

An Efficient Adaptive Decoding for Block Turbo Codes

Xingcheng Liu^a, Wei Zhang^a, Zhongfeng Wang^b, Paul Cull^b

^a Dept of Electronics and Communications Engineering, Sun Yat-Sen University, Guangzhou, China

^b School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, USA

E-mail: isslxc@mail.sysu.edu.cn, pc@eecs.oregonstate.edu

Abstract

A novel adaptive threshold-decoding algorithm for block turbo codes (BTCs) was proposed. Simulations for a few combinations of BCH codes were carried out with the proposed algorithm and Chase algorithm for the purpose of comparison. The error correction performance with the suggested algorithm was negligibly degraded comparing with that using the Chase algorithm while it gained 1dB at BER=10⁻⁴ against Fragiaco et al.'s results. Furthermore, the suggested algorithm reduced the number of codewords to be searched and speeded up the decoding process.

Keywords: Block Turbo Codes (BTC), Adaptive Decoding, Bit Error Rate (BER), Chase Algorithm

1. INTRODUCTION

Depending on the concatenation structure, turbo codes are classified as convolutional turbo codes (CTCs) and block turbo codes (BTCs). Although most research has been focused on CTCs, BTCs have been proven to be efficient in providing good performance approaching the Shannon limit on a given AWGN channel at high code rates as Pyndiah explained^[1]. Another advantage of BTCs is that the performance of simple row/column interleaver is as good as a random one, which allows the simpler interleaving structures and hence enjoys a saving in terms of system complexity.

Similar to CTC decoding, there also is a major problem in BTC decoding that is the high complexity with the maximum a posteriori (MAP) optimal decoding algorithm^[2]. To alleviate the problem, a sub-optimal decoding structure was suggested in [1]. This decoder implements iterative decoding of BTCs using a soft-input hard-output decoder based on Chase algorithm-2 (CHS-2)^[7], followed by reliability calculations to obtain soft-decisions from the hard-output of the decoder. The performance of this decoder is sub-optimal and hence the decoding scheme presents a good tradeoff between decoding performance and computation complexity, which is very attractive for practical applications.

According to the BTC decoding mechanism, adaptive decoding could be introduced to improve the decoding performance and complexity^[3-6]. In BTC decoding with the Chase algorithm, some parameters such as α and β can be computed to adapt to the conditions of channel, codes and modulation scheme^[4-5]. In the case of using Kaneko algorithm^[12] as decoding algorithm, adaptive scaling technique can be used to estimate parameter β ^[6]. Although these modified adaptive algorithms can work adaptively and improve performance to some degrees, they increased computation overhead, which leads to more complicated decoder implementations.

In this paper, we will propose a modified Chase decoding algorithm which adapts to channel conditions and code rates. Then we present results of simulations with C Programming. This is followed by performance comparison and result analyses. Finally, the conclusions of the research are drawn.

2. BTC DECODING

In practical applications, BTCs have serial concatenation structure with two or more linear block coders separated by a simple row-column interleaver^[1,5], leading to a product code structure as shown in Fig. 1, where linear block codes (n_1, k_1, d_1) and (n_2, k_2, d_2) are employed and both denoted as $C(n, k, d)$ in brief when necessary. Therefore, the decoding procedure for BTCs is similar to that for product codes. Decoding can be performed in the order of row-wise first and then column-wise, or vice versa. For iterative decoding, the decoding process at each step needs exchange of soft reliability information.

A schematic of the Chase row or column decoder with a modified block adapted for our purpose is shown in Fig. 2, where m is the index of iteration number. Let us consider the transmission of binary elements $\{0, 1\}$ coded with a linear block code $C(n, k, d)$ on an AWGN channel using binary symbols $\{-1, +1\}$ under the symbol mapping: $0 \rightarrow -1$ and $1 \rightarrow +1$. Let C be a codeword of the linear block code $C(n, k, d)$, $X = (x_1, \dots, x_l, \dots, x_n)$ be the transmitted codeword and $R = (r_1, \dots, r_l, \dots, r_n)$ be the received data corresponding to X . Then R can be given by

$$R = X + G \quad (1)$$

This work was supported by the National Natural Science Foundation of China (No: 90304011, 60403045) and the Natural Science Foundation of Guangdong Province (No: 04009739).

