the [3] separately. Generally, the performance of our method is very similar to those specified in [3] while the shortening strategies are completely different. Hence, the pruning strategy presented in this letter can be used as an alternative implementation of a variable- length interleaver for the UMTS standard with very similar performance at least at the lengths tested.

Example 3. To compare our method with those proposed in [9] where the shortened–extended Hamming codes are used, we should choose a mother code firstly. In Figure 3 and Figure 4 of [9], a square (turbo product code) TPC $(64, 57, 4)^2$ is used. The code length is 64^2 =4096 and the input length is 57^2 =3249. The code rate is about 0.79. According to these parameters, the mother code is constructed as follow. A 3GPP interleaver with length 4096 is adopted. The code rate is 1/3 and the generator polynomial of component encoder is $G(D) = (1+D+D^3)/(1+D^2+D^3)$. The code length is $4096\times 3=12288$ (the terminating bits are not considered). To match the code length 4096, some parity check bits should be deleted. Now we take the puncturing pattern as (1000000000; 0000010000). The puncturing period is 10. After puncturing, the code length becomes $4096 + 2\times409=4914$. And the code rate becomes 4096/4914 = 0.833. We take this code as the mother code. Then simulating this mother code at SNR=3dB to produce the BER distribution to reorder the shortened information bits. Finally, by shortening the 4096 to 3249, we get a shortened turbo code that has code length (3249+818) = 4067 and code rate = 3249/4067=0.799 which are nearly the same length and code rate as the one used in [9].

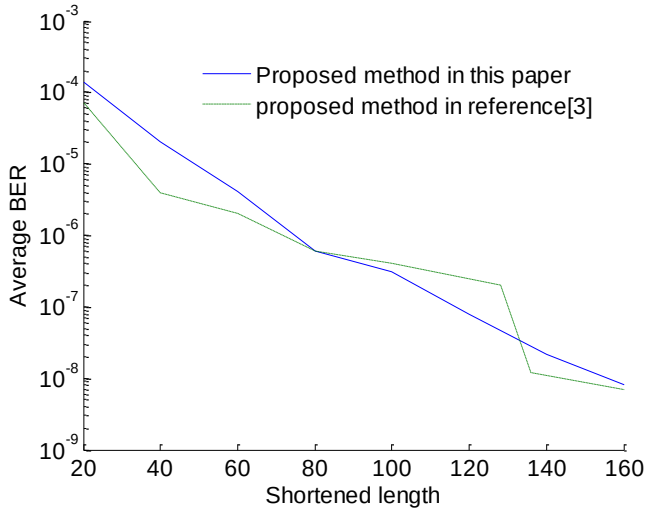


Figure 4. The BER as a Function of the Shortened Length at a SNR of 4 dB

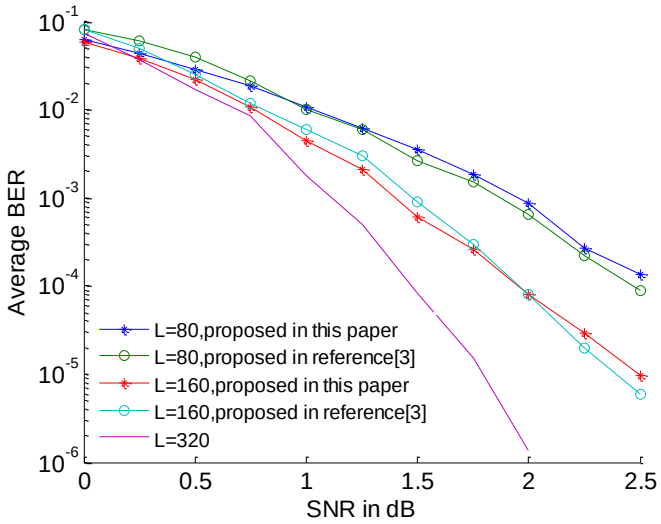


Figure 5. BER Performance Comparison of a Variable-Length Turbo Code at Lengths 80; 160; and 320

Apparently, the selection of the puncturing patterns will affect the BER performance. This is because that different puncturing pattern will produce different weight spectrum. But it is difficult to analysis the weight spectrum for a code with long length with different puncturing patterns. Here we have done the simulation with several different puncturing patterns, such as pp1 = (1000000000; 0000010000); pp2 = (1000000000; 0100000000), pp3 = (0000010000; 1000000000); pp4 = (1000010000; 0000000000) and pp5=(0000000000; 1000010000), to observe the BER performance. We find that at low and moderate SNRs, the performances of the different puncturing patterns are nearly the same. At high SNR, the behaviors are different according to different puncturing patterns. After comparison, the puncturing patter pp1 has the best performance among the puncturing patterns mentioned above.

