Paper

Application of convolutional interleavers in turbo codes with unequal error protection

Sina Vafi and Tadeusz A. Wysocki

Abstract— This paper deals with an application of convolutional interleavers in unequal error protection (UEP) turbo codes. The constructed convolutional interleavers act as block interleavers by inserting a number of stuff bits into the interleaver memories at the end of each data block. Based on the properties of this interleaver, three different models of UEP turbo codes are suggested. Simulation results confirm that utilizing UEP can provide better protection for important parts of each data block, while significantly decreasing the number of stuff bits.

Keywords— convolutional interleavers, unequal error protection, turbo codes.

1. Introduction

Unequal error protection (UEP) is introduced as an efficient technique for forward error correcting (FEC) codes to suitably protect encoded data based on their importance against channel errors. This is specifically utilized in the transmission of the compressed information such as voice, video and multimedia services which are very sensitive to bit and burst errors.

Among the known channel codes, turbo codes are introduced as effective FEC codes having good performance in error reduction. The turbo code is basically constructed by two recursive systematic convolutional (RSC) codes which are linked by an interleaver [1]. When the UEP property is implemented for the turbo code, a different interleaving compatible with the data length and determined for each protection level should be conducted in addition to the puncturing process [2]. To date, several methods have been suggested mainly for conventional block interleavers, like allocating an exclusive interleaver for each level or a single interleaver for all levels, where the interleaver length is adjusted for different levels. For the block interleaver with a fixed permutation, an interleaver for each level has been proposed in [2], while a circular-shift interleaver for all protection levels usable for the short data lengths is suggested in [3]. In addition, a suitable interleaver for all protection levels has been designed, providing a UEP turbo code without the need for a puncturing process [4].

In contrast to the fixed permutation rule, it is possible to implement interleavers with random permutations for a code with the variable data length. Semi-random interleavers are known as the most efficient interleavers with random permutation. In this type of interleaver, the distance between two adjacent permuted bits should not be less than an allocated value. The best performance of this interleaver with the length L is achieved when the minimum distance is set to the $\lfloor\sqrt{\frac{L}{2}}\rfloor$ value [5]. A structure of the semi-random interleavers usable for permutation of the data blocks with the variable length has been proposed in [6]. The obtained interleaver is named the prunable semi-random interleaver. In this interleaver, a semi-random interleaver according to the shortest data length is designed. Then for the longer lengths, the new required position is randomly inserted. In this interleaver, if after several runs the selected positions do not satisfy the appointed minimum distance, the minimum distance value will be decreased and the above procedure is followed based on the new minimum distance. This reduction degrades the code performance and in order to overcome this problem, a new algorithm has been introduced to apply the semi-random interleaver for different data block lengths, without decreasing the threshold value [7]. Recently, Dioni and Benedetto presented a modification on the prunable interleaver, which improves the code performance with less complexity [8].

The main issue of the interleaver design for the UEP turbo code application is related to the flexibility of adjusting its specifications according to the varying length of data blocks. In contrast to the block interleavers, convolutional interleavers are designed with less complexity and more flexibility to adjust their structures with the length variations of data blocks. Due to the non-block behavior of the convolutional interleaver, turbo codes constructed with these interleavers are analyzed from the continuous performance point of view. The continuous analysis of the turbo code shows that it has a similar performance to the block-wise performance of the code, especially for the constituent RSC codes with the low constraint length value. In order to simplify analysis of the turbo code, convolutional interleavers are designed as block interleavers through the insertion of enough stuff bits at the end of each data block returning the interleaver memories to the zero state. This property makes it possible to utilize conventional iterative decoding techniques applied for the block-wise operation of the turbo code. Based on the application of the convolutional interleaver, three different techniques are presented to design the UEP turbo codes.

In the first technique, only one interleaver for all protection levels is considered. In this technique, different puncturing patterns are applied providing different code rates for the protection levels. In order to improve the turbo code performance, a number of the interleaver lines – which represent the interleaver period value and determine the overall number of stuff bits – should be considered proportional to the data block length. Since the length of protection levels usually differs from each other, the application of an interleaver will not guarantee the provision of a suitable performance for all protection levels.

In the second technique, a single interleaver is allocated for each level of the protection. An interleaver compatible with the longest length of the protection levels is designed. Then based on the interleaver properties, interleavers with the shorter periods relevant to the length of other levels are designed without increasing any complexity in the design. Due to the independent interleavers been designed for each level, this technique has more flexibility to be utilized for applications with high variations of data block lengths.

