Structured Low-density Parity-check Codes over Non-binary Fields

Sarah J. Johnson and Steven R. Weller

Abstract—**This paper considers low-density parity-check (LDPC) codes defined over non-binary finite fields $\mathrm{GF}(q), q = 2^p$. We find that the difficulty in constructing q-ary LDPC codes whose Tanner graphs are free of short cycles increases as the order of the field increases. By employing combinatorial designs to devise structured q-ary LDPC codes with guaranteed minimum girth, we show that significant improvements in performance over binary LDPC codes are possible, particularly in the case of short, or high rate, codes. We present a simple construction for rank deficient q-ary LDPC codes.**

Index Terms—**Low-density parity-check codes, non-binary codes, rank-deficient parity-check matrices.**

I. Introduction

Low-density parity-check (LDPC) forward error-correcting codes were first presented by Gallager [1] in 1962. Having been largely overlooked for some 35 years, LDPC codes are presently the focus of intense research interest in view of their Shannon limit approaching performance when combined with iterative decoding algorithms such as the sum-product algorithm [2], [3].

While a substantial, and rapidly expanding, literature now exists on binary LDPC code construction and decoding, comparatively little is known about generalizations to non-binary alphabets, in which codeword symbols are selected from finite fields $\mathrm{GF}(q), q = 2^p, p \geq 2$; see, for example [4]–[7]. In some of the earliest work on such q-ary LDPC codes, Davey and MacKay demonstrated that LDPC codes defined over non-binary fields can substantially outperform binary LDPC codes over the binary symmetric channel (BSC) and additive white Gaussian noise (AWGN) channel [4], [5].

Davey showed that there is an optimum column weight which decreases as the order of the field increases, concluding that the best results could be generated by choosing the highest order field that is feasible and then selecting an appropriate mean column weight [5]. As is the case for most binary LDPC code constructions, once the optimal weight distribution has been determined, the q-ary codes are constructed randomly. Consequently there are no guarantees on the performance or properties of any individually randomly generated code.

For very long codes this is not a problem as good codes are easily constructed randomly, and convergence to an ensemble average in the long codeword limit has been established [3]. However for short codes there is typically a significant performance gap between the best and worst codes of a particular ensemble and, especially for higher rate codes, good codes can be difficult to construct. By way of example, Davey notes that for q-ary LDPC codes with rates exceeding 2/3, and for codes of length 2000 or less, it is necessary to impose constraints on the parity-check matrix to avoid low weight codewords responsible for unacceptable error floors at high signal-to-noise ratios [5].

The constraints typically imposed on the pseudo-randomly constructed parity-check matrices of binary LDPC codes are that the parity-check matrix be regular (or nearly so), and that the code be free of short cycles, especially cycles of length 4. However, as we will see in this paper, the difficulties in constructing code without 4-cycles are compounded as the order of the field increases. As a solution to this problem we consider structured q-ary LDPC codes defined by taking as our starting point (binary) parity-check matrices chosen as the incidence matrices associated with certain combinatorial designs, selected to achieve the required code properties.

An interesting outcome of the research into binary algebraic LDPC codes has been the recognition of the key role played by rank deficient parity-check matrices in improving LDPC code performance [8]–[10]. In this paper we propose a method of incorporating linear dependence in q-ary LDPC codes and show that rank deficient parity-check matrices can also play a significant role in improving the performance of q-ary LDPC codes.

II. q-ary LDPC codes

A q-ary LDPC code is defined as the null space of a sparse parity-check matrix H having non-zero entries selected from a finite field of order $q = 2^p$, denoted $\mathrm{GF}(q)$. Thus in a q-ary LDPC code, each code symbol $c_i \in \mathrm{GF}(q)$ represents p data bits, and codewords $c = [\, c_1 \cdots c_N \,]$ satisfy $cH' = 0$.

For the sake of concreteness in this paper, we focus on LDPC codes defined over GF(4), although the results generalize naturally to higher order fields $\mathrm{GF}(q), q = 2^b$, for $b \geq 3$. The field GF(4) can be thought of as the element set $\{0, 1, \alpha, \alpha+1\}$, where $\alpha^2 = \alpha + 1$. Writing β in place of $\alpha + 1$, so that $\alpha\beta = \alpha(\alpha+1) = \alpha^2 + \alpha = 1$ and $\beta^2 = (\alpha+1)^2 = \alpha^2 + 1 = \alpha$, gives the field operations shown in Fig. 1.

