터보 부호의 오류 취약 비트 보완 알고리듬

오 왕 록*, 정회원 전 경 훈*, 김 진 우*, 정회원 양 경 철*

Protection Algorithm for Error Prone Bit Positions of Turbo Codes

Wangrok Oh*, Kyungwhoon Cheun* ***Regular Member***, Jinwoo Kim*,
Kyeongcheol Yang*, ***Regular Member***

ABSTRACT

In this paper, we propose a simple protection scheme for error prone bit positions of turbo codes using the error detection capability of the CRC, which is almost always employed in practical systems. The proposed scheme based on bit flipping with CRC offers flexibility on selecting the level of protection. Also, not having send additional parity bits or discarding useful bit positions, it offers the best error performance for a given level of protection.

I. Introduction

Turbo codes, though exhibiting remarkable performance at low SNRs due to spectral thinning, suffer from the so called error floor at moderate to high SNRs due to relatively small free distances [1]. The structure of the assumed rate 1/3 parallel concatenated turbo encoder is shown in Fig. 1. The two constituent recursive systematic convolutional (RSC) codes are identical with generator $(7,5)_8$ and each frame contains N_f bits including CRC bits. The CRC is nearly always employed in practical applications in order to guarantee data integrity to the upper protocol layer or to indicate a decoding failure at the physical layer [3]. It is also quite useful as a decoder iteration stopping criterion [4]. We assume that the employed CRC is sufficiently powerful that essentially all errors are detected. Also, both constituent codes are terminated using the termination scheme proposed in [2].

The decoder employed is basically the standard iterative MAP algorithm [1] with a preset maximum iteration number and an additional process called bit flipping. If a frame passes the CRC test after any given decoder iteration, the decoder iteration is stopped. The bit flipping operation is activated when a frame fails the CRC even after the maximum preset number of decoder iterations. Given a list of error prone bit position (EPBP) patterns, the decoder flips the bits corresponding to each pattern and rechecks the CRC until the CRC check is successful or the decoder runs out of EPBP patterns.

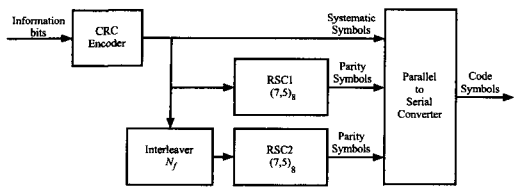


Fig. 1. The structure of the assumed rate 1/3 parallel concatenated turbo encoder.

* 포항공과대학교 전자전기공학과 (cheun@postech.ac.kr)
※본 연구는 국방과학연구소의 지원에 의하여 수행되었음.
JCCI 추천논문

The error floor and the existence of EPBPs are mainly due to specific small weight (usually two and three) input patterns that generate low weight codewords at both of the two RSC outputs. Also contributing to the error floor, though to a lesser degree, is the edge effect [5]. It is invoked by input data patterns resulting in a trellis path diverging from the all-zeros path at RSC1 near the end of the frame being mapped to a similar trellis paths at RSC2.

There have been several previous attempts attacking the problem of EPBPs in (parallel concatenated) turbo codes. One is due to Öberg and Siegel [6] which simply discards the EPBPs, i.e., transmits dummy information bits in these bit positions. This results in a reduction of code rate and wasted energy in the discarded bit positions. All other approaches are based on some form of serial concatenation. In [7] and [8], high rate BCH and the Reed-Solomon codes were respectively used as outer codes to the turbo code in order to lower the error floor. In [9], Narayanan and Stüber improved on the serial concatenation schemes of [7] and [8] using a scheme called selective concatenation. Here, extra protection is provided only for those bit positions identified as being error prone. Based on the fact that EPBPs occur in fixed patterns corresponding to nonzero bit positions resulting in low weight codewords, Kim and Lee in [10] proposed only protecting one representative bit position within each EPBP patterns using single error correcting BCH codes.