In addition, since increasing the interleaver period affects the code performance, it is possible to define the new model of UEP by applying interleavers with different periods while the puncturing pattern is kept identical for all levels.

In this paper, performance of the proposed techniques to design the UEP turbo codes is verified. Based on the simulation results, the best suitable model corresponding to the specifications of the protection levels is determined. Our simulations confirm that the first technique is more applicable for protection levels, where data lengths are similar, while the second technique is more reliable for varying data block lengths. The third technique can be utilized when some data parts for a given protection level need more protection than other data parts. The organization of the paper is as follows: Section 2 describes the basic structure of convolutional interleavers and explains their application in the construction of the three techniques to design UEP turbo codes. In Section 3 performance of the 4-state turbo code $(1, \frac{5}{7})$ employing three mentioned techniques is verified based on the maximum-likelihood iterative decoding. Finally, Section 4 concludes the paper.

2. Convolutional interleaver structure

Convolutional interleavers consist of T parallel lines which define their period. Each line of these interleavers have conventionally M memory units more than the previous line, which define the space value parameter of the interleaver. Hence, depending on the distribution of input data to each line of the interleaver, the interleaved input data appear in different time slots at the interleaver output. Figure 1 shows the convolutional interleaver structure with the period $T = 8$ and space value $M = 1$. In order to make isolated interleaved data, some stuff bits are inserted at the end of each input data block returning the memories to the zero state. Then, an optimization is carried out through the deletion of zero stuff bits at the end part of

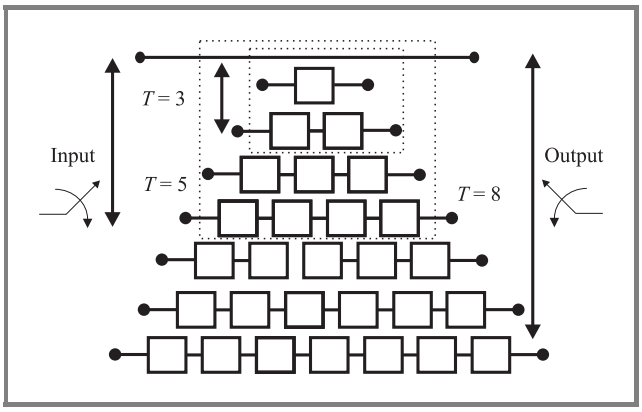


Fig. 1. Consideration of convolutional interleaver $(T = 3,\ M = 1)$ and $(T = 5,\ M = 1)$ from the interleaver $(T = 8,\ M = 1)$.

the interleaved data, reducing the number of stuff bits to the number of the interleaver memories. Figure 2 shows the optimized interleaving procedure for the interleaver $(T = 8,\ M = 1)$ and the length $L = 57$.

The convolutional interleaver with a specific period and space value has the flexibility to interleave data blocks with different lengths. In turbo code applications, when the encoded data blocks, with the variable lengths obtained from an interleaver are punctured with different rates, the UEP turbo codes can be achieved. Conducted simulations of turbo codes with different interleaver lengths indicate that the increment of the data block length, and period of the convolutional interleaver should be increased to provide sufficient performance for the code with a reasonable number of stuff bits [9]. This is more sensitive for an interleaver with a short data block length and leads to a design of an interleaver compatible with the required performance of the code with the longest data block length for the given protection level.

However, since data with the highest protection level require the lower code rate, the data block length is normally considered shorter at this level than at other levels. Hence, designing a convolutional interleaver based on the longest length for all protection levels, increases the number of stuff bits for the levels with shorter lengths and can result in producing the overall number of stuff bits greater than the number of valid data allocated to that level. This is observed when the length variations between different levels is relatively high. Therefore, the convolutional interleaver applied for this type of UEP turbo code is designed based on the shortest block length for all protection levels.

