Performance study of Non-binary LDPC Codes over GF(q)

V.S. Ganepola[1], R.A. Carrasco[1], I. J. Wassell[2] and S. Le Goff[1]
[1] School of Electrical, Electronic and Computer Engineering, University of Newcastle
Email:{v.s.ganepola, r.carrasco, stephane.le-goff}@ncl.ac.uk
[2] Computer Laboratory, University of Cambridge
Email : ijw24@cam.ac.uk

Abstract - Low-Density Parity Check (LDPC) codes are known to perform well in the presence of Additive White Gaussian Noise (AWGN) but for very large block lengths. It has been proposed to define the codes over high order Galois fields to overcome this limitation. In this paper we construct new quasi-cyclic non-binary LDPC codes with moderate code lengths from Reed-Solomon codes with two message symbols proposed by Lin et al defined over large finite fields. We evaluate the performance of these codes on the AWGN channel by computer simulation and show that they outperform binary LDPC codes of the same length in binary bits.

***Index Terms* – non-binary, LDPC codes, AWGN, FFT-BP decoding, finite fields.**

I. INTRODUCTION

Binary Low Density Parity Check (LDPC) codes over $GF(2)$ rediscovered by Mackay and Neal [3], [4] have been observed to display near Shannon limit performance when decoded using probabilistic soft decision decoding algorithms. However these near Shannon's limit performances are obtained for randomly constructed codes of very large block lengths. It has been shown that this limitation can be overcome by defining the code over higher order Galois Fields [5], [6], [8]. It is reasonable to assume that non binary LDPC codes perform better than binary LDPC codes on channels with noise bursts, given the fact that consecutive bits are grouped together forming symbols in the non binary alphabet $GF(q)$. The classical BP algorithm used in decoding non binary LDPC codes has a computational complexity dominated by $O(q^2)$ making the decoding over higher order fields computationally infeasible. However it has been shown that the belief propagation over $GF(q)$ can be conveniently transferred into frequency domain scaling down the complexity to $O(q.\log_2 q)$ [8]. This paper shows that non binary codes with only moderate code lengths outperform binary LDPC codes with large block lengths by considerable margins. We also demonstrate by simulation that with working in very high order fields we can approach channel capacity for shorter block lengths.

II. CONSTRUCTION OF NONBINARY LDPC CODES

Non binary LDPC codes over Galois fields $GF(q)$, where q is a prime number, can be seen as a generalization of binary LDPC codes over $GF(2)$. A vector space projected over a finite field $GF(q)$ is used to denote the elements in $GF(q)$. We select q to be of the form

$$q = 2^b \tag{1}$$

where b is an positive integer such that $b > 1$. The code is defined in terms of an ultra sparse parity check matrix H. K is denoted as the length of the message while N is used to denote the codeword length. We define the rate of the LDPC code R,

$$R = (N - M)/N \tag{2}$$

where $M = N - K$. The rectangular $[M \times N]$ ultra sparse parity check matrix H is constructed having a mean column weight γ at least equal to two while the row weight ρ is made as uniform as possible. Such algebraic construction methods ensure that 1) each row has exactly ρ number of elements; 2) each column has exactly γ number of elements; 3) any two rows or two columns have more than one place where they both have non zero components. The first two conditions ensure that the parity check matrix H has uniform row and column weights forming a (γ , ρ) regular LDPC code while the third condition ensures that the minimum distance of the code generated is at least $\gamma + 1$ and the Tanner graph of the code is free of cycles of length four. Algebraic construction methods of LDPC codes generally involve first constructing circulant permutation matrices using non binary elements in $GF(q)$ and dispersing them into a base matrix constructing the overall parity check matrix H. The sparse parity check matrices considered in this paper are regular parity check matrices having uniform row weights ρ and column weights γ generated based on Reed Solomon (RS) codes over $GF(q)$ with two information symbols [11], [12]. The non-zero elements of the parity check matrix H are defined from the RS code to maximize the entropy of the corresponding symbol of the syndrome vector such that

$$z = c \cdot H^T \tag{3}$$

where $c = (c_1, c_2, \ldots, c_n, \ldots c_N)$ such that $c_n \in GF(q)$ denotes a valid code word and H^T denotes the transpose of the parity check matrix H. Gaussian elimination can be used on the parity check matrix H in order to obtain the systematic generator matrix G.