There are several shortcomings to the serial concatenation methods. One obvious shortcoming is the reduction in the code rate due to additional parity bits that need to be transmitted, especially for short frame sizes. Also, these schemes must be designed into a system before deployment. Hence, the level of protection, namely the size of the EPBPs protected, is fixed and not adjustable after deployment. Furthermore, with the scheme proposed in [10], extra care must be taken in selecting the particular bit position within the EPBP pattern to be encoded, especially for short frame sizes. This is due to the possibility that some EPBPs occur in multiple EPBP patterns in which case the single error correcting BCH code fails and the scheme breaks down. The proposed scheme based on bit flipping with CRC offers flexibility that serial concatenation techniques do not. Also, not having to send additional parity bits or discard useful bit positions, it offers the best error performance for a given set of EPBPs protected.

Table 1. EPBP patterns for a set of randomly generated interleavers for frame sizes N_f=128 and 1024.

$N_f = 128$			$N_f = 1024$		
EPBP patterns	w	N_w	EPBP patterns	w	N_w
(73, 74, 75)	7	1	(430, 433)	10	2
(16, 18, 20)	9	2	(480, 483)		
(66, 69)	10	4	(97, 100)	12	6
(110, 113)			(334, 340)		
(38, 44)	12	8	(417, 423)		
(39, 45)			(861, 867)		
(51, 54)			(680, 683)	14	9
(82, 83, 87).			(915, 924)		
(67, 72, 77)	13	10	(978, 981)		
(79, 83, 87)			(3, 6)	16	14
(99, 108)	14	14	(637, 646)		
(103, 112)			(794, 806)		
(108, 114)			(921, 933)		
(74, 75, 79).			(934, 940)		
(79, 82).	15	20	(12, 18)	18	28
(37, 38, 39)			(43, 58)		
(59, 60, 67)			(142, 148)		
(85, 89, 90)			(333, 348)		
(89, 90, 94)			(337, 346)		
(94, 107, 108)			(379, 390)		
(1, 10)	16	29	(643, 649)		
(10, 16)			(705, 708)		
(11, 23)			(739, 745)		
(25, 37)			(845, 854)		
(34, 46)			(883, 892)		
(86, 92)			(894, 906)		
(107, 110)			(943, 949)		
(114, 126)			(996, 1005)		
(104, 105, 109).			(1012, 1020)	19	30
			(144, 157, 161)		
			(186, 198)	20	35
			(482, 485)		
			(513, 522)		
			(577, 595)		
			(623, 629)		

II. System Description and Numerical Results

The EPBP patterns for a set of randomly generated interleavers for frame sizes N_f=128 and 1024 are shown in Table 1. All

input patterns of weight one, two and three were investigated. The resulting input EPBP patterns are tabulated in an ascending order according to the resulting codeword weights w up to 16 and 20 for N_f =128 and 1024, respectively. The EPBP patterns corresponding to the aforementioned edge effect are identified with the subscript "*". The accumulated number of codewords up to a particular codeword weight, Nw are also shown and the error prone bit positions that occur in multiple EPBP patterns are underlined.

As described in Section I, the CRC is assumed to detect all decoding errors with probability one. Hence, if we decide to protect the first L EPBP patterns in Table 1, it is guaranteed that the bit flipping algorithm will correct all error patterns corresponding to these entries. For example, if we protect the first two entries in Table 1 for N_f =128, EPBP patterns resulting in codeword weights 7 and 9 are eliminated. Since we have only investigated input patterns of weight up to three, we cannot claim that all codewords of weight up to 9 have been eliminated for this particular example. However, one may be reasonably confident that codewords of small weight are most likely generated by input patterns of weight up to three. Hence, elimination of the EPBP patterns corresponding to low weight codewords will, in most cases, result in an effective increase in the minimum distance of the code.