In order to apply an interleaver corresponding to the data specification of each level, it is necessary to employ an independently designed interleaver for each level. It is easily observed that by choosing some lines of an interleaver with the higher period another convolutional interleaver

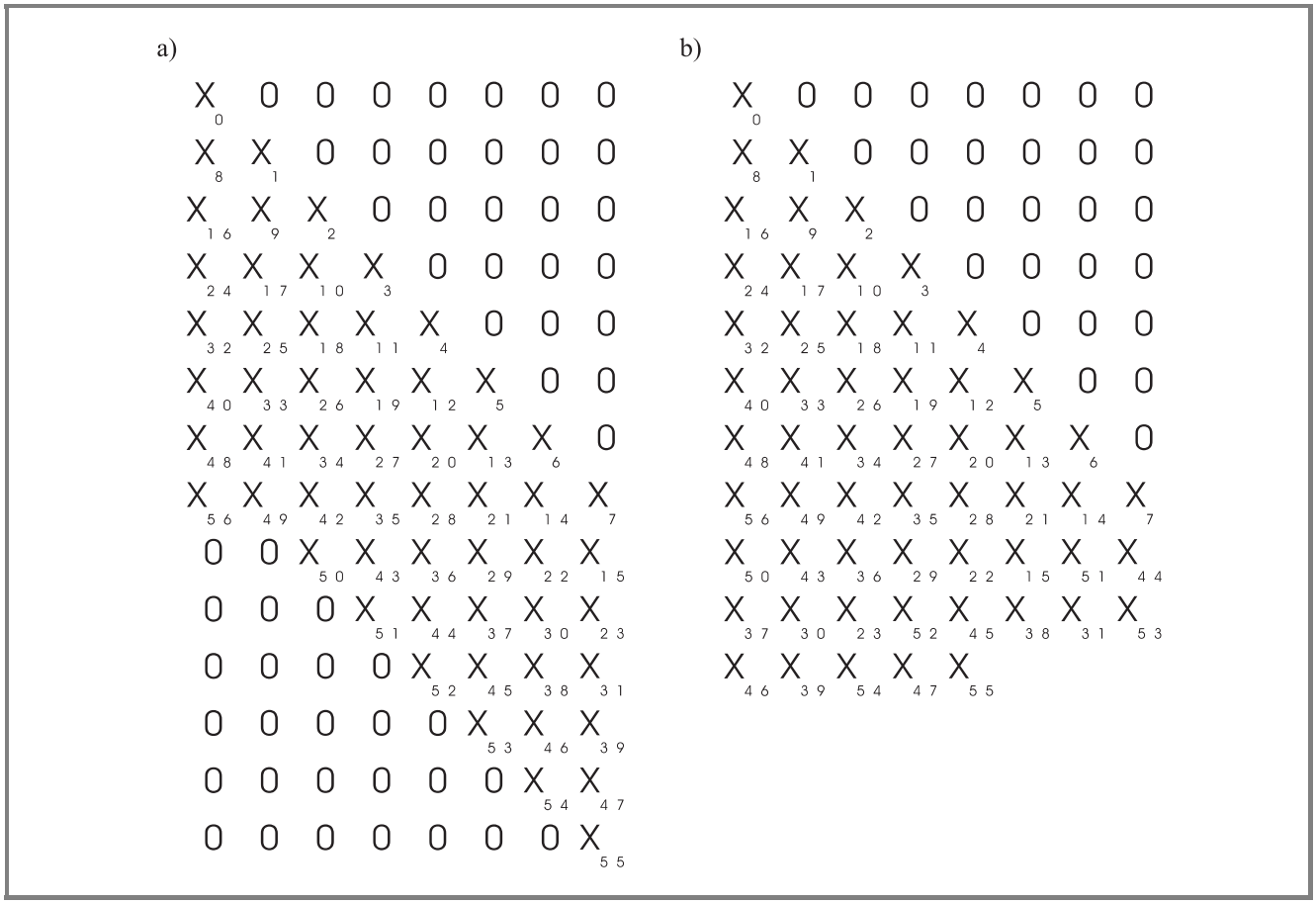


Fig. 2. Interleaved data block with an interleaver $(T = 8,\ M = 1)$ and length $L = 57$: (a) non-zero bit deletion; (b) zero bit deletion.