where the components g_l of $G = (g_1, \dots, g_l, \dots, g_n)$ are the AWGN noise with mean zero and variance σ^2 . Assume $C^i = (c_1^i, \dots, c_l^i, \dots, c_n^i)$ is the i -th codeword of C and the squared Euclidean distance between R and C^i is defined as

$$|R - C^i|^2 = \sum_{l=1}^n (r_l - c_l^i)^2. \quad (2)$$

Then the optimum decision $D = (d_1, \dots, d_l, \dots, d_n)$ can be made according to

$$D = C^i \text{ if } |R - C^i|^2 \leq |R - C^l|^2 \quad \forall l \in [1, 2^k]. \quad (3)$$

However, searching all the codewords is increasingly prohibitive as m increases. So, the sub-optimum algorithm - Chase algorithm-2 (CHS-2)^[7] is often adopted for less complexity in practical linear block code decoding^[1, 3, 5]. This algorithm follows such procedures for decoding: Locate no more than $\lfloor d/2 \rfloor$ positions with least reliable binary elements in Y that is the hard decision of the received data R , $Y = (y_1, \dots, y_l, \dots, y_n)$ and $y_l = 0$ if $r_l < 0$; otherwise, $y_l = 1$. Then test patterns $T^p = (t_1^p, \dots, t_l^p, \dots, t_n^p)$ are formed according to [1] and [7], and accordingly, a test sequence $Z^p = (z_1^p, z_2^p, \dots, z_n^p)$, where $z_l^p = y_l + t_l^p$, is obtained. An algebraic hard decision decoder is used to decode Z^p . The result of the decoding is the codeword C^p which is then followed by adding this estimated codeword C^p to the competing codeword set Ω . Finally, the optimum decision D is made according to the above presented decision rule.

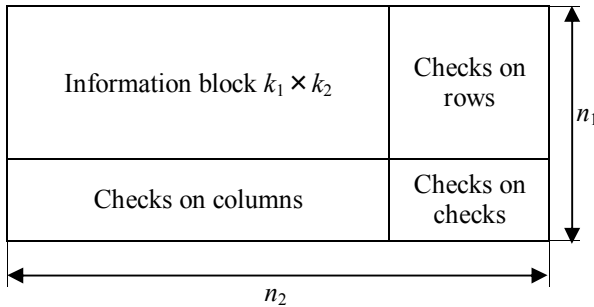


Figure 1. Code structure of block turbo codes

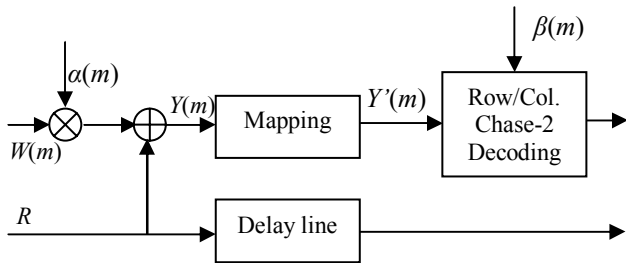


Figure 2. Adaptive-threshold Chase decoder of BTCs

The reliability of component y_j can be defined using the log-likelihood ratio (LLR) of decision y_j :

$$A(y_j) = \ln \left(\frac{\Pr\{x_j = +1 | r_j\}}{\Pr\{x_j = -1 | r_j\}} \right) = \left(\frac{2}{\sigma^2} \right) r_j. \quad (4)$$

Obviously, the LLR can be normalized to r_j since the scaling factor $2/\sigma^2$ keeps constant when a stationary channel is considered.

When decoding for rows or columns is finished, iteration for columns or rows will proceed. Therefore, we need compute the soft decision reliabilities of each component in vector D , which will feed back to the column or row decoder. Using the LLR of transmitted symbol x_j , the reliability of decision d_j in D is defined as:

$$A(d_j) = \ln \left(\frac{\Pr\{x_j = +1 | R\}}{\Pr\{x_j = -1 | R\}} \right). \quad (5)$$

If the codewords are uniformly distributed and the SNR is high, then the above equation can be approximated^[1]:

$$\begin{aligned} A(d_j) &\approx A'(d_j) = \frac{1}{2\sigma^2} \left(|R - C^{-1(j)}|^2 - |R - C^{+1(j)}|^2 \right) \\ &= \frac{2}{\sigma^2} \left(r_j + \sum_{l=1, l \neq j}^n r_l c_l^{+1(j)} p_l \right) \end{aligned} \quad (6)$$

where $C^{+1(j)}$ is a codeword in C^q such that $c_j^q = +1$ and $C^{-1(j)}$ is similarly defined with $c_j^q = -1$, and $p_l = \begin{cases} 0, & \text{if } c_l^{+1(j)} = c_l^{-1(j)} \\ 1, & \text{if } c_l^{+1(j)} \neq c_l^{-1(j)} \end{cases}$.