For example, the matrix

$$H = \begin{bmatrix} 0 & 1 & 1 \\ \alpha & \beta & 0 \end{bmatrix}$$

is the parity-check matrix of a 4-ary code of rate $R = 1/3$ and length $N = 3$. The generator matrix for this code, obtained

Sarah Johnson is with the Wireless Signal Processing Program at National ICT Australia Ltd. email: sarah.johnson@nicta.com.au

Steven Weller is with the School of Electrical Engineering and Computer Science at the University of Newcastle, Callaghan, NSW 2308, Australia. email: steve@ee.newcastle.edu.au. This work was supported by Bell Laboratories Australia, Lucent Technologies, and by the Australian Research Council under Linkage Project Grant LP0211210.

$+$	0	1	α	β
0	0	1	α	β
1	1	0	β	α
α	α	β	0	1
β	β	α	1	0

$\cdot$	0	1	α	β
0	0	0	0	0
1	0	1	α	β
α	0	α	β	1
β	0	β	1	α

Fig. 1: Field operations for GF(4) $= \{0, 1, \alpha, \beta\}$

using Gaussian elimination, is

$$G = \begin{bmatrix} 1 & \beta & \beta \end{bmatrix},$$

and the codewords are thus:

$$[\,0\,0\,0\,], \quad [\,1\,\beta\,\beta\,], \quad [\,\alpha\,1\,1\,], \quad [\,\beta\,\alpha\,\alpha\,].$$

The Tanner graph of a q-ary LDPC code is defined as for a binary LDPC code, with one constraint vertex in the graph for each parity-check constraint, and a symbol vertex corresponding to each code symbol. A symbol vertex is connected to a constraint vertex only if the corresponding entry of H is non-zero. Tanner's introduction of such graphs [11] extended the single parity-check constraints of Gallager's LDPC codes to arbitrary linear code constraints. In the case of non-binary LDPC codes the code constraint is a single parity-check, over GF(q), of the connected symbol nodes. A code parity-check matrix is uniquely described by a Tanner graph, although translating from the graph to the matrix requires an ordering of nodes corresponding to an ordering of rows and columns in H.

A cycle in the Tanner graph of a q-ary LDPC code is defined as a sequence of connected vertices in the graph which start and end at the same vertex, and which contains no other vertex more than once. The length of the cycle is the number edges it contains, and the *girth* of a graph is the length of its shortest cycle. With a slight abuse of terminology, we call the girth of an LDPC code what is strictly the girth of the Tanner graph associated with the given parity-check matrix H for that code.

Traditionally, to construct q-ary LDPC codes a sparse binary matrix, H_2, is first constructed pseudo-randomly, as in [12], [13], and the non-zero entries in the binary matrix randomly assigned a value from the field GF(q) to give the matrix H_q [4], [5]. In this way the density and girth properties of the randomly generated binary matrix, H_2, are retained in the q-ary code. In this paper we propose instead that a structured binary incidence matrix be used as the basis for the non-binary code to better control the code properties. Secondly we show that better codes can be produced by carefully controlling the allocation of the field elements to the entries of H_2.

A. Description of q-ary sum-product decoding

The aim of sum-product decoding is to compute the *a posteriori probability* (APP) for each codeword symbol, $P_i^a = P\{c_i = a \mid \mathcal{N}\}$, which is the probability that the ith codeword symbol is a, $a \in GF(q)$, conditional on the event $\mathcal{N}$ that all parity-check constraints are satisfied. The *intrinsic* or *a priori probability*, Pi_i^a, is the original symbol probability independent of knowledge of the code constraints, and the *extrinsic* probability Pe_i^a represents what has been learned from the parity checks.

The extrinsic probability that symbol i is a from the jth parity-check equation is:

$$\mathrm{Pe}_{i,j}^a = \sum_{\mathbf{x}:x_i=a} P(z_j \mid \mathbf{x}) \prod_{i' \in B_j, i' \neq i} \mathrm{Pi}_{i',j}^a, \qquad (1)$$

where the set B_j is the set of codeword symbols in the jth parity-check equation of the code and $\mathbf{x}$ is the set of all valid codewords.