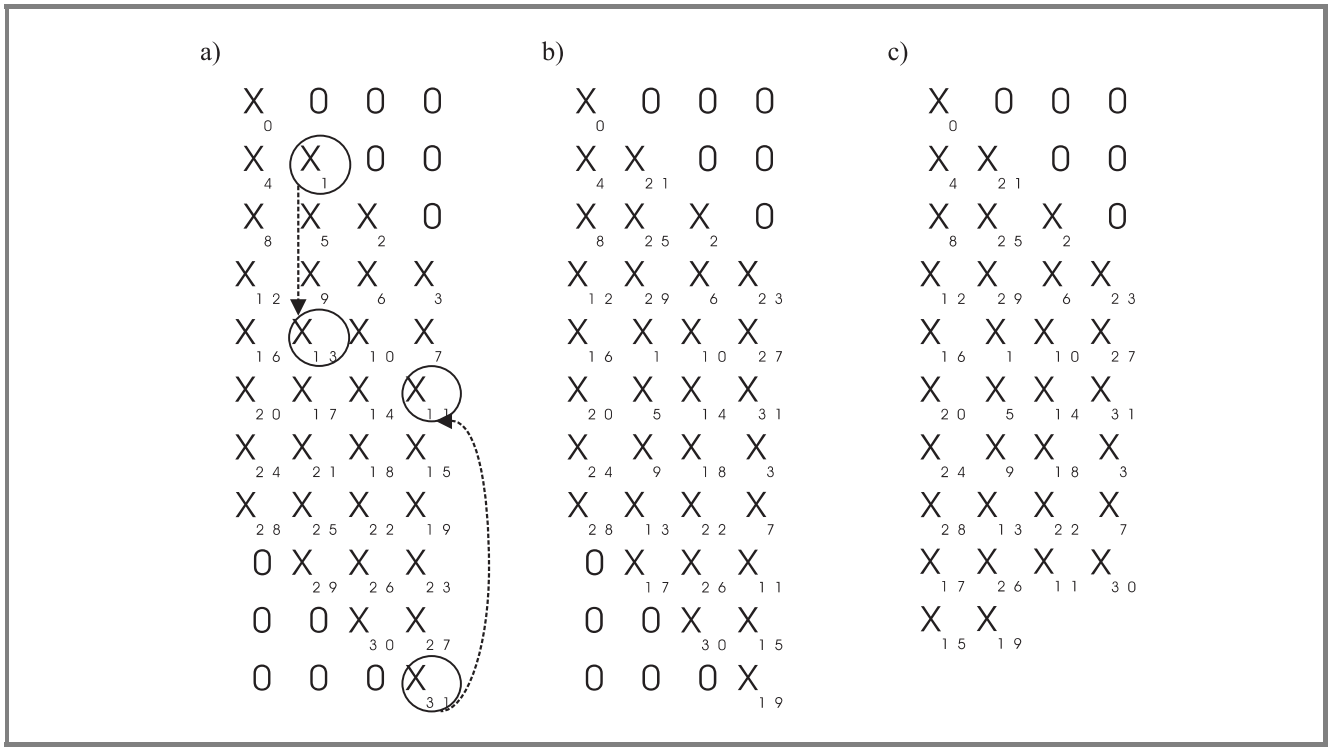


Fig. 3. Modification procedure for the interleaver $(T = 4,\ M = 1)$: (a) interleaved data length $L = 32$; (b) shifted even column bits equal to $3*T$; (c) deletion of zero bits at the end part of the interleaver.

with a shorter period is obtained. For example, Fig. 2 shows that convolutional interleavers $(T = 3,\ M = 1)$ and $(T = 5,\ M = 1)$ can be obtained from the convolutional interleaver $(T = 8,\ M = 1)$ when the relevant input bit streams are distributed to the first 3 and first 5 lines, respectively. These interleavers are created by controlling the distribution of input data blocks to some of the interleaver lines to generate different interleaved data. Interleaved data obtained from different periods are specifically punctured to provide UEP turbo codes. Based on the above observation, many interleavers with shorter periods can be constructed from an original interleaver with a longer period. For simplicity, interleavers with the space value 1 $(M = 1)$ are considered, where the distribution of data always starts from the line without the memory.

Despite applying different puncturing patterns, an interleaver for each level with different periods and a fixed code rate for all levels to provide different protection levels is applied. In this case, for the highest protection level, an interleaver with the longest period is designed such that it produces a reasonable number of stuff bits. Then, based on the order of other protection levels, interleavers with shorter periods are constructed by selecting of some lines of the original interleaver. For example, in Fig. 2, contrasting to the previous technique, the interleaver $(T = 8,\ M = 1)$ is applied for the highest protection level, while the interleavers $(T = 5,\ M = 1)$ and $(T = 3,\ M = 1)$ are used for to the second and third protection levels, respectively.

