ISSN: 2250–3676

[IJESAT] INTERNATIONAL JOURNAL OF ENGINEERING SCIENCE & ADVANCED TECHNOLOGY Volume-2, Issue-4, 1138 – 1143

CONVOLUTIONAL APPROACH FOR DECODING FEC IN LTE AND WiMAX SYSTEMS THROUGH UBP ALGORITHM

A.Madhu[1], C.Prakash Rao[2], S.Srinivasulu[3]

[1]*Student (M.Tech), Dept. of CSE, Prakasam Engineering College, AP, India,* ***madhuattanti@gmail.com***
[2]*Associate Professor, Dept. of CSE, Prakasam Engineering College, AP, India,* ***bhanu_0211@yahoo.co.in***
[3]*Head of the Department, Dept. of CSE, Prakasam Engineering College, AP, India,* ***sreenivasulusadineni@gmail.com***

Abstract

Forward Error correcting Codes (FEC) is a technique used for controlling errors in data transmission over unreliable or noisy communication channels. This scheme is overhead in many wireless communication systems, even though they have enhanced data rates for the GSM evolution (EDGE), because, worldwide interoperability for microwave access (WiMAX) and long term evolution (LTE) have adopted low-density parity-check (LDPC). There are many efficient algorithms for decoding these codes. However, the different decoding approaches for these two families of codes usually lead to different hardware architectures. The belief propagation BP algorithm provides a highly effective general methodology for devising low-complexity iterative decoding algorithms for all convolutional code classes as well as turbo codes. In this paper, we exploit the parity-check matrix (H) representation of tailbiting convolutional and turbo codes, enabling decoding through a unified belief propagation (UBP) algorithm. This *unified propagation and scaling* algorithm reduces to a convergent alternative to loopy belief propagation when no constraints are present. *While a small performance loss is observed when decoding turbo codes with BP, this is offset by the lower complexity of the BP algorithm and the inherent advantage of a unified decoding architecture.*

Index Terms: *Forward Error correcting Code, unified belief propagation, parity-check matrix, decoding*

---------- *** ----------

1. INTRODUCTION

Forward Error Correction (FEC) is a well-known technique for improving the reliability of packet transmission over networks that do not provide guaranteed packet delivery, especially in multicast and broadcast applications.

The two main categories of FEC codes are block codes and convolutional codes as follows.

- Block codes work on fixed-size blocks (packets) of bits or symbols of predetermined size. Practical block codes can generally be decoded in polynomial time to their block length.
- Convolutional codes work on bit or symbol streams of arbitrary length. They are most often decoded with the Viterbi algorithm, though other algorithms are sometimes used. Viterbi decoding allows asymptotically optimal decoding efficiency with increasing constraint length of the convolutional code, but at the expense of exponentially increasing complexity. A convolutional code can be turned into a block code, if desired, by "tail-biting".

Note that the term "Forward Erasure Correction" is sometimes used, erasures being a type of error in which data is lost and this loss can be detected, rather than being received in corrupted form. The focus of this document is strictly on erasures, and the term "Forward Error Correction" is more widely used.

The most common known decoding algorithms for convolutional codes were implemented based on either algebraically calculating the error pattern or on trellis graphical representations such as in the MAP and Viterbi algorithms till recently. A decoding principle has appeared: iterative decoding, with the advent of turbo coding. Iterative decoding is a general framework based on bipartite graphs for the description of LDPC codes and their decoding through the belief propagation (BP) algorithm.

In most of the cases, convolutional codes are just like as block codes. For example, a block code whose code words correspond to all trellis paths to the truncation depth is created if we truncate the trellis by which a convolutional code is represented. But, the problem can be raised in error performance by this truncation, because the last bits lack error protection. The onventional solution to this problem is to encode a fixed number of message blocks L followed by m additional all-zero blocks, where m is the constraint length of the convolutional code. This method provides uniform error protection for all information digits, but causes a rate reduction for the block code as compared to the convolutional code by the multiplicative factor L/(L+m). In the tail-biting convolutional code, zero-tail bits are not needed and replaced by payload bits resulting in no rate loss due to the tails. Therefore, the spectral efficiency of the channel code is improved.

