Design of Nonbinary Quasi-Cyclic LDPC Cycle Codes

Rong-Hui Peng and Rong-Rong Chen
Dept. of Electrical and Computer Engineering
University of Utah, Salt Lake City, UT 84112
Email: {peng, rchen}@eng.utah.edu

***Abstract*— In this paper, we study the design of nonbinary low-density parity-check (LDPC) cycle codes over Galois field GF(q). First, we construct a special class of nonbinary LDPC cycle codes with low error floors. Our construction utilizes the cycle elimination algorithm to remove short cycles in the normal graph and to select nonzero elements in the parity-check matrix to reduce the number of low-weight codewords generated by short cycles. Furthermore, we show that simple modifications of such codes are parallel sparse encodable (PSE). The PSE code, consisting of a quasi-cyclic (QC) LDPC cycle code and a simple tree code, has the attractive feature that it is not only linearly encodable, but also allows parallel encoding which can reduce the encoding time significantly. We provide a systematic comparison between nonbinary coded systems and binary coded systems. For the MIMO channel considered, our results show that the proposed nonbinary system employing the PSE code outperforms not only the binary LDPC code specified in the 802.16e standard, but also the optimized binary LDPC code obtained using the EXIT chart methods.**

I. Introduction

In this work, we consider a special class of nonbinary LDPC codes with the property that each column of the parity-check matrix contains exactly two nonzero elements. The binary counterpart of such codes are called "cycle" codes in the literature [1]. A distinguished feature of cycle codes is that they are linearly encodable. However, binary cycle codes usually do not perform well due to poor minimum distance spectrum. When the size of the Galois field is sufficiently large, [2] shows that the hamming weight spectrum of the random ensembles of nonbinary cycle codes asymptotically approach the classical binomial distribution of the Shannon equiprobable random ensembles. This makes nonbinary cycle codes good candidates for both optimum maximum likelihood (ML) and iterative decoding. Our results show that nonbinary LDPC cycle codes achieve excellent performance over MIMO channels. To reduce encoding complexity, we propose a class of nonbinary codes called the parallel sparse encodable (PSE) codes, each consisting of a quasi-cyclic (QC) LDPC cycle code and a simple tree code. The encoding complexity of the parallel sparse codes is $O(md_c)$, where m is the number of checks in the LDPC code, and d_c is the degree of check node. The QC structure of such codes facilitates parallel encoding which results in significant reduction of encoding time.

The contributions of this paper are summarized as follows:

This work is supported in part by NSF under grant ECS-0547433.

(1) We construct a special class of QC nonbinary LDPC cycle codes with low error floors. Our construction starts with a mother base matrix with large girth. Then we utilize the cycle elimination algorithm to remove short cycles in the normal graph and to select nonzero elements in the parity-check matrix to reduce the number of low-weight codewords generated by short cycles. Our construction reduces the error floor of nonbinary codes significantly.

(2) We propose the use of QC nonbinary LDPC cycle code for MIMO channels. Starting from any base QC nonbinary LDPC cycle code, which in general is not sparse encodable, we construct a PSE code which allows not only linear-time encoding but also parallel implementation. For PSE codes, our encoding method has a much lower complexity than that of the encoding method in [3]. Furthermore, we show that the PSE code achieves a performance that is very close to the base code at a much reduced encoding complexity. Compared to other nonbinary LDPC codes in the literature, such as the randomly constructed LDPC codes and the algebraically constructed codes in [4], the proposed PSE code is much more amenable for implementation due to its simple structure.

(3) Our results show that the proposed joint detection and decoding (JDD) system employing the nonbinary PSE code over GF(16) outperforms the JDD system employing the best *optimized* binary LDPC code in [5] by 0.38 dB. Due to its highly irregular degree sequence, the encoding complexity of the optimized binary LDPC code is also higher than that of the proposed PSE encodable code. When compared with a more practical QC binary LDPC code defined in the 802.16e standard [6] that is amenable for implementation, the proposed JDD system employing the PSE code achieves a larger performance gain of 0.6 dB.