For each technique, a modification can be performed to the interleavers, improving the code reliability with the lower number of stuff bits. This is generally accomplished by shifting the bits of the interleaved data located at the even columns. Figure 3 shows the modification procedure of the interleaver $(T = 4,\ M = 1)$. First, the input data blocks are regularly interleaved and then the bits located at the even columns are shifted by $3{*}T$ units. Similarly to the proposed modification in [10], the number of shifted bits is considered even. In case of an odd number of bits, the zero stuff bits located on the top of the first bit of even columns are involved in the modification process. Finally, zero stuff bits located at the end part of the interleaved data are deleted to optimize the number of stuff bits.

3. Simulation results

In simulations, convolutional interleavers with short and long data block lengths have been applied for the three mentioned types of UEP with the 4-state tubo code $(1, \frac{5}{7})$. For the code, trellis termination and truncation is utilized in the first and the second RSC encoders, respectively. To reduce the number of stuff bits to be equal to $\frac{T(T-1)M}{2}$, they will be removed from the end part of the systematic and the first parity data, since stuff bits are inserted after trellis termination and do not have any effect on the code performance. For simplicity, the effect of these stuff bits for the systematic and the first parity data are considered getting the exact code rate at each level. At the decoder, the iterative decoding is accomplished and the BER is only calculated based on the length of the original bit stream without stuff bits. Regarding this structure, the code rate of each level is calculated by

$$R_i = \frac{l_i}{n_{\mathbf{P_i}} + n_{\mathbf{Q_i}} + n_{\mathbf{O_i}}}, \tag{1}$$

where l_i, $n_{\mathbf{O_i}}$, $n_{\mathbf{P_i}}$ and $n_{\mathbf{Q_i}}$ denote the length of the puncturing matrix, length of the matrix of 1 s for the systematic data, and number of bit 1 in puncturing matrices of the ith level for the first and second RSC encoder with the length of l_i, respectively. For the short and long data block lengths, 3 and 4 protection levels have been considered, respectively. Tables 1–5 give specifications of puncturing patterns and protection levels of each UEP type.

Table 1
Puncturing patterns for different protection levels

Rate	l	**P**	**Q**	**O**
1/3	1	[1]	[1]	[1]
2/5	2	[1 1]	[1 0]	[1 1]
1/2	2	[1 0]	[0 1]	[1 1]
2/3	4	[1 0 0 0]	[0 0 1 0]	[1 1 1 1]
3/4	6	[1 0 0 0 0 0]	[0 0 0 1 0 0]	[1 1 1 1 1 1]

Table 2
Specifications of 3 protection levels with the fixed interleaver period and different code rates

Level	Length (L')	Interleaver period (T)	Rate (R)
1	32	4	1/3
2	48	4	1/2
3	112	4	2/3
Overall	192	4	$\approx 1/2$

Table 3
Specifications of 3 protection levels with different interleaver periods and code rates

Level	Length (L')	Interleaver period (T)	Rate (R)
1	32	4	1/3
2	48	5	2/5
3	112	6	1/2
Overall	192	5	$\approx 1/2$

Table 4
Specifications of 3 protection levels with different interleaver periods and the fixed code rate

Level	Length (L')	Interleaver period (T)	Rate (R)
1	32	6	1/3
2	48	5	1/3
3	112	4	1/3
Overall	192	4	$\approx 1/3$

Table 5
Specifications of 4 protection levels with different interleaver periods and code rates

Level	Length (L')	Interleaver period (T)	Rate (R)
1	128	7	1/3
2	512	14	1/2
3	1024	20	2/3
4	2432	30	3/4
Overall	4096	25	$\approx 2/3$

In order to compare performance of the protection levels with the equal error protection (EEP) codes, the overall specification of the code should be determined. With the employment of puncturing at each level, the average code rate with l protection levels is determined by [11]

$$R_{av} = \frac{\sum_{i=1}^{l} L_i}{\sum_{i=1}^{l} \frac{L_i}{R_i}}, \tag{2}$$

where $L_i = L'_i + N_i$ denotes the data block length of the ith level after stuff bit insertion, obtained from summation of the original input data block length L'_i and the number of stuff bits N_i. The above protection parameters have been simulated by the soft output Viterbi algorithm (SOVA) [12] with 8 iterations in the presence of additive white Gaussian noise (AWGN). The equivalent interleaver specifications can be determined based on the number of stuff bits or the interleaver periods and the data block lengths for each level. In this case, the equivalent interleaver period for the overall rate is given by

$$T_{av} = \frac{\sum_{i=1}^{l} L_i T_i}{\sum_{i=1}^{l} L_i}, \tag{3}$$

where T_i represents the interleaver period of the ith level.