The estimated APP of the ith symbol at each iteration is the product of the intrinsic and extrinsic probabilities for that symbol:

$$\mathrm{Pi}_{i,j}^a = \gamma_{i,j} \mathrm{Po}_i^a \prod_{j' \in A_i, j' \neq j} \mathrm{Pe}_{i,j'}^a, \qquad (2)$$

where A_i is the set of parity-check equations which check on the ith code symbol. Here $\gamma_{i,j}$ is a scaling factor to ensure that the probabilities sum to 1, i.e. for all i, j,

$$\sum_{a \in \mathrm{GF}(q)} \mathrm{Pi}_{i,j}^a = 1.$$

The probability Po_i^a is the initial probability that symbol i is a based only on the received signal and knowledge of the channel.

The estimated APP probabilities at the end of the iteration are then

$$P_i^a = \delta_i \mathrm{Po}_i^a \prod_{j' \in A_i} \mathrm{Pe}_{i,j'}^a, \qquad (3)$$

where again δ_i is a scaling factor to ensure that for all i,

$$\sum_{a \in \mathrm{GF}(q)} P_i^a = 1.$$

Calculation of the extrinsic probability is via the forward backward algorithm. The forward probabilities

$$F_{i,j}^a = P\left(\sum_{k<i} H_{j,k} x_k = a\right), \qquad (4)$$

and backward probabilities,

$$B_{i,j}^a = P\left(\sum_{k>i} H_{j,k} x_k = a\right), \qquad (5)$$

together give the probability that check j is satisfied if symbol i is a:

$$\mathrm{Pe}_{i,j}^a = \sum_{s,t \in GF(q): H_{j,k}a+s+t=0} F_{i,j}^s B_{i,j}^t. \qquad (6)$$

For a detailed description of the application of the forward-backward algorithm to the calculation of extrinsic probabilities in the sum-product algorithm see [5].

The extrinsic symbol information obtained from a parity-check constraint is calculated independently of the a priori value for that symbol at the start of the iteration. However, the extrinsic information provided in subsequent iterations remains independent of the original a priori symbol probability only until that information is returned via a cycle. If the Tanner graph of the code is cycle free the probabilities remain independent and the exact APP value is calculated for each symbol. Decoding is terminated if the hard decision on the estimated APP probabilities, $(\hat{c})$, is a valid codeword $(\hat{c}H' = 0)$, or if the maximum number of allowed iterations has been reached.

To summarize, the operation of q-ary sum-product decoding is as follows.

- *Initialization:* The symbol probabilities are initialized with the received probabilities: $Pi^a_{i,j} = Po^a_i$
- *Step 1:* The extrinsic probabilities (1) are calculated using equations (4)–(6)
- *Step 2:* The partial APPs of each codeword symbol are estimated; see (2)
- *Step 3:* If the hard decision of the estimated symbol APPs (3) is a valid codeword, decoding halts, otherwise return to Step 1.

For each decoding iteration, the q-ary LDPC codes require the calculation of q separate extrinsic and intrinsic probabilities for each non-zero entry in H, as compared to the calculation of just two such probabilities for binary codes. However, the equivalent length binary LDPC code, $N_b = \log_2(q)N$, (with the same average column weight) will have $\log_2(q)$ times as many non-zero entries. Increasing the order of the field is comparable to increasing the memory of convolutional codes, since the state space of each node in the graph is increased by decoding over GF(q) [5].

III. Randomly constructed q-ary LDPC codes

In this section we show the performance of binary and 4-ary LDPC codes which are randomly constructed as in [4], [5]. To avoid biased results caused by randomly selecting a particularly good or bad code from the ensemble of all regular codes with a given size and column weight for H we have randomly generated a new LDPC code for each decoding trial.

Following Davey [5], we consider transmission on binary-input AWGN channels, so that noise is added bitwise and independently of the grouping of bits into symbols. The a priori symbol probability is the product of the a priori bit probabilities of its constituent bits.

In Fig. 2 the decoding performance of randomly constructed binary and 4-ary LDPC codes with equivalent rate (0.5) and length (100 bits) is shown. To generate the binary LDPC code requires a sparse binary matrix of size 50×100 while to generate a 4-ary LDPC code requires a sparse binary matrix with the same column weight but of size 25×50 for which the non-zero entries are randomly assigned values from GF(4).