The additional complexity associated with the bit flipping algorithm is the memory required for the EPBP pattern table and additional CRC computations. The number of additional CRC computations range from the minimum value of one to the maximum value corresponding to the size of the EPBP pattern table. However, for sufficiently large SNRs, this number will be quite small since most errors at relatively high SNRs correspond to low weight codewords. In any respect, the additional complexity required for the bit flipping algorithm for a reasonable EPBP pattern table size is negligible compared to the

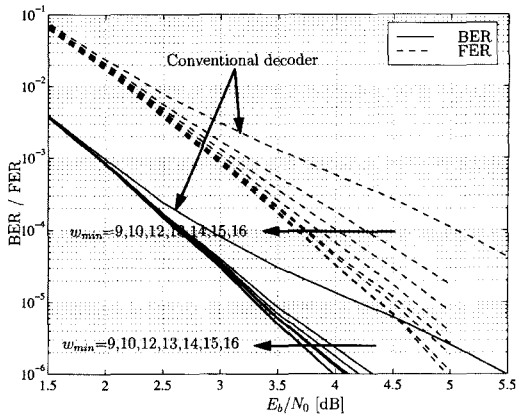


Fig. 2. BER and FER performance of the bit flipping algorithm for N_f=128.

complexity of the original iterative MAP decoder. For the numerical results that follow, we take the preset number of maximum decoder iterations to be 4 and 8 for N_f =128 and 1024, respectively.

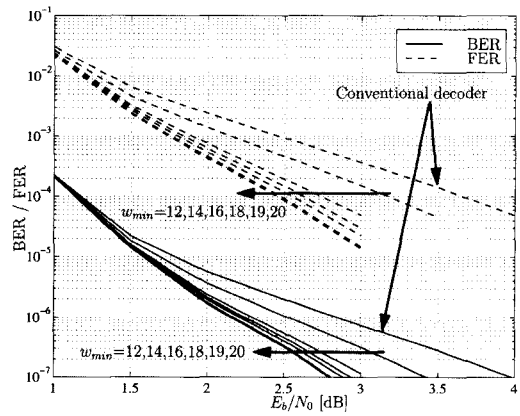


Fig. 3. BER and FER performance of the bit flipping algorithm for N_f=1024.

In Figs. 2-3, we have plotted the simulated BER and the frame error rate (FER) curves for frame sizes N_f =128 and 1024, respectively. The channel is assumed to be an AWGN channel with two-sided power spectral density $N_0/2$ with BPSK modulation. The energy per bit, E_b denotes the received energy per each bit in the frame including the CRC bits. The parameter w_{min} is defined as $w_{min}=w$ meaning that all EPBP patterns in Table 1 up to codeword weight w-1 are included in the decoder EPBP pattern table. We

observe clear gains that may be achieved with the bit flipping algorithm, especially for the FER. Unlike the serial concatenation schemes, the gains shown in these figures are obtainable without any increase in the required bandwidth. Also, the decoder may choose any of the shown BER/FER curves (or curves in between for that matter) at will by controlling the size of the EPBP pattern table maintained at the decoder. Clearly, increasing the size of the EPBP pattern table at the decoder monotonically improves the error performance of the decoder. However, in order to achieve w_{min} above a certain value, e.g., if we include EPBP patterns corresponding to input patterns with weight larger than 3, the required size of the EPBP pattern table quickly becomes unreasonable.

Table 2. Average number of additional CRC check operations performed for the bit flipping algorithm for two EPBP pattern table sizes.

	$N_f = 128$		$N_f = 1024$	
$\frac{E_b}{N_0}$ [dB]	2 EPBP patterns $w_{min} = 10$	20 EPBP patterns $w_{min} = 16$	2 EPBP patterns $w_{min} = 12$	30 EPBP patterns $w_{min} = 20$
1.0	1.98	18.99	1.94	22.68
2.0	1.89	15.4	1.79	8.86
3.0	1.56	7.24	1.70	4.42
4.0	1.31	2.87		
5.0	1.16	1.60		

Table 2 shows the average number of additional CRC check operations performed for the bit flipping algorithm at various values of SNRs for two EPBP pattern table sizes. Note that at sufficiently large SNRs, the difference in the actual number of CRC checks performed is not very sensitive to the EPBP pattern table size and is quite small as expected.