II. System Model

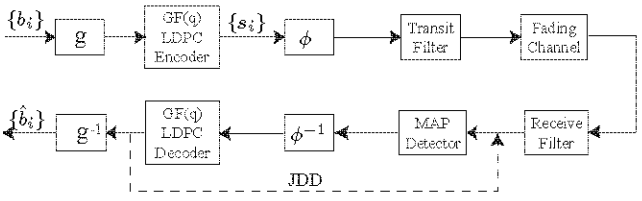

Fig. 1. A schematic block diagram of JDD system.

Fig. 1 shows a block diagram of the nonbinary LDPC coded

MIMO system. Assume that a LDPC code over GF(q) is used, where $q = 2^p$. At the transmitter side, a sequence of information bits $\{b_i\}$ is mapped to a sequence of nonbinary symbols in GF(q) (every p bits are mapped to a single nonbinary symbol) through a bit-to-symbol mapper g, before passing to the nonbinary LDPC encoder. Let t denote the number of transmit antennas. At the output of the LDPC encoder, every group of n_0 coded nonbinary symbols $\mathbf{s} = \{s_1, \cdots, s_{n_0}\} \in \text{GF}(q)$ is mapped to a group of t constellation symbols $\mathbf{x} = (x_1, \cdots, x_t) = \phi(\mathbf{s})$ through the mapper ϕ. Given the constellation size $M = 2^{m_0}$, we have $p \cdot n_0 = t \cdot m_0$. The sequence of constellation symbols is then passed to the transmit filter and sent through the t transmit antennas. At the receiver side, optimal maximum *a posterior* probability (MAP) detection is performed to compute the prior probabilities for each group of t transmitted constellation symbols. These prior probabilities will then be passed (after the mapper ϕ^{-1}) to the LDPC decoder for iterative decoding. After a finite number of decoding iterations, hard decisions on the nonbinary symbols are made at the output of LDPC decoder, which are then demapped to the sequence of estimated information bits.

When $n_0 > 1$, the prior probabilities of the group of n_0 nonbinary symbols are dependent because they are mapped to complex symbols that are transmitted simultaneously. For such systems, it is necessary to pass soft information about the dependent symbols from the LDPC decoder back to the MAP detector to produce updated symbol-wise probabilities. This corresponds to a JDD system that performs joint detection and decoding. As shown in Fig. 1, the JDD system requires a feedback loop from the channel decoder to the MAP detector to allow iterative exchange of soft information.

Next, we explain how the MAP detector shown in Fig. 1 works. The channel model is given by

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n} \tag{1}$$

where $\mathbf{x} \in \mathcal{C}^{t\times 1}$ denotes the complex transmitted signal vector, $\mathbf{y} \in \mathcal{C}^{r\times 1}$ denotes the complex received signal vector, r is the number of receive antennas, $\mathbf{H} \in \mathcal{C}^{r\times t}$ denotes the channel fading matrix with independent and identically distributed (i.i.d.) entries that are complex Gaussian distributed with zero mean and unit variance, $\mathbf{n} \in \mathcal{C}^{r\times 1}$ denotes the vector of zero mean, complex Gaussian white noise with variance σ^2 per dimension. We assume that the channel matrix is known to the receiver but not to the transmitter.

Given each received signal vector $\mathbf{y}$, we perform MAP detection to determine the a *posterior* probabilities (APP) of each nonbinary symbol $s_j, j = 1, \cdots, n_0$, by computing the log-likelihood-ratio vector (LLRV) over GF(q). Let $\{0, \alpha_1, \cdots, \alpha_{q-1}\}$ denote elements in GF(q). The LLRV of s_j is defined by $\mathbf{z} = \{z_0, z_1, \cdots, z_{q-1}\}$, where $z_i = \ln[P(s_j = 0)/P(s_j = \alpha_i)]$. From equation (1) we have

$$z_i = \ln \frac{\sum_{\mathbf{s}: s_j = 0} \exp[-\|\mathbf{y} - \mathbf{H}\phi(\mathbf{s})\|^2/(\mathbf{2}\sigma^2)]\mathbf{P}(\mathbf{s})}{\sum_{\mathbf{s}: s_j = \alpha_i} \exp[-\|\mathbf{y} - \mathbf{H}\phi(\mathbf{s})\|^2/(\mathbf{2}\sigma^2)]\mathbf{P}(\mathbf{s})} \tag{2}$$

where $\|\cdot\|^2$ denotes the norm square of a vector and $P(\mathbf{s})$ denotes the prior probabilities of $\mathbf{s}$ which are passed from the LDPC decoder. Subsequently, these LLRV values are passed to the LDPC decoder for iterative decoding.