We see that even for this very short code there is a modest performance improvement gained by considering 4-ary codes. However, as we shall see in the following, it is more difficult to constrain the code ensembles, for example by requiring a minimum girth of 6, as the field order is increased.

A. Constraining code properties

For short LDPC codes, particularly for higher rate codes, good codes can be difficult to construct and so there are a number of modifications to the pseudo-random construction to attempt to remove any 4-cycles in the randomly generated binary matrix (see e.g. [12], [14]). We employ the pseudo-random code construction procedure of MacKay and Neal [12], using source code from [13].

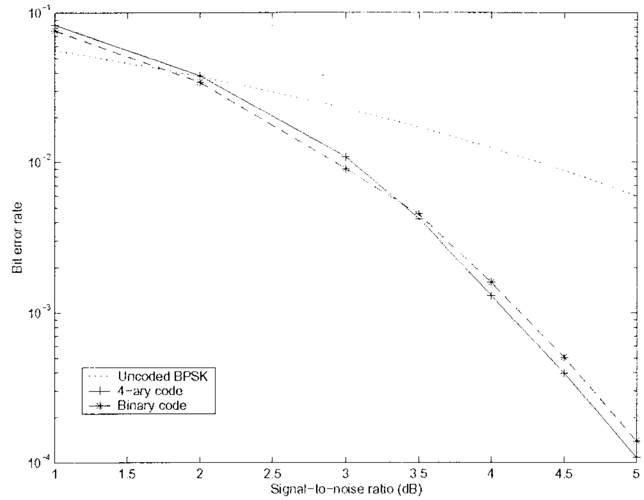

Fig. 2: The error correction performance of binary and 4-ary LDPC codes with equivalent rate (0.5) and length (100 bits). The codes are constructed randomly with column weight 3 and row weight 6.

Using this process it becomes significantly more difficult to construct 4-cycle free codes as the field order is increased. This is because the size of the initial binary matrix required is $\log_2(q)$ times smaller than for the binary code, and so as q is increased we need to be able to generate progressively smaller matrices which are free of 4-cycles, a significantly more difficult task.

For example, we use the same code parameters as in the previous section, a code with binary length of 100 bits and rate 1/2, but this time attempt to remove cycles from the LDPC codes. To generate a binary LDPC code requires a sparse binary matrix of size 50×100 which is 4-cycle free while to generate a 4-ary LDPC code requires a sparse binary matrix with the same column weight but of size 25×50 which is also 4-cycle free. With a factor of just $1/2$ in the matrix size, we are now unable to remove most of the 4-cycles using the construction method of [12].

The average performance of binary and equivalent length 4-ary codes constructed in this way is shown in Fig 3. With a maximum of just one iteration, cycles in the code can have no effect and so the performance of both codes is similar. However, since we were unable to remove 4-cycles from the majority of the 4-ary codes we see that, as expected, the performance of the 4-ary code is worse than the equivalent length binary code once more than one iteration is allowed. For this reason we consider in the following the algebraic design of q-ary LDPC codes to achieve the code properties we require.

IV. Structured q-ary LDPC codes

A natural solution to the difficulty of pseudo-randomly constructing small high rate q-ary LDPC codes is to consider algebraically constructed matrices, using incidence structures from combinatorial designs and finite geometries, a technique

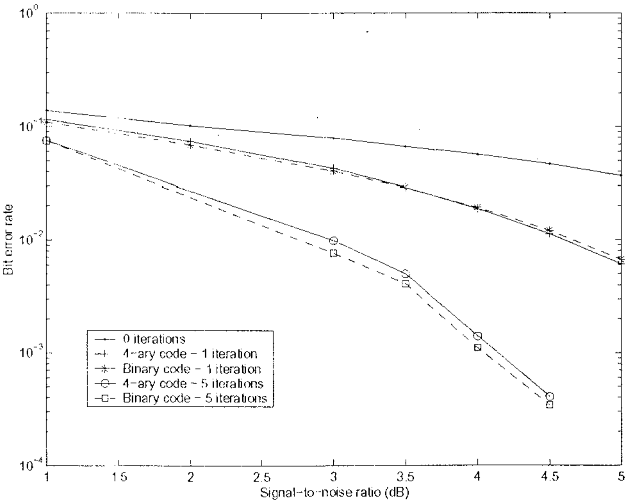

Fig. 3: The error correction performance of binary and 4-ary LDPC codes with equivalent rate (0.5) and length (100 bits) using sum-product decoding with a maximum of 5 iterations. The codes are constructed randomly with column weight 3 and row weight 6, and with modifications made to the codes to avoid 4-cycles.

