Journal of the Research Institute of Industrial Technology, Vol. 24, 2007

Parallel Turbo Product Code for Satellite Communications

*Ji-Won Jung**

* *Department of Radio Sciences and Engineering, Korea Maritime University, Busan 606-791, Korea*

ABSTRACT : This paper present a parallel algorithm of turbo product codes enable simultaneous decoding of row and column optimal for satellite communications. The row and column decoders operate in parallel and update each other after row and column has been decoded. Simulation results show that the performance of proposed parallel turbo product code is almost the same as that conventional scheme for several turbo product codes

KEY WORDS : turbo product codes, satellite communications, parallel turbo product code

1. Introduction

In 1993, Berrou introduced a new class of error correcting codes for digital transmission : the "turbo code"[1]. The important problems of high-speed applications of turbo decoder are decoding delay and computational complexity. Therefore, the real difficulty in the field of channel coding is essentially a problem of decoding complexity of powerful codes in satellite communications. Digital transmission via Ka-band satellite can be severely affected by rain-induced signal fade. Rain-fade countermeasure has been one important design objective of any Ka-band satellite communication systems, especially those offering broadband multimedia services. Furthermore satellite communication requires high-speed data rate and low decoding complexity while maintain performance. Therefore there has been intensive focus on turbo product code(TPC) which has low latency and simple structures compare with turbo code[2]. It can achieve performances near Shannon limit. TPCs are two dimensional code constructed from small component codes. Different than original TPC decoder, which performs row and column decoding in a serial fashion, In this paper, we propose a parallel decoder structure to reduce the latency. Furthermore, only one delay element is needed in contrast to two delay elements, i.e., decoding time is halved with this structure while maintaining

* jwjung@hhu.ac.kr 051)410-4424

the same performance level.

2. Turbo Product Codes(TPCs)

As shown is Fig. 1, a two-dimensional product code is obtained as follows. The information symbols are initially arranged in a $k_1 \times k_2$ array. Then, the columns are encoded using a linear block code $C_1(n_1, k_1, \delta_1)$. Afterwards, the resulting n_1 rows are encoded using a linear block code $C_2(n_2, k_2, \delta_2)$ and the product code, which consists of n_1 rows and n_2 columns, is obtained. The parameters of code $C_i(i=1, 2)$, denoted as n_i, k_i, and δ_i, are the codeword length, number of information symbols, and minimum Hamming distance, respectively. Codes C_1 and C_2 are called the constituent (or component) codes. To decrease implementation complexity of both the encoder and the decoder, usually $C_1 = C_2$.

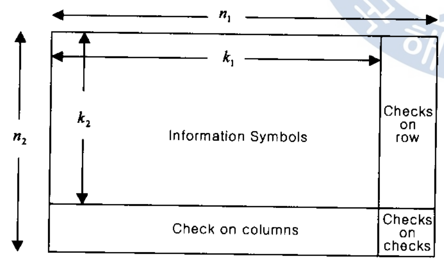


Fig.1 **Encoder construction of turbo product code ($P = C_1 \times C_2$)**

Considering all decoding algorithms for product codes, the TPC decoding method by Pyndiah[2] enabled high performance at low complexity. If we consider the transmission of block coded binary symbols {-1,+1} using a QPSK modulation over a Gaussian channel, the sample vector $R=(r_1, \ldots, r_l, \ldots, r_n)$ at the output of the coherent demodulator, conditionally to a transmitted code word $E=(e_1, \ldots, e_l, \ldots, e_n)$, can be modelized by :

$$R = E + N \tag{1}$$

where $N=(n_1, \ldots, n_l, \ldots, n_n)$ are Additive White Gaussian Noise(AWGN) samples of standard deviation σ. From the received samples R, the decoded optimum sequence is given by :

$$D = C^i \text{ if } \Pr\{E = C^i | R\} > \Pr\{E = C^j | R\} \ \forall j \neq i \tag{2}$$

where $C^i = (C_1^i, C_2^i, \ldots, C_l^i, \ldots, C_n^i)$ is the i^th code word of code C with parameters (n, k, δ) and $D=(d_1, \ldots, d_l, \ldots, d_n)$ is the decision corresponding to maximum likelihood transmitted sequence conditionally to R For received samples corrupted by AWGN, Eq. (2) is simplified into :

$$D = C^i \text{ if } |R - C^i|^2 > |R - C^j|^2 \quad \forall \ j \neq i \tag{3}$$

$$|R - C^i|^2 = \sum_{l=1}^{n} (r_l - c_l^i)^2 \tag{4}$$

For block codes with high code rate R, the number of code works 2^k is relatively large and optimum sequence decoding is too complex for implementation. In 1972, Chase proposed an algorithm[3] which approximates optimum sequence decoding of block codes with low computation complexity and a small performance degradation. Instead of reviewing all the code words, the Chase algorithm searches for the codes words at