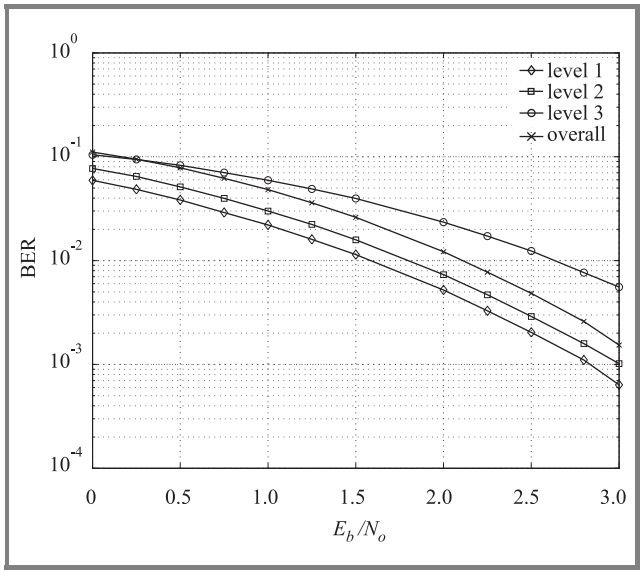


Fig. 4. Unequal error protection for 4-state turbo codes with the fixed interleaver period ($T = 4$) and different rates.

Table 6
Shifted unit values for the even columns bits of the interleaved data of different interleavers

Interleaver period T	4	5	6	7
Shifted unit value	3*T	4*T	4*T	10*T

Figure 4 shows performance of the UEP turbo code based on the fixed interleaver period ($T = 4,\ M = 1$) and the variable code rates of the protection levels. Also, modifications are performed to the interleavers at each level. In the modification process, those shift values which provide better reliability for the code performance are selected. Table 6 gives specification of modifications applied for different interleaver periods. The graphs of Fig. 4 show that levels 1 and 2 are better than the overall performance of the code by 0.5 dB and 0.25 dB, respectively.
Figure 5 illustrates the code performance with different interleavers and code rates applied for protection levels based on the specifications in Table 3. In this figure, level 1 has 0.25 dB better performance than the overall level. When

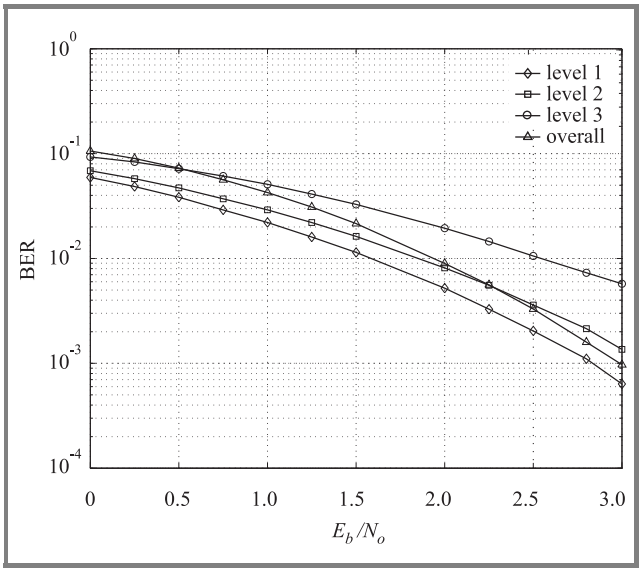


Fig. 5. Unequal error protection for 4-state turbo codes with different interleaver periods and code rates.

the modification is applied to the interleaver[1] of the equivalent EEP for the turbo code, for the overall level, the distance between adjacent bits of the original bit stream increases due to the longer length of every column of the interlever. Therefore a higher weight for the equivalent code with the overall protection level is produced, which consequently improves the code performance at the medium to high signal to noise ratio region.
Figure 6 shows the code performance when different protection is achieved through different interleaver periods with the code rate fixed for all levels. Level 1 has a performance better by 0.5 dB than the overall performance, while

[1]This means that the EEP turbo code has the performance equivalent to the average performance of the considered UEP code.

the level 2 is slightly better than the overall performance. Also, in comparison with the two other methods, level 3 performance has been efficiently improved. Figure 7 shows the performance of the UEP turbo code with four level protection and interleaver length $L = 4096$. In this example, modification is only carried out for level 1. This is accomplished by shifting bits located in the even interleaver lines by $10*T$. In this figure, levels 1 and 2 have 1 and 0.5 dB better performance than the average code performance, while number of stuff bits at these levels has been reduced by 93% and 69.6%, respectively.