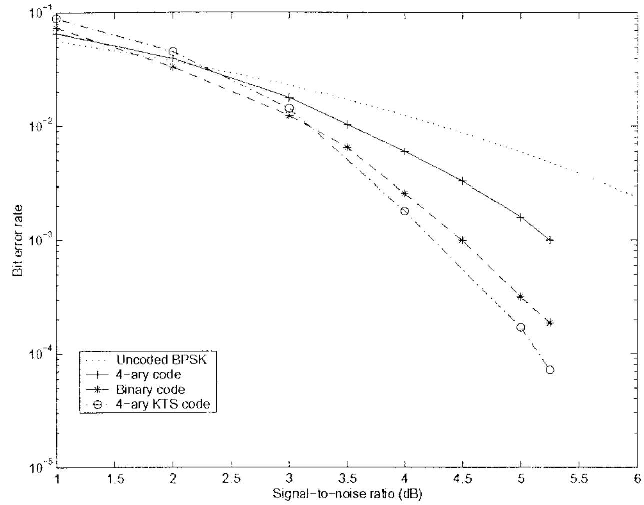

Fig. 4: The error correction performance of binary and 4-ary LDPC codes with equivalent rate (0.57) and length (70 bits) using sum-product decoding with a maximum of 5 iterations. The value of the entries in the 4-ary KTS codes are allocated randomly however the position of the non-zero entries remains as determined by the incidence of the KTS(15) design. The 4-ary KTS code is compared to both binary and 4-ary codes constructed randomly with column weight 3, row weight 7, and 4-cycles removed when possible.

which has proven successful for the construction of binary LDPC codes with guaranteed minimum girth and minimum distance [8]–[10], [15]–[18].

In this section we design q-ary LDPC codes from finite incidence structures known as Kirkman triple systems (KTSs) and unital designs. Let $\mathcal{P}$ be a v-set, and suppose that $\mathcal{B}$ is a collection of γ-subsets of $\mathcal{P}$ with the property that each t-subset of $\mathcal{P}$ is in exactly λ of the elements of $\mathcal{B}$. Then the ordered pair $\mathcal{D} = (\mathcal{P}, \mathcal{B})$ is called a t-(v, γ, λ) *design*, or simply a t-design. The elements of $\mathcal{P}$ are called *points*, and the elements of $\mathcal{B}$ are *blocks*.

If $t \geq 2$ and $\lambda = 1$, then a t-design is called a *Steiner system*, or a Steiner t-design. A *Steiner 2-design* thus has the property that every pair of points in the design occur together in exactly one block of the design. A *resolution* of a design $\mathcal{D}$ is a partition of the blocks of $\mathcal{D}$ into classes such that each point of $\mathcal{D}$ is in precisely one block from each class; such a design is said to be *resolvable*.

Resolvable Steiner 2-designs with block size $\gamma = 3$ are known as Kirkman triple systems. It is these incidence structures that we employ in this section; see also [16]. The second class of incidence structures that we employ are the unital designs, an infinite class of 2-$(m^3 + 1, m + 1, 1)$ designs; see [19] for construction details and an application to the construction of binary LDPC codes.

Given a binary parity-check matrix, we randomly assign values from GF(q) to the non-zero entries of structured binary matrices, and maintain their guaranteed regularity and minimum girth. The minimum distance properties guaranteed for binary LDPC codes free of 4-cycles can also be translated to q-ary LDPC codes, as we see in the following.