1.1 Convolutional encoding

To convolutionally encode data, start with *k* memory registers, each holding 1 input bit. Unless otherwise specified, all memory registers start with a value of 0. The encoder has *n* modulo-2 adders (a modulo 2 adder can be implemented with a single Boolean XOR gate, where the logic is: 0+0 = 0, 0+1 = 1, 1+0 = 1, 1+1 = 0), and *n* generator polynomials — one for each adder (see figure below). An input bit m_1 is fed into the leftmost register. Using the generator polynomials and the existing values in the remaining registers, the encoder outputs *n* bits. Now bit shift all register values to the right (m_1 moves to m_0, m_0 moves to m_{-1}) and wait for the next input bit. If there are no remaining input bits, the encoder continues output until all registers have returned to the zero state.

The figure below is a rate 1/3 (*m*/*n*) encoder with constraint length (*k*) of 3. Generator polynomials are G_1 = (1,1,1), G_2 = (0,1,1), and G_3 = (1,0,1). Therefore, output bits are calculated (modulo 2) as follows:

$$n_1 = m_1 + m_0 + m_{-1}$$

$$n_2 = m_0 + m_{-1}$$

$$n_3 = m_1 + m_{-1}.$$

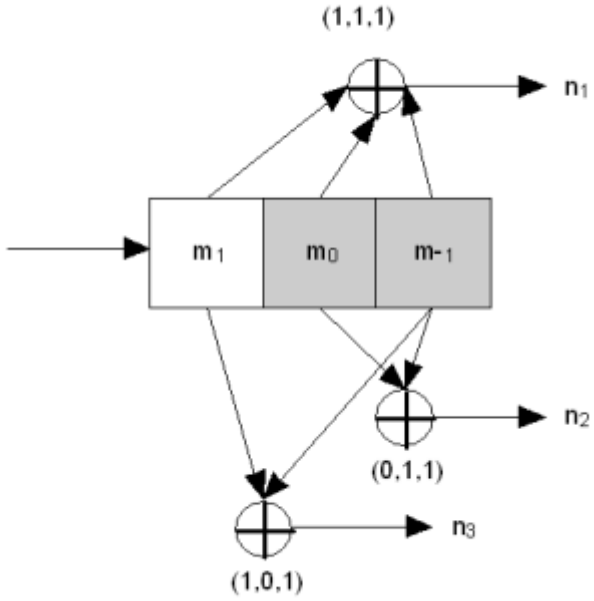

Fig.1. Rate 1/3 non-recursive, non-systematic convolutional encoder with constraint length 3

1.2 Recursive and non-recursive codes

The encoder on the picture above is a *non-recursive* encoder. Here's an example of a recursive one:

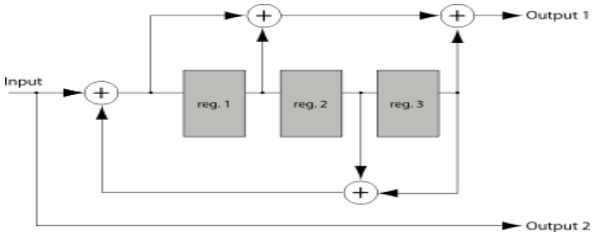

Fig.2. Rate 1/2 recursive, systematic convolutional encoder with constraint length 4

One can see that the input being encoded is included in the output sequence too (look at the output 2). Such codes are referred to as *systematic*; otherwise the code is called *non-systematic*.

Recursive codes are almost always systematic and, conversely, non-recursive codes are non-systematic. It isn't a strict requirement, but a common practice.

1.3. Analysis of turbo and LPDC

Both turbo and LDPC codes have been extensively studied for more than fifteen years. However, the formal relationship between these two classes of codes remained unclear until Mackay in [8] claimed that turbo codes are LDPC codes. Also, Wiberg in [9] marked another attempt to relate these two classes of codes together by developing a unified factor graph representation for these two families of codes. In [10], McEliece showed that their decoding algorithms fall into the same category as BP on the Bayesian belief network. Finally, Colavolpe [11] was able to demonstrate the use of the BP algorithm to decode convolutional and turbo codes. The operation in [11] is limited to specific classes of convolutional codes, such as convolutional self orthogonal codes (CSOCs). Also, the turbo codes therein are based on the serial structure while the parallel structure is more prevalent in practical applications.