Hamming distance within a sphere of radius $(\delta_i - 1)$, i =1, 2 centered on $Y=(y_1, \ldots, y_l, \ldots, y_n)$, Y_i = 0, 5 (1+sgn(r_i)) where y_l is equal to 1 if $r_l > 0$ and to 0 if $r_l \leq 0$. To further reduce the number of reviewed code words, only the most probable code words within the sphere are selected by using channel information R. This search procedure can be decomposed in there steps :

Step1: determine the position of the P least reliable binary symbols of Y using R.

Step2 : form test sequence T^q defined as all the combination of sequences with at least a "1" in the P least reliable position. $q = 2^p$ define.

Step3 : decode Y and $Z^q = Y \oplus T^q$ using an algebraic decoder and memorize the code words C^q.

Decision D is then given by Eq.(2) with reviewed code words restricted to the subset of code words at step 3 above. Coming back to the decoding of a product code, the chase algorithm yields for each row(or column) the decision D of the component block code for given input data R. In order to generate soft decision at the output of decoder, we have to calculate the reliability of D. The reliability of decision d_j is defined as Eq.(5).

$$(LLR)_j = \ln\left(\frac{\Pr(e_j = +1/R)}{\Pr(e_j = -1/R)}\right) \tag{5}$$

By considering the different authorized code words, the Eq.(5) can be expressed as :

$$\Pr(e_j = +1/R) = \sum_{i \in S_j^{+1}} \Pr(E = C^i/R)$$
$$\Pr(e_j = -1/R) = \sum_{i \in S_j^{-1}} \Pr(E = C^i/R) \tag{6}$$

where S_j^{+1}(or S_j^{-1})is the set containing the index of code words such that C_j^i=+1(or C_j^i=-1). By applying BAYES rule to Eq.(6) and (7), the assumption that the different code words are uniformly distributed, we obtain for $(LLR)_j$ the following Equation :

$$(LLR)_j = \ln\left(\frac{\sum_{i \in S_j^{+1}} \exp\left(-\frac{|R - C^i|^2}{2\delta^2}\right)}{\sum_{i \in S_j^{-1}} \exp\left(-\frac{|R - C^i|^2}{2\delta^2}\right)}\right) \tag{7}$$

Eq.(7) can be expressed as Eq.(8) after some manipulations. The extrinsic information W_j for the j^{th} bit position is found by

$$w_j = \frac{(LLR)_j - r_j}{\beta d_j} \tag{8}$$

The term W_j is a correction term applied to the input data and it plays the same role as the extrinsic information in Convolutional Turbo Code. The extrinsic information, W_j, is a random variable with a Gaussian distribution since it is a linear combination of identically distributed random variables.

Where β is a reliability factor to estimate W_j in case no competing codeword can be found. Once the extrinsic information has been determined, the input to the next decoding stage is updates as

$$r'_j = r_j + \alpha w_j \quad (9)$$

The normalized LLR r'_j is taken as the soft output of the decoder. Where α is a weight factor to combat high standard deviation in W_j and high BER during the first iterations. The operations above are performed on all bits of a product codeword, shown in Fig 2., hence, Eq.(9) can be expressed in matrix form as

$$R[2] = [R] + \alpha[2] \cdot [W[2]] \quad (10)$$

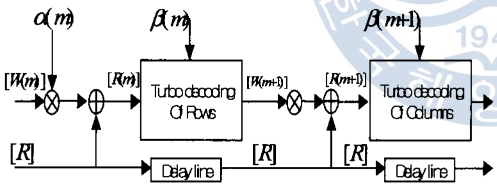


Fig.2 TPC decoder structure that Pyndiah proposes

For example, if we consider (63. 57. 3) BCH code, the performance of TPC (63. 57. 3) ⊗ (63. 57. 3) is shown in Fig.3. We observe that for additional iteration we obtain a reduction of BER. However, for BER of 10^{-5}, the reduction in terms of Eb/No becomes negligible for additional iterations beyond iteration 4.