If the standard deviation σ is assumed constant, then the following equation^[1] is obtained after $A'(d_j)$ normalized:

$$r'_j = r_j + w_j \quad (7)$$

where r'_j is the soft output of the iteration, and

$$w_j = \sum_{l=1, l \neq j}^n r_l c_l^{+1(j)} p_l.$$

Obviously, the term w_j represents the extrinsic information that plays an important role in the iterative decoding of BTCs, as in CTCs. In practical computation of soft output, r'_j is given below^[1]:

$$r'_j = \left(\frac{|R - C|^2 - |R - D|^2}{4} \right) d_j \quad (8)$$

where C is a competing codeword of D at the minimum Euclidean distance from R with $c_j \neq d_j$.

To find the competing codeword C , the search range for Chase algorithm should be widened to increase the number of least reliable bits, p . Nevertheless, the competing codeword C is not necessarily found in some cases. The solution to this problem^[1] is to get the soft output using the following equation^[8]:

$$r'_j = \beta \times d_j \quad (9)$$

where $\beta \geq 0$ and $\beta \approx \ln \left(\frac{\Pr\{d_j = x_j\}}{\Pr\{d_j \neq x_j\}} \right)$.

Now we could get the input data (vector) to the decoder at the m -th stage of the full iteration, given by

$$Y(m) = R + \alpha(m)W(m) \quad (10)$$

where $\alpha(m)$ is the scaling vector factor^[1], and $W(m)$ is the extrinsic vector information from the previous decoder, its component is w_j (that is, it comes from the row decoder if the column decoding is being performed at the current stage of iteration). The current decoder gets newer soft-output reliability information $r_j'(m+1)$ based on $Y(m)$ and delivers at its output for the next decoder:

$$W(m+1) = r_j'(m+1) - r_j(m) \quad (11)$$

where, $r_j(m)$ is the soft-output reliability information to d_j at the m -th stage of iteration. We call every row (or column) decoding a half-iteration while each row and column decoding is referred to as a full-iteration.

3. ADAPTIVE DECODING ALGORITHM

Chase-2 algorithm heavily depends upon the number of test patterns, in which case there are $2^{\lfloor d/2 \rfloor}$ cases to be considered, where d is the minimum distance of the concerned code. The least reliable bits (the number is denoted as p) are reasonably chosen as the test patterns. However, this number p can be adjusted to adapt the code rates and channel levels so as to decrease p and shorten the time for searching the competing codewords. So we formulate an equation to accomplish the task as follows:

$$r_i = \begin{cases} +1 & \text{if } r_i > T \\ -1 & \text{if } r_i < -T, \quad i=1,2,\dots,p \\ r_i & \text{if } |r_i| \leq T \end{cases} \quad (12)$$

where $T = A_s(\gamma+1)\sqrt{(E_b/N_0)/R_c}$, E_b/N_0 is defined as the signal-to-noise ratio (SNR), $R_c = k/n$ is the code rate, A_s is a scaling factor for calibration at a specific point such as $SNR=5\text{dB}$, and γ is a reliability factor. Here we define $E_b/N_0 = a^2/(2R_c\sigma^2)$, where σ^2 is the variance on AWGN channel, and obviously, $a = 1$ for BPSK signaling using $\{0, 1\} \rightarrow \pm 1$ mapping. Comparing with Mahran et al.'s method^[14], a different mapping was adopted here, where we used an explicit value r_j instead of e of eq. (4) in [14]. Further, we could decrease the number of least reliable bits in order to speed up decoding, which was explained below.

The proposed algorithm (ACHS-2) could be described as following:

Step 1: Make hard decisions and hence get a sequence Y corresponding to a received sequence R sorted in an increasing reliability order.