In LTE and WiMAX systems, the proposed decoders for the tail-biting convolutional codes and turbo codes are based on the Viterbi and MAP algorithms, respectively. However, many other efficient algorithms have been proposed to decode tail-biting convolutional codes as well as turbo codes. For example, in [3], the reduced complexity wrap-around Viterbi algorithm was proposed to decode tail-biting convolutional codes in the WiMAX system to reduce the average number of decoding iterations and memory usage. In addition, other decoding algorithms such as double traceback and bidirectional Viterbi algorithms were also proposed for tail-biting convolutional codes in LTE [4]. Finally in [5], the design and optimization of low-complexity high performance rate-matching algorithms based on circular buffers for LTE turbo codes was investigated.

2. Decoding of Convolutional Coding

In this paper, we focus on the direct application of the BP algorithm used for LDPC codes to decode the tail-biting convolutional codes and turbo codes in WiMAX and LTE systems, respectively. Based on that, we propose a decoder with drastically lower implementation complexity than that proposed in the latest releases for these systems [5-7].

A convolutional code is called tail-biting when its codewords are those code sequences associated with paths in the trellis that start from a state equal to the last m bits of an information vector of k data bits. Many efficient algorithms have been proposed for decoding tail-biting convolutional codes such as the Viterbi and MAP algorithms. As shown below, we represent the tail-biting convolutional code by its generator and parity-check matrices in order to apply the BP algorithm directly.

2.1. Parity-check matrix of tail-biting convolutional codes

It is required to obtain generator matrix G and parity matrix H to be able to represent a tail-biting convolutional code by a Tanner graph and then apply the BP algorithm to its decoding.

The matrix representation by a simplified example is as follows:

Example 1: Consider the convolutional code with rate R = k/n = 1/2 , where k represents the number of input bits and n the output bits. Assume that the information sequence is **x** = (x0, x1, x2, ...). The encoder will convert this to the sequences

$\mathbf{y}^{(0)} = (y^{(0)}_0 , y^{(0)}_1 , y^{(0)}_2 , ...)$ and
$\mathbf{y}^{(1)} = (y^{(1)}_0 , y^{(1)}_1 , y^{(1)}_2 , ...)$.

Note that if there are multiple input streams, we can refer to a single interleaved input $\mathbf{x} = (x^{(0)}_0 , x^{(1)}_1 , ...)$. Also, the output streams are multiplexed to create a single coded data stream $\mathbf{y}= (y^{(0)}_0 \; y^{(1)}_0 , \; y^{(0)}_1 \; y^{(1)}_1 , \; ...)$ where **y** is the convolutional codeword. In addition, each element in the interleaved output stream **y** is a linear combination of the elements in the input stream $\mathbf{x} = (x^{(0)}_0 , x^{(1)}_0 , ..., x^{(k-1)}_0 , x^{(0)}_1 , x^{(1)}_1 , ..., x^{(k-1)}_1 , ...)$.

An impulse response $g^{(j)}$ is obtained from the encoder output by applying a single 1 at the input followed by a string of zeros, then strings of zeros are applied to all the other inputs (in the case of multiple inputs). The impulse responses for the encoder in our example are
$g^{(0)} = (1011)$,
$g^{(1)} = (1101)$.

The impulse responses are often referred to as generator sequences, because their relationship to the codewords generated by the corresponding convolutional encoder is similar to that between generator polynomials and codewords in a cyclic code. The generator sequences can be expressed in the following general form:

$$y^{(j)}{}_t = \sum_{l=0}^{m-1} x_{t-1} \; g_l^{(j)} . \qquad (1)$$

Each coded output sequence **y**(j) in a rate 1/n code is the convolution of the input sequence **x** and the impulse response $g^{(j)}$,

$$y^{(j)} = x * g^{(j)}. \qquad (2)$$

In vector form, this is expressed

$$y^{(j)} = \sum_{i=0}^{k-1} x^{(i)} * g_i^{(j)} \qquad (3)$$

which can be developed thus

$$y_t^{(j)} = \sum_{i=0}^{k-1}\left[\sum_{l=0}^{m_i-1} x_{t-1}^{(i)}\, g_{i,l}^{(j)}\right] \qquad (4)$$