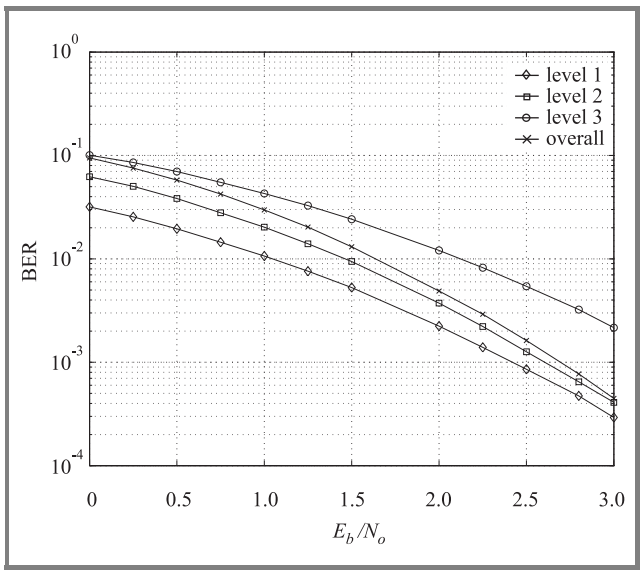


Fig. 6. Unequal error protection for 4-state turbo codes with different interleaver periods and the fixed rate $R = \frac{1}{3}$.

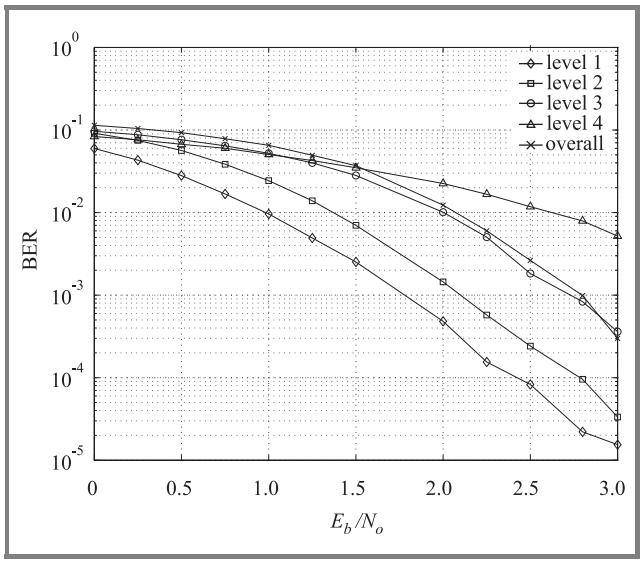


Fig. 7. Unequal error protection for 4-state turbo codes with overall length $L = 4096$.

In addition, level 3 with the lower period and consequently less stuff bits has behavior close to average for the code. However, due to application of the higher code rate and puncturing most of the encoded data, level 4 has the worst performance.

The obtained results from different types of UEP turbo codes indicate that this interleaver has the flexibility to be utilized in UEP turbo code applications with short and long data block lengths. The results represent that the first and third proposed UEP types are useful for the protection levels having similar data block lengths. This is specifically observed for the first type of UEP, when only one interleaver is implemented for all the levels with a lower number of stuff bits and less complexity. However, type 3 improves the performance of every level increasing with and reasonably increases the number of stuff bits.

Comparing the results obtained from the Figs. 5 and 7 indicates that the second suggested technique is more applicable for the cases when the data lengths vary significantly for different protection levels. In such cases, the technique effectively protects the important parts of the data blocks with the shorter periods and lower numbers of stuff bits.

4. Conclusions

In this paper a simple and efficient techniques to design UEP turbo codes with convolutional interleavers was presented. These techniques are implemented based on the interleaver properties and their performance has been examined for the short and long interleaver lengths. The simulation results confirm that the convolutional interleavers have the flexibility to be utilized for different specifications of protection levels. Every technique improves the code performance for the most important parts of data with a shorter period and lower number of stuff bits than the interleaver applied for the EEP turbo codes.

References

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: turbo codes", in *Int. Conf. Commun. ICC*, Geneva, Switzerland, 1993, pp. 1064–1070.

[2] A. S. Barbulescu and S. S. Pietrobon, "Rate compatible turbo codes", *IEE Electron. Lett.*, vol. 31, no. 7, pp. 535–536, 1995.