III. DECODING OF LDPC CODES OVER FINITE FIELDS

The decoding problem in non-binary LDPC codes is to iteratively process check nodes and variable nodes and determine the most probable received code word $\hat{c}$ such that $z = \hat{c} \cdot H' = 0$, where the likelihoods of c is determined according to the channel model. The decoding of non binary LDPC codes using classical belief propagation (BP) algorithm was first proposed in [5] and [6] with BCJR algorithm for check node processing. The classical BP algorithm yielded a computational complexity dominated by $O(q^2)$ mainly owing to the BCJR block in the check node processing step, making the decoding over higher order fields computationally infeasible. Evidently BP decoding of LDPC codes over higher order fields require prohibitively large number of computations ruling them out for practical implementation. The idea of transferring the check node processing into the frequency domain and scaling down the decoding complexity was first proposed in [8] and followed up by [13]. This paper contains a simple description of the FTT-BP algorithm and shows how the Fast Hadamard Transforms can be used in check node processing.

Factor graphs used to decode non binary LDPC codes require two additional blocks 1) permutation block; 2) re-ordering block; compared to binary factor graphs. We can modify the binary factor as shown in Figure 1 to represent the iterative decoding of LDPC code. The factor graph is a bipartite graph consisting of set of variable nodes (circular blocks at the top) and check nodes (square blocks at the bottom). Inference over factor graph can accomplished by means of passing messages between check nodes and variable nodes alternatively. In non binary LDPC codes over $GF(q)$, the messages passed along the edges of the factor graph corresponds to q point discrete probability set rather than a single message. These probability distributions are exchanged iteratively in finding a valid code word.

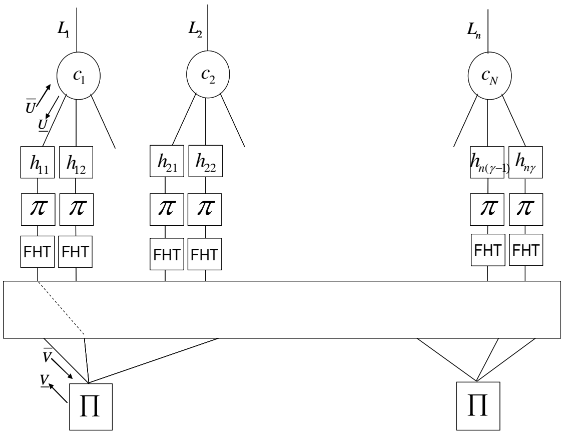

Figure 1. Factor graph of non binary LDPC codes decoding

The factor graph shown in Figure 1 connects variable nodes to check nodes through intermediate stages; permutation and re ordering. We can see from the factor graph that there are γ number of permutation blocks connected to each variable node and they each correspond to non-zero entries found under corresponding columns in the parity check matrix H. The permuted likelihood values are then sent through a re ordering block before connecting them to the check nodes. The algorithm is initialized with the likelihood values L of the received codeword. The likelihood value L_n of codeword c over $GF(q)$ corresponds to the probabilities of n th received codeword symbol being equal to each non binary element in $GF(q)$. If we define the n th received symbol by y_k and the likelihood value of the n th symbol being equal to a , $a \in GF(q)$ by $p(y_n | c_n = a)$, we can define the n th symbol likelihood values at the output . Each likelihood value L_n is connected to γ number of non-zero entries in each column. The symbol likelihood values are used to initialize the messages sent from each variable node towards check nodes during the first run of the decoder. We can see that this is just a copy of the symbol likelihood values along each edge connected to the variable node. If we use $\overline{U}$ and

$\underline{U}$ to denote the incoming and out going messages relative to the variable node respectively, we can denote the pdf message sent along the j th edge connected to the n th variable node by $\underline{U}_{jn}$. We can thus initialize the decoder by setting

$$\underline{U}_{jn} = L_n \tag{4}$$

It is quite convenient to reference the symbol likelihood values sent from variable nodes to the check nodes using the binary representation of the elements in $GF(q)$. This referencing system plays a pivotal role in the Fast Fourier transforming likelihood values as explained shortly. Each U_{jn} contains q number of discrete probabilities, and effectively becomes a probability distribution. The initialized symbol likelihood values are then sent from variable nodes to corresponding permutation blocks. It can be seen from the Figure 1 that γ number of permutations blocks are connected to each variable node, relating to each non-zero entry in the corresponding column. In the case of (γ, ρ) regular non binary code, there are ρ number of non-zero entries in each row. Therefore, if $h_{j1},\ldots h_{ji},\ldots,h_{j\rho}$ are the non-zero entries in the j th row of the parity check matrix H, we can write the j th parity check equation as,