III. Construction of quasi-cyclic nonbinary LDPC cycle codes

In this section, we describe the construction of nonbinary QC cycle codes. The code construction consists of several steps. First, we use cage graphs to construct a binary mother matrix with large girth. Second, we replace each "1" in the mother matrix with a binary circulant permutation matrix to yield a binary QC code. Here, we apply the cycle elimination (CE) algorithm to design the shifting coefficients of the permutation matrices such that shorts cycles in the parity-check matrix are removed. Third, we obtain the nonbinary QC cycle code from the binary QC code by replacing each binary permutation matrix by a nonbinary, *β-multiplied circulant permutation matrix*. We show that a modified version of the CE algorithm can be applied to choose the appropriate nonzero elements in the permutation matrices. This reduces the number of low-weight codewords generated by the remaining cycles of the parity-check matrix. Next, we elaborate on each of these steps in detail.

A. Construction of the mother matrix

In order to obtain a QC cycle code with large girth, we first design a mother matrix with the maximal girth possible for a given code length. It is well known that a cycle code can be represented by *normal graphs* [1], [7], where each row of the parity-check matrix H corresponds to a *vertex* and each column corresponds to an *edge* whose two *end vertices* correspond to the two rows with nonzero elements in that column. This motivates us to use cage graphs to construct mother matrices with large girths. A (k, g) cage graph is a graph that has the minimal number of vertices for a specified vertex degree k and girth g [8]. Hence, a cage graph has the largest girth among graphs with the same number of vertices.

A list of known cage graphs is presented in [8]. However, cage graphs are not available for arbitrary code lengths. In our construction, we apply some simple search methods to expand existing cage graphs to graphs with the desired number of vertices, while keeping the girth as large as possible. For example, assume that we want to construct a GF(16) (2, 4) QC code of length 600, $\zeta = 15$. The number of vertices in the corresponding normal graph is 20. From the table in [8], we select a $(4, 5)$ cage graph with 19 vertices. By slightly modifying the selected cage graph, we can construct a mother matrix with 20 vertices and still keep the girth to be 5. This is the largest girth possible for this code length.

B. Design of binary circulant permutation matrices through cycle elimination

Once the mother matrix is constructed, we replace each "1" in the mother matrix by a circulant permutation matrix P^{a_i} to obtain a binary QC cycle code. Here P^{a_i} is a circulant permutation matrix obtained by shifting the identity matrix to the right by a_i positions. We then search for appropriate cyclic shift coefficients $\{a_i\}$ using similar approaches as described in [9] [10].

The following property [9] relates the cycles of the mother code to the cycles of the QC code. To avoid confusion, we

refer to a cycle in the mother matrix as a block cycle. Let $P^{a_1} \rightarrow P^{a_2} \rightarrow \cdots \rightarrow P^{a_{2l}} \rightarrow P^{a_1}$ be a chain of permutation matrices corresponding to a $2l$-block-cycle. Here P^{a_i} and $P^{a_{i+1}}$ are located in either the same column block or the same row block of H and both P^{a_i} and $P^{a_{i+2}}$ are located in the distinct column blocks and row blocks. If r is the least positive integer such that

$$r \cdot \sum_{i=1}^{2l} (-1)^{i-1} a_i \equiv 0 \bmod \zeta, \tag{3}$$

then the block-cycle leads to a cycle (in H) of length $2lr$.