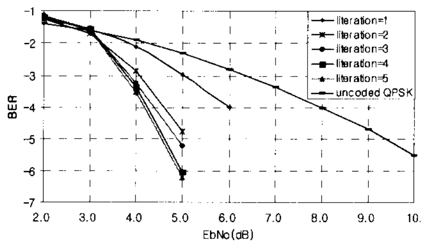


Fig.3 Performance of TPC (63. 57. 3) ⊗ (63. 57. 3) at each iteration number

3. Proposed Parallel Turbo Product Code Decoder

For the decoder shown in Fig.2, the conventional TPC decoder needs to decode row or column before the next half-iteration can begin. To reduce decoding latency, we proposed parallel TPC decoder in shown Fig.4. The row and column decoders operate in parallel and update each other immediately after a row or column has been decoded. Different than in the conventional serial TPC decoder, decoding time of the proposed algorithm is halved. The matrices $[W^{row}]$ and $[W^{col}]$which are the row and column extrinsic information matrices are transferred to row or column decoder by block basis simultaneously.

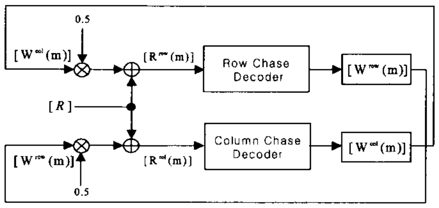


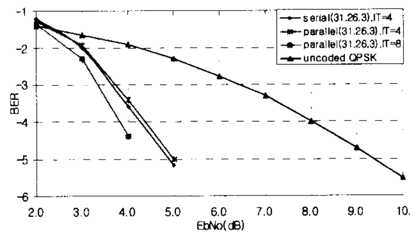
Fig.4 Parallel decoder structure and decoder process

For the first iteration at m=0, we set $[R^{row}(0)] = [R^{col}(0)] = [R]$. Let us assume the same code of block length n is used as the row and column code of the product code. As soon as the row and column decoders have finished decoding of all rows and columns, respectively, they pass their updated matrices $[R^{row}(m+1)]$and $[R^{col}(m+1)]$ as inputs to the next decoding stage. As a result of parallel updating, we have relation Eq. (11).

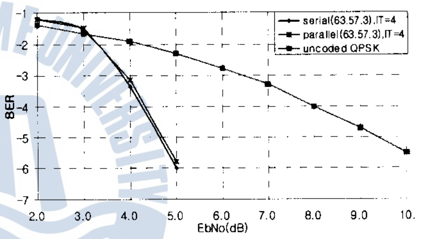
$$[R^{row}(m+1)] = [R] + \alpha(m) \cdot [W^{col}(m)]$$
$$[R^{col}(m+1)] = [R] + \alpha(m) \cdot [W^{row}(m)] \quad (11)$$

4. Simulation Results

The results that we present here concern BCH product code with several code rates. Especially, TPCs with one-error correcting BCH component codes are suitable for applications which require both high data and code rates. Serial (n, k, δ) is TPC decoder which is used to serially two BCH decoder at row and column. Parallel(n, k, δ) is TPC decoder which is used to parallel two BCH decoder at row and column. For chase decoding, the number of least reliable bits was chosen to be p=4. To iterate decoder, the weight and reliability factor were fixed for each iteration as α =0.5 and β =1. With these settings, our simulation results for two TPCs ($(31.\ 26.\ 3)^2$, $(63.\ 57.\ 3)^2$), with four decoding iterations are given in Fig.5. We observe that the performance parallel TPC decoder is similar to that of conventional TPC decoder at BER = 10^{-4}. In Fig.5. we also included the performance of a parallel (31. 26. 3) with eight iterations. This TPC has the same decoding latency as the conventional decoding approach with four iterations, but achieves an additional gain of 0.4dB at BER = 10^{-4}.



(a) TPC $(31.\ 26.\ 3)^2$



(b) TPC $(63.\ 57.\ 3)^2$

Fig.5 Performance of original and parallel decoded BCH TPCs on AWGN channel (IT denotes number of iteration)

5. Conclusion

In this paper, we present a parallel decoder of TPC based on parallel concatenation to reduce the decoding latency. In our simulation one-bit error BCH component codes are used. Simulation results show the performance of proposed scheme almost the same as conventional one. Therefore, The proposed algorithm is suitable for application which require both high data and code rate.

References

[1] C. Berrou, A. Glavieux, and P.Thitimajshima, "Near Shannon Limit Error-Correcting Code and Decoding : Turbo Codes", in *Proc. Of ICC'93, 1993.*

[2] R.M. Pyndiah, "Near-optimum decoding of product codes: Block turbo codes," *IEEE Trans. Commun., vol. 46, pp1003-1010,Aug.1998.*

[3] D.Chase, "A class of algorithms for decoding block codes with channel measurement information," *IEEE Trans.Inform. Theory, vol. IT-18,pp.170-182,Jan.1972*

[4] Cenk Argon, "A parallel decoder for low latency decoding of turbo product codes," *IEEE commun, LETTER, vol. 6, No 2, Feb, 2002.*

Received : 4 January 2007
Accepted : 19 January 2007