$$\sum_{i=1}^{\rho} h_{ji} c_i \tag{5}$$

It is evident from the equation (5) that unlike in the case of binary LDPC, the parity check equations of non binary LDPC codes contain non-zero elements in its parity check equations. This implies that the probability distributions contained in the messages needs to be permuted accordingly, taking the non binary elements in the parity check equation into account. Due the structure of the Galois fields, the permutation block becomes a cyclic shift of the probabilities except for the probability of the received code word being equal to 0. We can see that the it requires i number of cyclic shifts in the direction of ascending order of filed elements are needed to permute the likelihood values in the case that non-zero entry $h_{ji} = \alpha^i$. The permuted pdfs are the subjected to check node processing, progressing the belief values further. The classical BP uses well known BCJR algorithm with all possible forward and back partial sums [5], [6] yielding a decoding complexity of $O(q^2)$.

It can be seen from the trellis that this method yields a computational complexity dominated by $O(q^2)$ ruling out the decoding over high order Galois fields. That this limitation can be conveniently overcome by transferring the check node processing in to frequency domain by using FFT and converting the convolutional node into a product node as shown in Figure 1. The FFT over Galois fields has a special structure that can be efficiently represented by a radix-2 butterfly diagram as shown in Figure 2. The Fourier transforms over finite sets, including Galois fields of $GF(q)$ where $q = 2^b$, can be decomposed in to set of 2nd order Fourier transforms applied along each dimension of field [8], [13]. The fast Fourier transform over finite fields, groups is reduced in to a recursive set of sums and differences of the values changed by its bit locations in the reference. We also define the dimension of the field elements as the bit location under consideration; ranging from 1 to b.

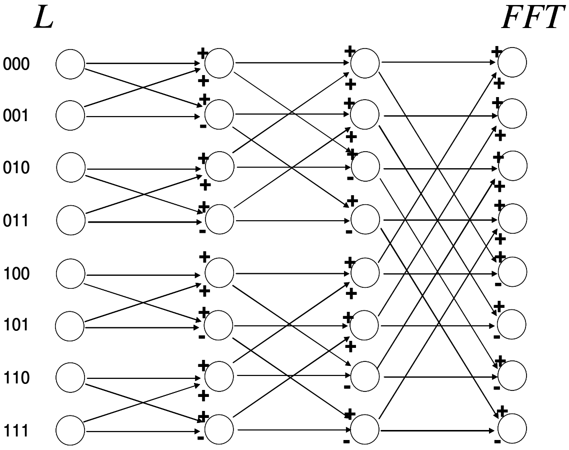

Figure 2. Radix-2 butterfly of FFT over $GF(8)$

The radix-2 butterfly shown in Figure 2 allows us to compute the sums and differences of the likelihood values recursively along each dimension with each recursion representing the application of 2nd order FFT, which is a sum and a difference, along a separate dimension. Further analysis confirms that the FFT over Galois field reduces in to Fast Hadamard Transform (FHT) which can be performed using Walsh-Hadamrd matrix of the order equal to the field order. We can use the elemental Walsh-Hadamard matrix H_2 which also corresponds to the 2nd order FFT over Galois fields in (6)

$$H_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{6}$$

and apply the observation (7) repeatedly in order to construct Walsh-Hadamard matrix matching the field order H_q.

$$H_{2n} = \frac{1}{\sqrt{2n}} \begin{bmatrix} H_n & H_n \\ H_n & -H_n \end{bmatrix} \tag{7}$$

This conjecture can be accommodated in our calculations by re-ordering the likelihood values projected $GF(q)$ in the ascending order before the

Fast Hadamard Transformation step. The re-ordered q point likelihood sets can then be transferred into frequency domain simply by multiplying the likelihood values by q order Walsh- Hadamard Matrix H_q as shown in equation (8). The Fast Hadamard Transform generates q number of discrete probability values.

$$W = F(U) = U^T.H_q \tag{8}$$

As described in Figure 1, the decoder consists of total M number of check nodes which were simplified into simple product nodes using FHT. Every check node connects ρ number of Fourier transformed q point probability sets using term-by-term product operation in frequency domain. The term-by-term product of two q point pdfs can be defined by

$$\begin{aligned} & V_1(1,\ldots q)\,.\,V_2(1,\ldots,q) \\ &= \left(V_1(1).V_2(1),\ldots,V_1(q).V_2(q)\right) \end{aligned} \tag{9}$$