The simulation results with various iterations in the AWGN channel is depicted in Figure 6 and Figure 7. The curves produced by the method proposed in [9] are with the Chase algorithm. The decoding algorithm in this paper is BCJR. In Figure 6, the number of test patterns is 16. The number of iteration is from 1 to 4. From the Figure we can see that, for the curves proposed in this paper, no matter what the number of iteration is, there are significant improvements at low and moderate SNR regions to the curves proposed by [9]. At high SNR, for example, SNR=3.5 for iteration =4, the BER curve appears the error floor.

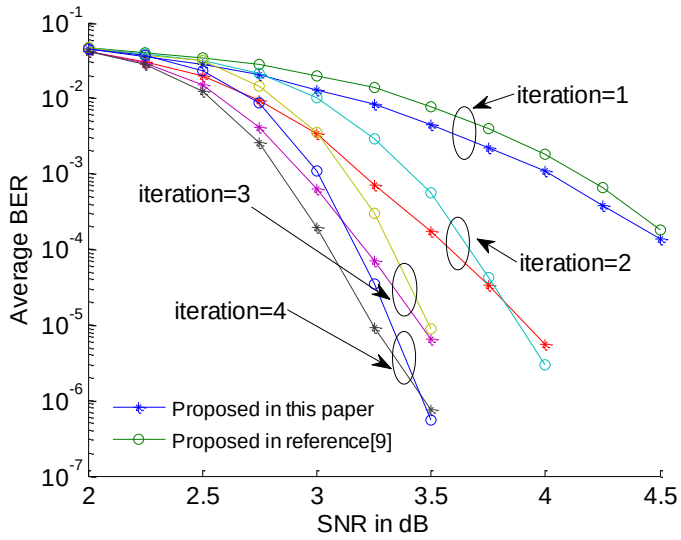


Figure 6. BER Performance Comparison of a v TPC (64, 57, 4)^2 and a Variable-Length Turbo Code with Various Iterations in the AWGN Channel

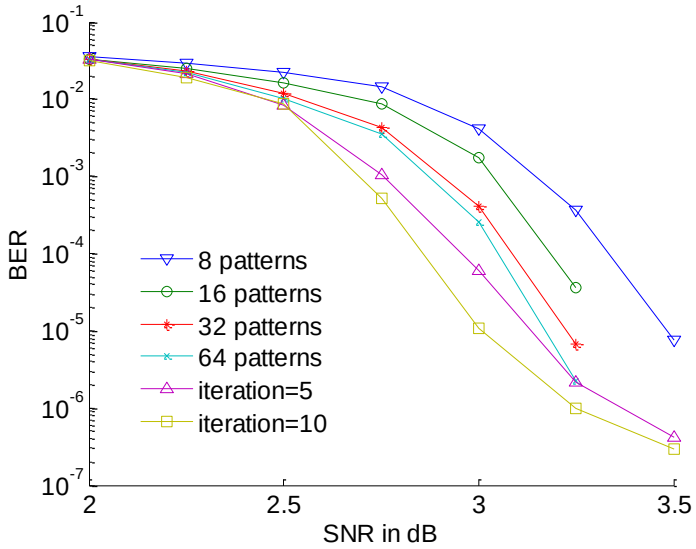


Figure 7. BER Performance Comparison of a v TPC (64, 57, 4)^2 and a Variable-Length Turbo Code with 5 and 10 Iterations in the AWGN Channel

In Figure 7, there are six curves. The first four curves are copied from Figure4 of [9] that is produced with 8, 16, 32 and 64 test patterns separately. The other two curves noted by iteration=5 and 10 are produced by the method proposed in this paper with iterations of 5 and 10 respectively. It is clear that even with 5 iterations, the BER performance is much better than the curve with 64 test pattern before SNR=3.25dB. As for the curve with 10 iterations, the BER is improved nearly 2 orders of amplitude at SNR=3dB.

5. Conclusion

In this paper, we propose a very simple method to realize the variable-length turbo codes based on the inherent feature of unequal error protection of turbo codes. Simulation results show that, compared with the scheme proposed in [3] where realizations of shortening turbo codes are based on designing specific interleavers

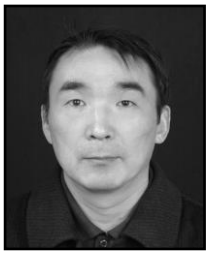
with variable interleaving span, they have the similar BER performance; compared with the scheme proposed in [9] where realizations of shortening turbo codes is based on using shortened–extended Hamming codes, the BER performances are much better at low and moderate SNR regions. Hence, the pruning strategy presented in this letter can be used as an alternative implementation of a variable-length code for the UMTS standard