[3] M. Salah, R. A. Rains, and A. Temple, "A general interleaver for equal and unequal error protections of turbo codes with short frames", in *Int. Conf. Inform. Technol. Cod. Comput. ITCC*, Las Vegas, USA, 2000, pp. 412–415.

[4] M. Grangetto, E. Magli, and G. Olmo, "Embedding unequal error protection into turbo codes", in *35th Asil. Conf. Sig. Syst. Comput.*, Pacific Grove, USA, 2001, vol. 1, pp. 300–304.

[5] S. Dolinar and D. Divsalar, "Weight distributions for turbo codes using random and nonrandom permutations", *TDA Progr. Rep.*, pp. 56–65, 1995.

[6] M. Ferrari, F. Scalise, and S. Bellini, "Prunable s-random interleavers", in *IEEE Int. Conf. Commun. ICC*, New York, USA, 2002, vol. 3, pp. 1711–1715.

[7] P. Popovski, L. Kocarev, and A. Risteski, "Design of flexible-length s-random interleaver for turbo codes", *IEEE Commun. Let.*, vol. 8, no. 7, pp. 461–463, 2004.

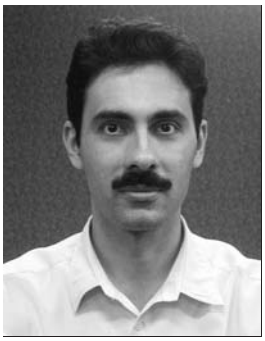
[8] L. Dioni and S. Benedetto, "Design of fast-prunable s-random interleavers", *IEEE Trans. Wirel. Commun.*, vol. 4, pp. 2540–2548, 2005.

[9] S. Vafi and T. Wysocki, "On the performance of turbo codes with convolutional interleavers", in *11th Asia-Pacific Conf. Commun. APCC*, Perth, Australia, 2005, pp. 222–226.

[10] S. Vafi and T. Wysocki, "Modified convolutional interleavers and their performance in turbo codes", in *2nd SympoTIC'04*, Bratislava, Slovakia, 2004, pp. 54–57.

[11] G. Caire and E. Biglieri, "Parallel concatenated codes with unequal error protection", *IEEE Trans. Commun.*, vol. 46, no. 5, pp. 565–567, 1998.

[12] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary block and convolutional codes", *IEEE Trans. Inform. Theory*, vol. 42, no. 2, pp. 429–445, 1996.



Sina Vafi received the B.E. degree in telecommunications engineering from the Khajeh Nasireddin-e Toosi (KNT) University of Technology, Tehran, in 1996 and M.E. in electronics engineering from the Islamic Azad University (South Tehran branch) in 1999. From 1996 to 2002, he was involved in research, design and implementation of digital transmission systems in IRAN Telecommunication Research Centre (ITRC) and Huawei Technologies. Since 2002, he has been studying towards a Ph.D. degree at the University of Wollongong, Australia. His research interests include digital transmission systems, wireless communications and error control coding.
e-mail: sv39@uow.edu.au
University of Wollongong
Northfileds Ave
Wollongong, NSW 2522, Australia

Tadeusz Antoni Wysocki received the M.Sc.E. degree with the highest distinction in telecommunications from the Academy of Technology and Agriculture, Bydgoszcz, Poland, in 1981. In 1984, he received his Ph.D. degree, and in 1990, was awarded a D.Sc. degree (habilitation) in telecommunications from the Warsaw University of Technology. In 1992, Doctor Wysocki moved to Perth, Western Australia to work at Edith Cowan University. He spent the whole 1993 at the University of Hagen, Germany, within the framework of Alexander von Humboldt Research Fellowship. After returning to Australia, he was appointed a Program Leader, Wireless Systems, within Cooperative Research Centre for Broadband Telecommunications and Networking. Since December 1998 he has been working as an Associate Professor at the University of Wollongong, NSW, within the School of Electrical, Computer and Telecommunications Engineering. The main areas of Doctor Wysocki's research interests include: indoor propagation of microwaves, code division multiple access (CDMA), space-time coding and MIMO systems, as well as mobile data protocols including those for ad hoc networks. He is the author or co-author of four books, over 150 research publications and nine patents. He is a Senior Member of IEEE.
wysocki@uow.edu.au
University of Wollongong
Northfileds Ave
Wollongong, NSW 2522, Australia