The M number of rows of H is represented by M number of product nodes in the decoder and the ρ number of non-zero entries in each row is represented by ρ number of edges connected to each of the product node. The belief propagation across horizontal plane can then be computed simply by multiplying term-by-term all the Fourier transformed probability values connected into a product node with each other, except with the message along the edge under consideration totalling up to $(\rho - 1)$ pdfs. If we use $\overline{V}$ and $\underline{V}$ to denote the incoming and out going pdfs relative to the variable node, we can denote the output of the i th edge connected to m th product node by

$$\underline{V}_{mi} = \prod_{\substack{j=1 \\ j\neq i}}^{\rho} \overline{V}_{mj} \tag{10}$$

The pdf messages sent from the check nodes to the variable nodes take the same path in the opposite way. Every operational block on its path towards the check nodes implies the inverse of operation it performed on its way towards the check node. The pdf messages at the output of the check nodes are in the frequency domain and needs to be converted back to its dual. The FHT block in the opposite direction implies the inverse Fast Hadamard Transform applied to the q point pdf. We can observe that the inverse Fast Hadamard Transform is exactly the same as the Fast Hadamrd Transform and we can multiply each q point pdf set, again by the q order Walsh-Hadamard matrix in order to obtain the inverse Fast Hadamard Transform as shown below.

$$U = IFHT(V) = V^T.H_q \tag{11}$$

The messages are then subjected to the inverse of the re-ordering, dispersing the likelihood values in to their original locations followed up by the inverse of the cyclic shift operation, exactly the same number of cyclic shifts of the likelihood values it was subjected now in the opposite direction.

The likelihood values U are then processed at the variable nodes, propagating the belief values in the vertical plane. Variable node processing implies the term-by-term multiplication of $(\gamma - 1)$ likelihood values along the edges connected to variable node, except for the message along the edge under consideration. We can define the output of the j th edge connected to n th variable node by

$$\underline{U}_{jn} = \prod_{\substack{i=1 \\ i\neq j}}^{\gamma} \overline{U}_{in} \tag{12}$$

Hard decision is then taken on the likelihood values at the variable nodes by determining the symbols from the non binary alphabet $GF(q)$ as shown below

$$\hat{c}_n = \frac{\arg\max}{GF(q)} L_n \prod_{i=i}^{\gamma} \overline{U}_{in} \tag{13}$$

The frequency domain implementation of the check node processing reduces the complexity to $O(q.\log_2 q)$ in comparison with the complexity dominated by $O(q^2)$. It is interesting to note that this is exactly the same complexity reduction found in Cooley-Tukey FFT algorithm. The complexity reduction allows fast processing of vast amounts of data enabling decoding nonbinary LDPC codes defined over very large order Galois fields. Decoding over higher order fields facilitates near asymptotic performance, drawing closer to the channel capacity

V. PERFORMANCE COMPARISON

The optimum fast belief propagation decoding using Fourier transforms is considered with respect to different field orders. In order to make a fair comparison, the code words with respect to different finite fields are constructed to have the same amount of binary information. In Figure 3 the codes considered are $GF(2), N = 2040$), ($GF(4)$, $N = 1016$), ($GF(16)$, $N = 504$), ($GF(64)$, $N = 378$), ($GF(256)$, $N = 248$). All the codes are constructed having the same row weight $\rho = 8$ and the same column weight $\gamma = 4$ leading to exactly the same code rates in all five cases $R = 1/2$. These codes are simulated in Additive White Gaussian Noise (AWGN) Channel. Also represented in the Figure 3 is the Bit Error Rate

(BER) performance curve for the uncoded bits under the influence of the same channel conditions. It could be observed that there is indeed a performance gain in moving into higher order field reaching towards the Shannon's limit.

VI. CONCLUSION

In this paper we establish that working in a higher order Galois field, significantly improve the performance of the LDPC code with moderate code lengths. This can be quite convenient in implementation of LDPC codes as it yields reasonable performance even with relatively smaller frame sizes. The fast decoding algorithm based on fast Fourier transforms reduces the computational complexity of the belief propagation algorithm significantly and working in higher order Galois fields are made computationally feasible. It is demonstrated by simulation that there is a significant performance gain between the codes over $GF(2)$, $GF(4)$, $GF(16)$, $GF(64)$ and $GF(256)$. It is evident from the simulation results that using very high order LDPC codes we could draw near the channel capacity realizing near-asymptotic performance levels.