Step 2: Determine the p least reliable bits in Y , make the mapping according to equation (12) and adjust the sequence Y so as to decrease the number of least reliable bits, p , to p' .

Hence the number of test patterns is also decreased. Keep p unchanged ($p'=p$) when finding nothing (i.e., $p'=0$) having least reliable bits based on equation (12).

Step 3: Generate test sequence based on the p' least reliable bits. Add (modulo-2) the test sequence and the hard-decision sequence, input the resulted sequence to an algebraic decoder to get an estimated codeword, and add it to the competing codeword set C^t .

Step 4: After finishing decoding all the combination of test and hard-decision sequences, choose a codeword in the competing codeword set C^t such that the Euclidean distance between it and the received sequence R is the minimum, and this chosen codeword is taken as the estimate output D of the received sequence.

Based upon this modified version of decoding, an additional block 'Mapping' is added to Fig. 2. So, in a half iterative decoding the previously obtained extrinsic information after normalization with factor $\alpha(m)$ is added to the received sequence R , then $2^{p'}$ test sequences are generated based upon the p' least reliable bits after mapping, and finally they are fed to the decoder. The half iterative decoding then generates newer extrinsic information and feedback to the next half-iteration. Since the number of test sequences is decreased using the proposed algorithm, the search time for the estimated codeword is reduced.

4. SIMULATION RESULTS AND ANALYSIS

The BTCs of concern are composed of BCH(15, 5, 7)², BCH(127, 106, 7)², and BCH(127, 113, 5)² respectively. The coded data are transmitted on AWGN channels with BPSK signaling. In simulation the SNRs and code rates are pre-defined and thereafter the scaling factors are changed based upon system BER performance and iteration times. The factor γ at the threshold function T is set to $\gamma = [0, 0, 0, 0, 1, 1, 1, 1]$ when 4 full-iterations are to be considered. The other parameters such as α and β are pre-defined as stated in the previous section.

In our study, the weighting factor α is pre-defined, $\alpha(m) = [0, 0.5, 0.7, 0.9, 1, 1, 1, 1]$. The reliability β is set to $\beta = m/N_{max}$, where N_{max} is the maximum number of iterations. Our experiments show that the BCH(127, 113, 5)² BTC has similar performance to Pyndiah's experiment results^[11] on AWGN channels using BPSK signaling although we take a different set of pre-defined values for α and β .

For BCH(15, 5, 7)², the number of bits in simulation is up to 2.2×10^6 . The BER performance is shown in Fig. 3 when the proposed algorithm with adaptive threshold is employed. Comparing with the results using the modified Kaneko algorithm to BCH(15, 5, 7) code^[13], the suggested algorithm gained 3dB at $BER=10^{-3}$. For BCH(127, 106, 7)², simulation results are shown in Fig. 4, where BER performance vs normalized signal-to-noise ratio is drawn using the same proposed adaptive threshold decoding algorithm. The figure shows that the proposed adaptive algorithm has a very good

BER performance. Coding gain of 6.3 dB was obtained at $BER=10^{-5}$ when 3 iterations were performed comparing with the uncoded curve.

For convenient comparison between the proposed adaptive and Chase algorithms, the performance curves are shown in Fig. 5 and Fig. 6, which reflect the performance of codes BCH(127, 113, 5)² and BCH(127, 106, 7)² on AWGN channels. At $BER=10^{-4}$, the performance using the proposed algorithm is slightly 0.1dB and 0.07dB worse than that using the Chase algorithm^[11]. On the other hand, comparing with the performance of Fragiaco et. al.'s results (see Table 3 in [9]) the proposed algorithm gains 1dB at $BER=2.5 \times 10^{-4}$ as shown in Fig. 6.