References

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes," in Proc. 1993 IEEE Int. Conf. Communications, Geneva, Switzerland, May **(1993)**, pp.1064–1070.
[2] S. Benedetto and G. Montorsi, "Unveiling turbo codes: Some results on parallel concatenated coding schemes," IEEE Trans. Inform. Theory, vol. 42, March **(1996)**, pp. 409–428.
[3] F. Daneshgaran and P. Mulassano, "Interleaver Pruning for Construction of Variable-Length Turbo Codes," IEEE Transactions on Information Theory, vol. 50 no. 3, March **(2004)**, pp. 455-467.
[4] D. Divsalar and F. Pollara, "Turbo codes for PCS applications," in Proc.1995 IEEE Int. Conf. Communications, Seattle, WA, May **(1995)**, pp. 54–59.
[5] M. Jeanne, J. C. Carlach and P. Siohan, "Joint Source-Channel Decoding of Variable-Length Codes for Convolutional Codes and Turbo Codes," IEEE Transactions on Communications, vol. 53 no. 1, January **(2005)**, pp. 10-15.
[6] L. Guivarch, J. C. Carlach and P. Siohan, "Joint source-channel soft decoding of Huffman sources with turbo-codes," in Proc. IEEE Data Compression Conf., Snowbird, Utah, March **(2000)**, pp. 83–92.
[7] M. Jeanne, J. C. Carlach, P. Siohan and L. Guivarch, "Source and joint source-channel decoding of variable length codes," in Proc. Int. Conf. Commun., New York, NY, vol. 2, April **(2002)**, pp. 768–772.
[8] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimal decoding algorithm," IEEE Trans. Inf. Theory, vol. IT-13, February **(1967)**, pp. 260-269.
[9] C. Xu, Y. C. Liang and W. S. Leon, "Shortened Turbo Product Codes: Encoding Design and Decoding Algorithm," IEEE Transactions on Vehicular Technology, vol. 56 no. 6, November **(2007)**, pp3495-3501.
[10] R. Pyndiah, "Near-optimum decoding of product codes: Block turbo codes," IEEE Trans. Commun., vol. 46 no. 8, August **(1998)**, pp. 1003–1010.
[11] R. Pyndiah, A. Glavieux, A. Picart and S. Jacq, "Near optimum decoding of product codes," in Proc. GLOBECOM, San Francisco, CA, vol. 1, November **(1994)**, pp. 339–343.
[12] R. H. M. Zaragoza, "The Art of Error Correcting Coding. Hoboken," NJ: Wiley, **(2002)**.
[13] S. A. Hirst, B. Honary and G. Markarian, "Fast chase algorithm with an application in turbo decoding," IEEE Trans. Commun., vol. 49 no. 10, October **(2001)**, pp. 1693–1699.
[14] C. Argon and S. W. McLaughlin, "An efficient Chase decoder for turbo product codes," IEEE Trans. Commun., vol. 52 no. 6, June **(2004)**, pp. 896–898.
[15] X. Jaspar and L. Vandendorpe, "Impact of Variable Length Codes on the Interleaving Gain of Turbo Systems: The Concept of Bounded Spectrum," IEEE Trans. on Comm., vol. 59 no. 7, July **(2011)**, pp. 1796-1806.
[16] X. Jaspar and L. Vandendorpe, "Impact of Variable Length Codes on the Interleaving Gain of Turbo Systems: The Concept of Bounded Spectrum," IEEE Transactions on Communications, vol. 59 no. 7, July **(2011)**, pp. 1796-1806.
[17] J. Liu, G. Tu, C. Zhang and Y. Yang, "Joint Source and Channel Decoding for Variable Length Encoded Turbo Codes," Hindawi Publishing Corporation EURASIP Journal on Advances in Signal Processing Volume 2008, Article ID 149839, pages1-7 doi: 10.1155/ 2008/ 149839, **(2008)**.
[18] W. Zhang and X. Shao, "Unequal Error Protection of JPEG2000 Images Using Short Block Length Turbo Codes," Communications Letters, IEEE, vol. 15 no. 6, **(2011)**, pp. 659-661.

Authors

Shao Xia, She received her M.S. degree and B. S. degree from Zhengzhou University in 2007 and 1992 separately. Currently she is with the Department of Information Engineering, North China University of Water Resources and Electric Power. Her research focuses on key techniques for telecommunication theory and engineering.



Zhang Weidang, He received his PhD from Xidian University in 2005. He academically has visited Simon Fraser University from 2008 to 2009. Currently he is with the School of Information Engineering, Zhengzhou University where he is a professor and his research interests include information theory and coding theory.