This property implies that, in order to eliminate a cycle of length $2lr$, we should avoid those cyclic shift coefficients $\{a_i\}$ that satisfy equation (3). Based on this idea, in [10], a cycle elimination (CE) algorithm is developed to search for proper cyclic shift coefficients to eliminate cycles of length less than some specified value. For each block cycle detected in the mother matrix, a constraint $r \cdot \sum_{i=1}^{2l} (-1)^{i-1} a_i \neq 0 \bmod \zeta$ is generated and added to a constraint set. Subsequently, a search procedure is executed to find the cyclic shifts $\{a_i\}$ to meet all the constraints in the constraint set.

C. Selection of nonzero elements over GF(q)

After the previous two steps, we obtain a binary QC cycle code with most of its short cycles removed. In the next step, we replace the binary circulant permutation matrix with a β-multiplied circulant permutation matrix. Analog to the binary case, each row of the β-multiplied circulant permutation matrix is the right cyclic-shift of the row above it multiplied by β and the first row is the right cyclic-shift of the last row multiplied by β, where $\beta = \alpha^{\lambda}$, α is the primitive element of GF(q), ζ is the size of circulant matrix, $\lambda = (q-1)/\zeta$ and $\zeta | (q-1)$. The nonzero element δ in the first row of the circulant matrix is randomly chosen from GF(q). The *β-multiplied circulant permutation matrix* defined here generalizes the *α-multiplied circulant permutation matrix* in [4]. Our construction allows flexible choices of ζ and code lengths (integer multiples of ζ), while [4] assumes $\zeta = q-1$ and the code length is restricted to integer multiples of $q-1$.

For nonbinary cycle codes, carefully chosen nonzero elements can reduce the number of low weight codewords generated by short cycles. This is a direct consequence of the following lemma.

Lemma 3.1: Given the normal graph of a cycle code, if there exists a cycle of length w and its parity check matrix is rank-deficient, then there must exist a codeword of weight w.

The proof of Lemma 3.1 is omitted here for brevity. Similar results are also presented in [11] where a *full rank condition* (FRC) is introduced. The FRC is shown to be equivalent to the following equation:

$$(\text{FRC}) : \prod_i \beta_{2i+1} \neq \prod_i \beta_{2i}, \tag{4}$$

where $\{\beta_i\}$ denote nonzero elements in a cycle, and β_i and β_{i+1} are located in either the same column or the same row of H, and both β_i and β_{i+2} are located in distinct columns and rows. When the nonzero elements in a cycle are chosen such that the FRC is satisfied, the low weight codeword generated by this cycle can be eliminated.

Let $\beta_i = \alpha^{\rho_i}$, then equation (4) can be written as

$$\sum_i (\rho_{2i+1} - \rho_{2i}) \neq 0 \bmod (q-1). \tag{5}$$

Searching for the nonzero elements that satisfy the FRC may have high complexity when the code length is large. In the following, we show that, by utilizing the special structure of QC codes, the FRC can be simplified using β-multiplied circulant permutation matrix.

Theorem 3.2: Let $P^{a_1} \rightarrow P^{a_2} \rightarrow \cdots \rightarrow P^{a_{2l}} \rightarrow P^{a_1}$ be the chain of permutation matrices corresponding to a $2l$-block-cycle and the block-cycle leads to a cycle of length $2lr$. Then the FRC over the cycle is equivalent to:

$$r \cdot \sum_{i=1}^{2l} (-1)^{i-1} \rho_i \neq 0 \bmod (q-1), \tag{6}$$

where ρ_i corresponds to the power of the nonzero element in the first row of P^{a_i}.