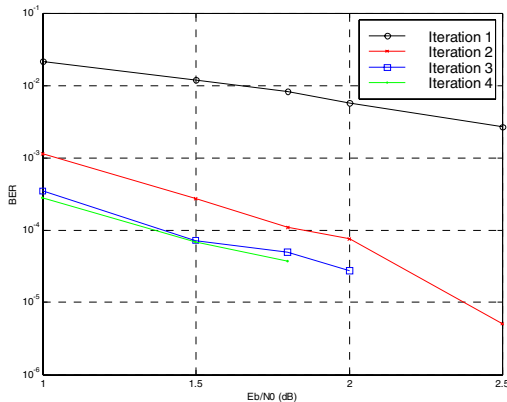


Figure 3. The BER performance of BCH (15, 5, 7)² code on AWGN channel using BPSK signaling with the proposed adaptive algorithm

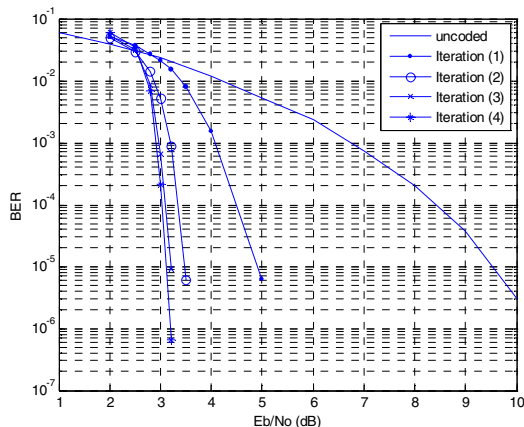


Figure 4. The BER performance of BCH (127, 106, 7)² on AWGN channel using BPSK signaling with the proposed adaptive algorithm

Furthermore, the proposed algorithm has less complexity and less number of codewords to be searched. As stated previously the half-iteration stands for a row decoding or a column decoding. Since the parameter $p' \leq p$ after the

proposed threshold mapping, the number of codewords to be searched is less than that using the normal iterative Chase decoding algorithm. Taking Chi's comparison suggestions on decoding complexity as ours that the number of hard decision decoding (HDD) is treated as a criteria of complexity^[10], a bar chart could be drawn in Fig. 7 for BCH(127, 106, 7)² BTC.

The bar to the left represents the number of HDDs needed with a straightforward implementation of the Chase algorithm. The bar to the right indicates the number of HDDs carried out when the adaptive Chase decoding is employed. A saving of more than 30% HDDs can be observed in favor of the proposed algorithm at the SNR of 3.0dB. The larger the SNR grows, the more the savings of HDDs.

5. CONCLUSIONS

In this paper, we give results on exploring an adaptive decoding algorithm for block turbo codes. The algorithm can reduce the number of codewords to be searched and lower the complexity of computation over 30% at $SNR=3.0dB$ in terms of the number of HDDs at a negligible BER performance degradation, which leads to lower power consumption at the receivers sides. The proposed method is applicable to all BCH/Hamming codes with different code rates and error-correction capability.

ACKNOWLEDGMENTS

We are deeply grateful to Professor B. Bose for his support during working at the Oregon State University.

REFERENCES

- [1] R. M. Pyndiah, "Near optimum decoding of product codes: Block turbo codes," *IEEE Trans. on Commun.*, 46(8): 1003-1010, Aug. 1998.
- [2] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. on Inform. Theory*, 20(2): 284-287, Mar. 1974.
- [3] P. A. Martin and D. P. Taylor, "On multilevel codes and iterative multistage decoding," *IEEE Trans. Commun.*, 49(11): 1916-1925, Nov. 2001.
- [4] P. A. Martin, and D. P. Taylor, "On Adaptive Reduced-Complexity Iterative Decoding", *IEEE Global Telecom. Conf. 2000 (GLOBECOM'00)*, Vol. 2: 772-776, San Francisco, CA, Nov. 2000.

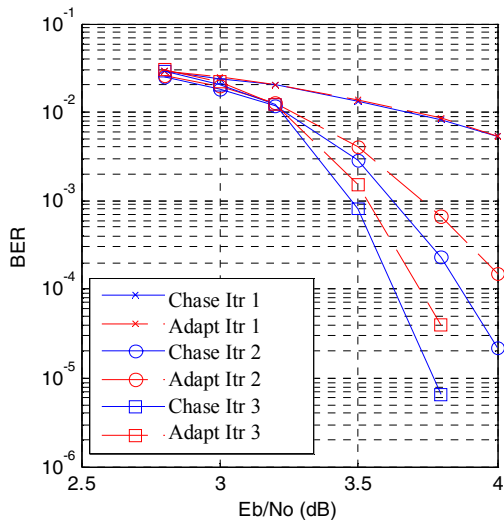


Figure 5. Comparison of BER performance between the proposed adaptive and Chase algorithms using BCH(127, 113, 5)²