Lemma 1: The minimum distance of a q-ary LDPC code with column weight γ and no 4-cycles is:

$$d_{\min} \leq \gamma + 1.$$

Proof: The result follows from Massey's observation that a weight w codeword in a code with parity-check matrix H corresponds to a set of w columns in H which are linearly dependent [20]. For a binary code this requires that any row of H incident in the set of w columns is incident an even number of times. For non-binary LDPC codes the requirement for linearly dependent columns is relaxed, namely that any row of H incident in the set of w columns is incident at least twice. However, if 4-cycles are avoided in H at least $\gamma + 1$ columns are needed to ensure that every row incident in the column set is incident twice and so in both cases, binary and q-ary, the code minimum distance is lower bounded by $\gamma + 1$. ■

Note that the minimum distance of the q-ary LDPC code will not necessarily be the same as the binary code constructed from the same matrix since different allocations of q-ary values to the non-zero entries in H can change the linear dependence of the columns.

To demonstrate the benefit of constructing q-ary LDPC codes using the incidence matrices of combinatorial designs, a very short ($N = 35$) and high rate ($R = 0.7$) 4-ary LDPC code which is free of 4-cycles is constructed by randomly allocating values from GF(4) to the non-zero entries of the incidence matrix of a Kirkman triple system on 15 points. Fig. 4 shows the performance of this code compared to randomly constructed binary LDPC codes of the same rate and equivalent length ($N_b = 70$).

The performance improvement achieved by using algebraic 4-ary LDPC codes is significant for such a short code: 1 dB gain at a bit error rate of 10^{-3}. The randomly constructed 4-ary codes perform quite poorly compared to their binary equivalents. However, due in most part to the guaranteed girth, the 4-ary KTS codes significantly outperform the randomly constructed 4-ary codes and outperform the binary codes as well.

A. Designing rank deficient q-ary LDPC codes

An interesting outcome of the research into binary algebraic LDPC codes has been the recognition of the role of rank deficient parity-check matrices in improving LDPC code performance [8]–[10]. Thus q-ary LDPC codes with parity-check matrices containing a significant portion of linearly dependent rows are investigated here.

In generating a q-ary LDPC code from a binary matrix which is rank deficient, assigning entries randomly will not necessarily preserve the linear dependence. To preserve linear dependence we require that the q-ary sum over any combination of rows of H_q is zero if the binary sum of those rows in H_2 is zero. Thus it is the allocation of the entries within a column rather that between columns which are important. An examination of the field operations for GF(4) in Fig. 1 shows that to preserve linear dependence we need only to allocate the same q-ary value to all of the non-zero entries in a column. Which q-ary value is assigned to each column can be randomly allocated across the matrix.

For example, a length 63, rate $2/3$, 4-ary LDPC code which is free of 4-cycles and contains 7 linearly dependent rows in H_q can be constructed by starting with the incidence matrix of the unital design on 28 points.

Note that had we wished to maintain the distance distribution of H_2 this could have been achieved by allocating the same field element to every entry in each row of H_q. However since the same minimum distance bound is guaranteed for both H_2 and H_q, regardless of the distribution of non-zero entries, and since the q-ary code would be expected on average to have a better minimum distance than the binary code this would not be expected to improve the code performance.

Fig. 5 shows the performance of this code compared to randomly constructed binary LDPC codes of the same rate and equivalent length. By constructing a q-ary code with linearly dependent parity-check equations a significant performance gain can be achieved over randomly constructed full rank codes with the same length and rate.

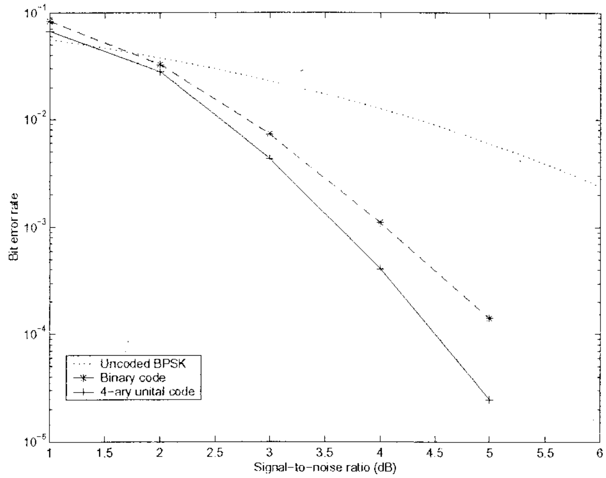

Fig. 5: The error correction performance of binary and 4-ary LDPC codes with equivalent rate (2/3) and length (126 bits) using sum-product decoding with a maximum of 5 iterations. The value of the entries in the 4-ary unital codes are uniform over a column of H with column values allocated randomly, however the position of the non-zero entries remains as determined by the incidence of the unital design. The 4-ary unital code is compared to both binary and 4-ary codes constructed randomly with 4-cycles removed when possible.