Dummy bit insertion scheme of Öberg covers the first two EPBP patterns in Table 1, Narayanan's scheme covers the first seven EPBP patterns in Table 1 with a (31,16,3) BCH code, Kim's scheme covers the first four EPBP patterns in Table 1 with a (7,4,1) BCH code. Bit flipping algorithm covers all 29 EPBP patterns in Table 1.

The comparison results of the average BER performance of various EPBP protection schemes are shown in Figs. 4-5. The performance of Narayanan [9] and Kim's [10] selective serial concatenation schemes along with the dummy bit insertion scheme of Öberg [6] are shown with those of the conventional decoder and the proposed bit flipping algorithm. The energy per bit for the selective serial concatenation and the dummy bit insertion schemes are appropriately normalized in order to take into account the parity/dummy bits that need to be transmitted. The number of dummy bit positions for Öberg's scheme and the code parameters for the two selective serial concatenation, schemes were optimized so as to minimize the BER near 10^{-6}. For the bit flipping scheme, the size of the EPBP pattern table is assumed to be 29 for N_f =128 and 35 for N_f =1024, corresponding to w_{min}=17 and w_{min}=21, respectively[1]. As expected, the proposed bit flipping algorithm outperforms all other algorithms at all SNRs.

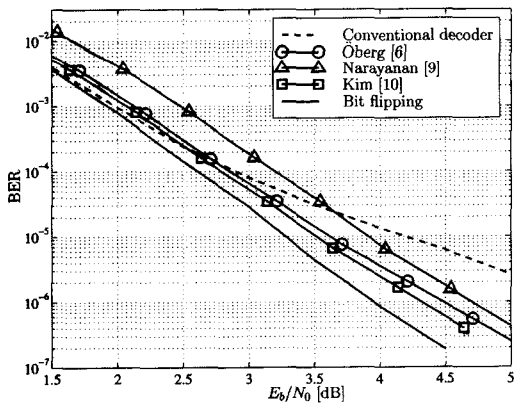


Fig. 4. BER performance of EPBP protection schemes. Frame size N_f=128. Dummy bit insertion scheme of Öberg covers the first two EPBP patterns in Table 1, Narayanan's scheme covers the first seven EPBP patterns in Table 1 with a (31,16,3) BCH code, Kim's schemes covers the first 4 EPBP patterns in Table 1 with a (7,4,1) BCH code. Bit flipping algorithm covers all 35 EPBP patterns in Table 1.

[1] Note that other than the proposed bit flipping scheme, the increase in the number of EPBPs protected does not directly imply an improvement in the resulting error performance since more parity bits or dummy bits need to be transmitted.

Ⅲ. Conclusions

In this paper, we proposed the simple protection scheme for error prone bit positions of turbo codes using the error detection capability of the CRC. The proposed scheme based on bit flipping with CRC offers flexibility on selecting the level of protection. Also, not having send additional parity bits or discarding useful bit positions, it offers the best error performance for a given level of protection.

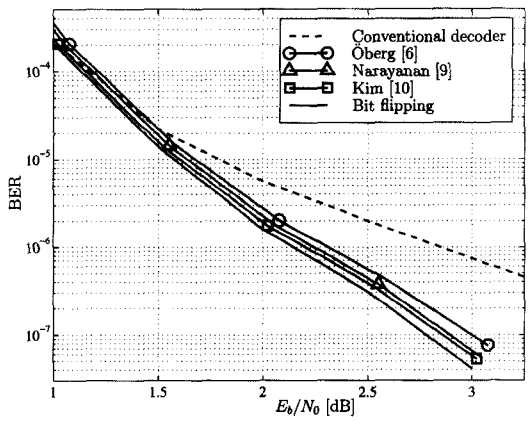


Fig. 5. BER performance of EPBP protection schemes. Frame size N_f=1024. Dummy bit insertion scheme of Öberg covers the first 9 EPBP patterns in Table 1, Narayanan and Kim's schemes both cover the first 26 EPBP patterns in Table 1 with a (63,51,3) and (31,26,1) BCH codes, respectively. Bit flipping algorithm covers all 35 EPBP patterns in Table 1.