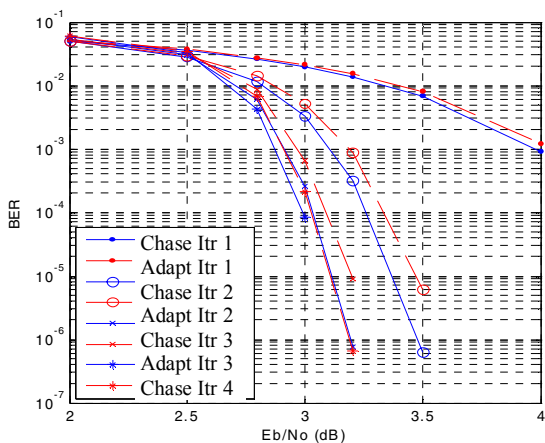


Figure 6. Comparison of BER performance between the proposed adaptive and Chase algorithms using BCH(127, 106, 7)²

[5] P. A. Martin, and D. P. Taylor, "Distance Based Adaptive Scaling in Suboptimal Iterative Decoding", *IEEE Trans. on Commun.*, 50(6): 869-871, June 2002.

[6] S. Dave, J. Kim, and S. C. Kwatra, "An Efficient Decoding Algorithm for Block Turbo Codes", *IEEE Trans. on Commun.*, 49(1): 41-46, Jan. 2001.

[7] D. Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. on Inform. Theory*, IT-18(1): 170-182, Jan. 1972.

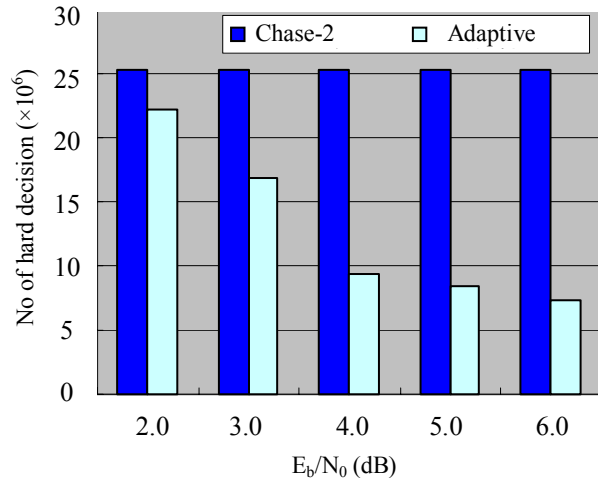


Figure 7. Comparison of the number of HDDs carried out for BCH(127, 106, 7) BTC under different decoding schemes after 4 full-iterations

[8] R. Pyndiah, "Iterative decoding of product codes: Block turbo codes," in *Proc. IEEE Int. Symp. on Turbo Codes & Related Topics*, pp. 71-79, Brest, France, Sept. 1997.

[9] S. Fragiaco, C. Matrakidis and J. O'Reilly, "Novel near maximum likelihood soft decision decoding algorithm for linear block codes", *IEE Proc.-Commun.*, 146(5): 265-270, Oct. 1999.

[10] Z. Chi, L. Song, and K.K. Parhi, "On the performance/complexity tradeoff in block Turbo decoder design", *IEEE Trans. on Communications*, 52(2): 173-175, Feb. 2004.

[11] R. Pyndiah, A. Glavieux, A. Picart, and S. Jacq, "Near optimum decoding of product codes", in: *Proc. IEEE Global Telecom. Conf. 1994 (GLOBECOM'94)*, Vol. 1: 339-343, San Francisco, CA, Nov. 28-Dec. 2, 1994.

[12] T. Kaneko, T. Nishijima, H. Inazumi, S. Hirasawa, "An efficient maximum-likelihood-decoding algorithm for linear block codes with algebraic decoder", *IEEE Trans. on Inform. Theory*, 40(2): 320-327, Mar. 1994.

[13] S. M. Elengical, F. Takawira, and H. Xu, "Reduced complexity of maximum-likelihood decoding", in: *Proc. 7th IEEE AFRICON Conf. in Africa (AFRICON 2004)*, Vol. 1: 217-220.

[14] A. Mahran and M. Benaissa, "Adaptive Chase algorithm for block turbo codes," *Electronics Letters*, 39(7): 617-619, April 2003.