V. Discussion and Future Directions

In this paper we considered algebraic constructions for non-binary LDPC codes. We find that the difficulty in constructing q-ary LDPC codes which are free of small cycles increases as the code order is increased and thus algebraic constructions which guarantee good girth are even more beneficial for non-binary LDPC codes.

The codes we have considered here have not been optimized with regards to either the choice of column weight or field order and so further performance improvements are likely. Future work will also consider the increased flexibility in designing matrices with many linearly dependent rows which is provided by increasing the field order.

References

[1] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. IT-8, no. 1, pp. 21–28, January 1962.

[2] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electron. Lett.*, vol. 32, no. 18, pp. 1645–1646, March 1996, Reprinted *Electron. Lett*, vol. 33(6), pp. 457–458, March 1997.

[3] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 599–618, February 2001.

[4] M. C. Davey and D. MacKay, "Low-density parity check codes over GF(q)," *IEEE Commun. Letters*, vol. 2, no. 6, pp. 165–167, June 1998.

[5] M. C Davey, *Error-correction using low-density parity-check codes*, Ph.D. thesis, University of Cambridge, Cambridge, 1999.

[6] K. Nakamura, Y. Kabashima, and D. Saad, "Statistical mechanics of low-density parity check error-correcting codes over Galois fields," *Europhys. Lett.*, vol. 56, no. 4, pp. 610–616, November 2001.

[7] H. Song and J. R. Cruz, "Reduced-complexity decoding of Q-ary LDPC codes for magnetic recording," *IEEE Trans. Magn.*, vol. 39, no. 2, pp. 1081–1087, March 2003.

[8] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity-check codes based on finite geometries: A rediscovery and new results," *IEEE Trans. Inform. Theory*, vol. 47, no. 7, pp. 2711–2736, November 2001.

[9] P. O. Vontobel and R. M. Tanner, "Construction of codes based on finite generalized quadrangles for iterative decoding," in *Proc. International Symposium on Information Theory (ISIT'2001)*, Washington, DC, June 24–29 2001, p. 223.

[10] S. J. Johnson and S. R. Weller, "Codes for iterative decoding from partial geometries," in *Proc. International Symposium on Information Theory (ISIT'2002)*, Lausanne, Switzerland, June 30–July 5 2002, p. 310.

[11] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. IT-27, no. 5, pp. 533–547, September 1981.

[12] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, no. 2, pp. 399–431, March 1999.

[13] R. M. Neal, ," ⟨www.cs.toronto.edu/radford/homepage.html⟩.

[14] J. A. McGowan and R. C. Williamson, "Removing loops from LDPC codes," in *Proc. Australian Communications Theory Workshop (AusCTW'03)*, Melbourne, Australia, 5–7 February 2003, pp. 140–143.

[15] R. Lucas, M. P. C. Fossorier, Y. Kou, and S. Lin, "Iterative decoding of one-step majority logic decodable codes based on belief propagation," *IEEE Trans. Commun.*, vol. 48, no. 6, pp. 931–937, June 2000.

[16] S. J. Johnson and S. R. Weller, "Resolvable 2-designs for regular low-density parity-check codes," *IEEE Trans. Commun.*, vol. 51, no. 9, pp. 1413–1419, September 2003.

[17] S. J. Johnson and S. R. Weller, "A family of irregular LDPC codes with low encoding complexity," *IEEE Commun. Letters*, vol. 7, no. 2, pp. 79–81, February 2003.

[18] S. R. Weller and S. J. Johnson, "Regular low-density parity-check codes from oval designs," *Eur. Trans. Telecommun.*, To appear.

[19] S. J. Johnson and S. R. Weller, "High-rate LDPC codes from unital designs," in *Proc. IEEE Globecom Conf.*, San Francisco, CA, December 1–5 2003.

[20] J. L. Massey, *Threshold Decoding*, M.I.T. Press, Cambridge, Massachusetts, 1963.