We can express these forms as a matrix multiplication operation, thus providing a generator matrix similar to that developed for block codes. In fact, the primary difference arises from the fact that the input sequence is not necessarily bounded in length, and thus the generator and parity check matrices for convolutional codes are semi infinite. However, herein we introduce the **G** and **H** matrices as equivalent to a tail-biting convolutional code having finite length. Therefore, the generator and parity-check matrices will be as follows :

$$G = \begin{bmatrix} G_m & \cdots & & G_0 & G_1 & \cdots & G_{m-1} \\ G_{m-1} & G_{m-2} & & G_0 & & G_{m-2} & \\ G_{m-2} & G_{m-1} & \cdots & & & \cdots\cdots & \\ & \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots & & & & & \\ & \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots & & & & & \\ G_0 & G_1 & \cdots & G_{m-1} & G_m & \cdots & \cdots \\ & G_0 & G_1 & & G_m & & \\ & & \cdots\cdots\cdots\cdots\cdots & & & & \\ & & \cdots\cdots\cdots\cdots\cdots & & & & \\ & & & G_0 & G_1 & \cdots & G_m \end{bmatrix} \qquad (5)$$

where

$$G_l = \begin{bmatrix} g_{1,l}^{(1)} & g_{1,l}^{(2)} & \cdots & \cdots & g_{1,l}^{(n)} \\ g_{2,l}^{(1)} & g_{2,l}^{(2)} & \cdots & \cdots & g_{2,l}^{(n)} \\ & & \cdots & & \\ & & \cdots & & \\ g_{k,l}^{(1)} & g_{k,l}^{(2)} & \cdots & \cdots & g_{k,l}^{(n)} \end{bmatrix} \qquad (6)$$

Note that each block of k rows in the **G** matrix is a circular shift by n positions of the previous such block. In general, the parity-check matrix of a rate k/n tail-biting convolutional code with constraint length m is

$$m = \begin{bmatrix} P_0^T|I & \cdots & P_m^T|O & P_{m-1}^T|O & \cdots & P_1^T|O \\ P_1^T|I & \cdots & P_{m-1}^T|O & P_m^T|O & \cdots & P_2^T|O \\ & & \ddots & & & \\ \cdots\cdots & P_1^T|I & P_0^T|I & & \cdots & P_m^T|O \\ P_m^T|O & \cdots & P_1^T|O & P_0^T|O & \cdots & \\ & & \ddots & & & \\ & \cdots & \cdots & & \cdots\cdots & \\ P_m^T|O & P_{m-1}^T|O & \cdots & \cdots & P_1^T|O & P_1^T|I \end{bmatrix} \qquad (7)$$

where **I** is the k × k identity matrix, **0** is the k × k all zero matrix, and P_i, i = 0, 1, ...,m, is a k ×(n−k) matrix whose entries are

$$P_t = \begin{bmatrix} g_{1,t}^{(k+1)} & g_{1,t}^{(k+2)} & \cdots & g_{1,t}^{(n)} \\ g_{2,t}^{(k+1)} & g_{2,t}^{(k+2)} & \cdots & g_{2,t}^{(n)} \\ & \vdots & & \\ g_{k,t}^{(k+1)} & g_{k,t}^{(k+2)} & \cdots & g_{k,t}^{(n)} \end{bmatrix} \qquad (8)$$

Here, $g_{p,i}$ is equal to 1 or 0 coresponding to whether or not the i^{th} stage of the shift register for the input contributes to output j(i = 0, 1, ...,m; j = (k + 1), (k + 2), ..., n; p = 1, 2, ..., k). Since the last m bits serve as the starting state and are also fed into the encoder, there is an end-round-shift phenomenon for the last m columns of **H**.

Example 2: Consider the previous encoder shown in Example 1, assume that a block of k = 6 information bits are encoded. Then the tail-biting construction gives a binary (12, 6) code with generator and parity check matrices,

$$G = \begin{bmatrix} 11\ 01\ 10\ 11\ \ 00\ 00 \\ 00\ 11\ 01\ 10\ 11\ 00 \\ 00\ 00\ 11\ 01\ 10\ 11 \\ 11\ 00\ 00\ 11\ 01\ 10 \\ 10\ 11\ 00\ 00\ 11\ 01 \\ 01\ 10\ 11\ 00\ 00\ 11 \end{bmatrix} \qquad (9)$$

and

$$H = \begin{bmatrix} 11\ 00\ 00\ 11\ 01\ 10 \\ 10\ 11\ 00\ 00\ 11\ 01 \\ 01\ 10\ 11\ 00\ 00\ 11 \\ 11\ 01\ 10\ 11\ 00\ 00 \\ 00\ 11\ 01\ 10\ 11\ 00 \\ 00\ 00\ 11\ 01\ 10\ 11 \end{bmatrix} \qquad (10)$$