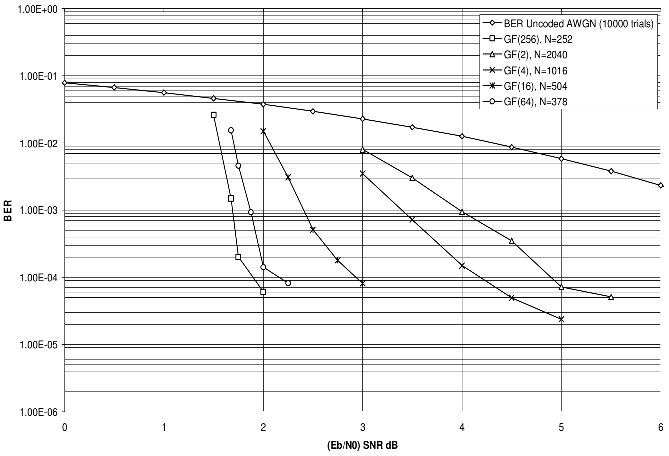

Figure 3. Performance comparison, LDPC code, $R = 1/2$, $GF(2)$, $N = 2040$), ($GF(4)$, $N = 1016$), ($GF(16)$, $N = 504$), ($GF(64)$, $N = 378$), ($GF(256)$, $N = 248$).

ACKNOWLEDGEMENT

This research is supported by the Engineering and Physical Sciences Research Council (EPSRC) UK, The UK government's leading funding agency for research and training in engineering and physical sciences.

REFERENCES

[1] V. Rathi and R. Urbanke, "Density Evolution, Thresholds and the Stability Condition for Non-binary LDPC Codes."

[2] R. Gallager and L. Codes, "Cambridge," in *MA: MIT Press Monograph*, 1963.

[3] D. MacKay and R. Neal, "Good Codes based on Very Sparse Matrices," *Cryptography and Coding: 5th IMA Conference, Cirencester, UK, December 18-20, 1995: Proceedings*, 1995.

[4] D. MacKay and R. Neal, "Near Shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 33, pp. 457-458, 1997.

[5] M. Davey and D. MacKay, "Low density parity check codes over GF (q)," *Information Theory Workshop, 1998*, pp. 70-71, 1998.

[6] M. Davey, "Error-correction using Low-Density Parity-Check Codes," *Univ. of Cambridge PhD dissertation*, 1999.

[7] D. Sridhara and T. Fuja, "Low density parity check codes over groups and rings," *Information Theory Workshop, 2002. Proceedings of the 2002 IEEE*, pp. 163-166, 2002.

[8] L. Barnault and D. Declercq, "Fast Decoding Algorithm for LDPC over GF (2^q)," *The Proc. 2003 Inform. Theory Workshop*, pp. 70–73, 2003.

[9] C. Poulliat, M. Fossorier, and D. Declercq, "Design of non binary LDPC codes using their binary image: algebraic properties," *Information Theory, 2006 IEEE International Symposium on*, pp. 93-97, 2006.

[10] S. Lin, S. Song, Y. Tai, L. Lan, and L. Zeng, "Algebraic Constructions of Nonbinary Quasi-Cyclic LDPC Codes," *Communications, Circuits and Systems Proceedings, 2006 International Conference on*, vol. 2, 2006.

[11] Y. Tai, L. Lan, L. Zeng, S. Lin, and K. Abdel-Ghaffar, "Algebraic Construction of Quasi-Cyclic LDPC Codes for the AWGN and Erasure Channels," *Communications, IEEE Transactions on*, vol. 54, pp. 1765-1774, 2006.

[12] B. Zhou, Y. Tai, L. Lan, S. Song, L. Zeng, and S. Lin, "CTH08-1: Construction of High Performance and Efficiently Encodable Nonbinary Quasi-Cyclic LDPC Codes," *Global Telecommunications Conference, 2006. GLOBECOM'06. IEEE*, pp. 1-6, 2006.

[13] D. Declercq and M. Fossorier, "Decoding Algorithms for Nonbinary LDPC Codes Over GF(q)," *Communications, IEEE Transactions on*, vol. 55, pp. 633-643, 2007.