References

[1] B. Vucetic and J. Yuan, *Turbo Codes-Principles and Applications*, Boston: Kluwer Academic Publishers, 2000.

[2] 3GPP, 3G Technical Specification, Multiplexing and Channel Coding, 25.212 v4.4.0, 2001.

[3] A. Shibutani, H. Suda and F. Adachi, "Reducing average number of turbo decoding iterations," *Electron. Lett.*, vol. 35, pp. 701-702, Apr. 1999.

[4] D. Divsalar and F. Pollara, "Turbo codes for PCS applications," in *Proc. IEEE Int. Conf. Commun.*, Jun. 1995, pp. 54-59.

[5] J. Hokfelt, O. Edfors and T. Maseng, "A survey on trellis termination alternative for turbo codes," in *Proc. IEEE Vehicular Tech. Conf.*, May 1999, pp. 2225-2229.

[6] M. Öberg and P. H. Siegel, "Lowering the error floor for turbo codes," in *Proc. 1st Int. Symp. Turbo Codes & Related Topics*, Nov. 1996, pp. 204-207.

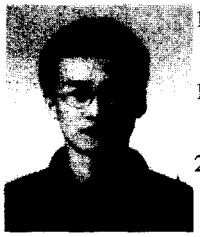
[7] J. D. Anderson, "Turbo codes extended with outer BCH code," *Electron. Lett.*, vol. 32, pp. 2059-2060, Oct. 1996.

[8] D. J. Costello and G. Meyerhans, "Concatenated turbo codes," in Proc. IEEE Int. Symp. Info. Theory and Appl., Sept., 1996, pp. 571-574.

[9] K. R. Narayanan and G. L. Stüber, "Selective serial concatenation of turbo codes," *IEEE Commun. Lett.*, vol. 1, pp. 136-140, Sept., 1997.

[10] H. Kim and P. Lee, "Performance of turbo codes with a single-error correcting BCH outer code," in Proc. IEEE Int. Symp. on Info. Theory, Jun. 2000, pp. 369.

오 왕 록(Wangrok Oh)



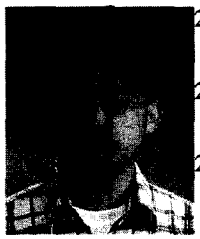
1994년 2월 : 포항공과대학교 전자전기공학과 졸업
1997년 2월 : 포항공과대학교 정보통신공학과 석사
2003년 8월 : 포항공과대학교 전자전기공학과 박사

1997년 3월~2000년 2월: 포항공과대학교 정보통신연구소 전임연구원
<관심분야> 이동통신, 확산대역통신, 터보류 부호

전 경 훈(Kyungwhoon Cheun) 정회원

1985년 2월 : 서울대학교 전자공학과 졸업
1987년 : Univ. of Michigan, Ann Arbor 전기컴퓨터공학과 석사
1989년 : Univ. of Michigan, Ann Arbor 전기컴퓨터공학과 박사
1989년 7월~1991년 7월 : Univ. of Delaware, Newark 전기공학과 조교수
1991년 7월~현재 : 포항공과대학교 전자전기공학과 조교수, 부교수, 교수
2000년 12월~2001년 12월 : Univ. of California, San Diego 방문교수
<관심분야> 통신이론, 이동통신, 확산대역통신, 터보류 부호, 시공간 부호

김 진 우(Jinwoo Kim)



2001년 2월 : 한양대학교 전자전기공학과 졸업
2003년 2월 : 포항공과대학교 전자전기공학과 석사
2003년 3월~현재: LG 전자
<관심분야> 이동통신, 터보류 부호

양 경 철(Kyeongcheol Yang) 정회원

1986년 2월 : 서울대학교 전자공학과 졸업
1988년 : 서울대학교 전자공학과 석사
1992년 12월 : Univ. of Southern California 전기공학과 박사
1993년 3월~1999년 2월 : 한양대학교 전자통신공학과 조교수
1999년 2월~현재 : 포항공과대학교 전자전기공학과 부교수
<관심분야> 부호이론, 시공간 부호, 터보류 부호, 수열설계, 정보보호