B. Degree distribution of Tanner graph for tail-biting convolutional codes

Looking at the **H** matrix of a tail-biting convolutional code, we can notice that it is similar to the **H** matrix of an irregular LDPC code where the number of non-zero elements is not a fixed number per row and column. Our goal is to represent the tail-biting convolutional codes through Tanner graphs in order to decode them using the BP algorithm. Therefore, it is important to obtain the degree distribution of the Tanner graph which describes the number of edges into the bit and check nodes in irregular LDPC codes. The fraction of edges which are connected to degree-i bit nodes is denoted _i, and the fraction of edges which are connected to degree-i check nodes is denoted _i. The functions

$$F(x) = f_1x^1 + f_2x^2 + \ldots\ldots + f_tx^t + \ldots\ldots \quad (11)$$

$$G(x) = g_1x^1 + g_2x^2 + \ldots\ldots + g_tx^t + \ldots\ldots \quad (12)$$

are defined to describe the degree distributions

3. WIMAX AND LTE CODING STRUCTURE

To address the low and high rate requirements of LTE, the 3rd Generation Partnership Project (3GPP) working group undertook a rigorous study of advanced channel coding candidates such as tail-biting convolutional and turbo codes for low and high data rates, respectively. We investigate here the application of the BP decoder for the proposed turbo code in LTE systems. Meanwhile, a rate ½, memory-6 tail-biting convolutional code has been adopted in the WiMAX (802.16*e*) system, because of its best minimum distance and the smallest number of minimum weight codewords for larger than 32-bit payloads which is used for both frame control header (FCH) and data channels. In fact, we will focus here on the FCH which has much shorter payload sizes (12 and 24 bits) as shown in the next subsection.

3.1. Tail-biting convolutional code in 802.16e

Here, we briefly describe the WiMAX frame control header structure. In the WiMAX Orthogonal Frequency Divisio Multiplexing (OFDM) physical layer, the payload size of the frame control header is either 24 bits or 12 bits and the smallest unit for generic data packet transmission is one subchannel. A subchannel consists of 48 QPSK symbols (96 coded bits). At a code rate of ½, one subchannel translates to 48 bits as the smallest information block size. Currentl , the FCH payload bits are repeated to meet the minimum number (48) of encoder information bits. The generator polynomials for the rate½ WiMAX tail-biting convolutional code are given by g1 = (1011011) and g2 = (1111001) in binary notation. These generator polynomials have the best dmin(minimum distance) and ndmin(number of codewords with weight dmin) for payload sizes _ 33 bits and for some payload sizes between 25 and 33 bits, under the constraint of memory size m = 6 and code rate ½.

3.2. Turbo code in LTE system

The 3GPP turbo code is a systematic parallel concatenated convolutional code (PCCC) with two 8-state constituent encoders and one turbo code internal interleaver. Each constituent encoder is independently terminated by tail bits. For an input block size of K bits, the output of a turbo encoder consists of three length K streams, corresponding to the systematic and two parity bit streams (referred to as the "Systematic", "Parity 1", and "Parity 2" streams in the following), respectively, as well as 12 tail bits due to trellis termination. Thus, the actual mother code rate is slightly lower than 1/3. In LTE, the tail bits are multiplexed to the end of the three streams, whose lengths are hence increased to (K + 4) bits each [5]. The transfer function of the 8-state constituent code for the PCCC is:

$$G(D) = \left[1, \frac{g_1(D)}{g_0(D)}\right],$$

$$g_0(D) = 1 + D^2 + D^3,$$

$$g_1(D) = 1 + D + D^3$$

The initial value of the shift registers of the 8-state constituent encoders will be all zeros when starting to encode the input bits. The output from the turbo encoder is $d_k^{(0)} = x_k$, $d_k^{(1)} = z_k$, and $d_k^{(2)} = z_k'$ for k = 0, 1, 2, . . . ,K−1. If the code block to be encoded is the 0-th code block and the number of filler bits is greater than zero, i.e., $F > 0$, then the encoder will set $c_k = 0$, $k = 0,\ldots,(F\text{-}1)$ at its input and will set d(0) k =< NULL >, $k = 0,\ldots,(F\text{-}1)$ and $d_k^{(0)}$ =< NULL >, $k = 0,\ldots,(F\text{-}1)$ at its output. The bits input to the turbo encoder are denoted by c_0, c_1, c_2, c_3, . . . , c_{K-1}, and the bits output from the first and second 8-state constituent encoders are denoted by z_0, z_1, z_2, z_3, . . . , z_{K-1} and z_0', z_1', z_2', … z_{k-1}' respectively. The bits output from the turbo code internal interleaver are denoted by c_0' , c_1', c_2', ….. c_{k-1}' and these bits are to be the input to the second 8-state constituent encoder.

4. CONCLUSION

In this paper, the feasibility of decoding arbitrary tailbiting convolutional and turbo codes using the BP algorithm was demonstrated. Using this algorithm to decode the tailbiting

convolutional code in WiMAX systems speeds up the error correction convergence and reduces the decoding computational complexity with respect to the ML-Viterbi-based algorithm. In addition, the BP algorithm performs a non-trellis based forward-only algorithm and has only an initial decoding delay, thus avoiding intermediate decoding delays that usually accompany the traditional MAP and SOVA components in LTE turbo decoders. However, with respect to the traditional decoders for turbo codes, the BP algorithm is about 1.7 dB worse at a BER value of 10^{-2}. This is because the nonzero element distribution in the parity-check matrix is not random enough. Also, there are a number of short cycles in the corresponding Tanner graphs. Finally, as an extended work, we propose the BP decoder for these codes in a combined architecture which is advantageous over a solution based on two separate decoders due to efficient reuse of computational hardware and memory resources for both decoders. In fact, since the traditional turbo decoders (based on MAP and SOVA components) have a higher complexity, the observed loss in performance with BP is more than compensated by a drastically lower implementation complexity. Moreover, the low decoding complexity of the BP decoder brings about end-to-end efficiency since both encoding and decoding can be performed with relatively low hardware complexity.

ACKNOWLEDGEMENT

We would like to express our sincere thanks to Sri. Dr. Kancharla Ramaiah Secretary and Correspondent, Prakasam Engineering College, Kandukur, A.P. India for his support with providing research environment . We are extremely thankful to our collogues , friends and family members who are cooperated in this work.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near Shannon limit error-correcting coding and decoding: Turbo codes,” IEEE Intl. Conf. on Commun., vol. 2, Geneva, Switzerland, pp. 1064-1070, 1993.

[2] R. M. Tanner, “A recursive approach to low complexity codes,” IEEE Trans. Inform. Theory, vol. 27, no. 5, pp. 533-547, 1981.

[3] G. D. Forney, Jr., “Codes on graphs: normal realizations,” IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 520–548, 2001.

[4] H. H. Ma and J. K. Wolf, “On Tail Biting Convolutional Codes,” IEEE Trans. On Commun., vol. 34, no. 2, pp. 104-111, 1986.

[5] 3GPP TS 45.003, “3rd Generation Partnership Project; Technical Specification Group GSM/EDGE Radio Access Network; Channel Coding (Release 7),” February, 2007.

[6] IEEE Std 802.16-2004, “IEEE Standard for Local and Metropolitan Area Networks – Part 16: Air Interface for Fixed Broadband Wireless Access Systems,” October, 2004.

[7] IEEE Std P802.16e/D10, “IEEE Standard for Local and Metropolitan Area Networks – Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems,” August, 2005.

[8] D. J. C. MacKay, “Good error-correcting codes based on very sparse matrices,” IEEE Trans. Inform. Theory, vol. 45, no. 2, pp. 399-431, 1999.

[9] N. Wiberg, “Codes and Decoding on General Graphs,” Linkoping Studies in Science and Technology, Dissertation No. 440, Linkoping University, -Linkoping, Sweden, 1996.

[10] R. J. McEliece, D. MacKay, and J.-Fu Cheng, “Turbo Decoding as an Instance of Pearl’s "Belief Propagation" Algorithm,” IEEE Trans. On Commun., vol. 16, no. 2, pp. 140-152, 1998.