Proof: We follow similar proofs of Proposition 3 in [9]. Without loss of generality, we assume that P^{a_1} and P^{a_2} are located in the same row block, and P^{a_2} and P^{a_3} are in the same column block and so on. Considering a cycle of length $2lr$ starting from the j-th row of P^{a_1}, where the power of the nonzero element is $((\rho_1 + j\lambda) \bmod (q-1))$. Since P^{a_2} is located in the same row block as P^{a_1}, the nonzero element in the cycle at P^{a_2} has the power of $((\rho_2 + j\lambda) \bmod (q-1))$. The nonzero element in the j-th row at P^{a_2} is located in the $((j + a_2) \bmod \zeta)$-th column. Hence, the nonzero element of P^{a_3} in the cycle is located in the $((j + a_2 - a_3) \bmod \zeta$-th row, and it has the power of $((\rho_3 + (j + a_2 - a_3)\lambda) \bmod (q-1))$ (since $\zeta\lambda = q - 1$). Continuing this process, it is straightforward to check that the nonzero element of the cycle which comes across P^{a_1} the second time is $((\rho_1 + (j + a_2 - a_3 + \cdots + a_{2l} - a_1)\lambda) \bmod (q-1))$. Repeating this process $r-1$ times until it returns to the j-th row of P^{a_1}. Hence, the FRC can be written as

$$r \cdot \sum_{i=1}^{2l} (-1)^{i-1} \rho_i + r\lambda \cdot \sum_{i=1}^{2l} (-1)^{i-1} a_i \neq 0 \bmod (q-1), \tag{7}$$

Since it is a cycle of length $2lr$, substituting (3) into (7) yields (6).

■

It is clear the FRC has a similar form as (3). Therefore, with minor modifications, we can apply the cycle elimination algorithm to eliminate low weight codewords.

We summarize our code construction process as follows:

1) Given the size of the mother matrix, select a cage graph of similar size to construct a mother matrix.
2) Identify all the cycles of length $l < l_g$. Transform each cycle to a constraint and add it to the constraint list.
3) Apply the cycle elimination algorithm to search for shifting coefficients $\{a_i\}$ such that all the constraints in this list are satisfied.
4) The remaining cycles that are not eliminated are transformed to constraints and added to the constraint list.

5) Apply the cycle elimination algorithm to search for the power coefficients $\{\rho_i\}$ such that all the constraints in the new list are satisfied.

We apply the procedure above to construct a regular LDPC cycle code over GF(16). The size of the circulant matrix is 15 and the code length is 600 GF(16) symbols. The girth of the code is 16 (in tanner graph). In addition, we also construct another nonbinary LDPC code over GF(16) based on the PEG construction. For the PEG construction, 60.4% of variable nodes have a local girth of 16, 36.3% of variable nodes have a local girth of 14 and 3.3% of variable nodes have a local girth of 12. Both codes will be used in the simulation section.

IV. Parallel sparse encodable nonbinary LDPC cycle codes

In this section we show that, by exploiting the QC structure of nonbinary cycle codes, we can obtain a class of nonbinary LDPC codes that allows not only linear-time sparse encoding from parity check matrix but also efficient parallel implementation. We refer to this class of codes as PSE codes in contrast to the general LDPC codes which are encoded from the dense generator matrices. Performance of the PSE codes proposed here will be examined in Section V.

A. Parity-check matrix based encoding of binary cycle codes

Even though it is well-known that binary cycle codes are linearly encodable [1], here we provide a novel proof for this important fact. The encoding method described in the proof will be extended to the encoding of nonbinary cycle codes in Section IV-B.

Theorem 4.1: Binary cycle codes are linearly encodable.

Proof: Since the normal graph of a binary cycle code, denoted by $N(H)$, must be a union of several connected graph, without loss of generality, it is sufficient to consider a single connected graph G. Assume that H has n columns and m rows, then G has n edges and m vertices. It is well-known that every connected graph contains a *spanning tree*, with any specified vertex as a its root [12]. Starting from an arbitrary vertex c_0 in G, let $\mathrm{Tr}(G)$ denote a spanning tree of G with c_0 as the root. Since G contains m vertices, there must be a total of $m-1$ edges in $\mathrm{Tr}(G)$. Let $b_1, b_2, \cdots, b_{n-m+1}$ denote edges in G but not in $\mathrm{Tr}(G)$. The encoding process proceeds as follows: let $b_1, b_2, \cdots, b_{n-m+1}$ correspond to information bits of the code, then the values of the edges in $\mathrm{Tr}(G)$ that are incident to the leaves can be computed since only one edge is unknown at each leaf. Subsequently, by removing all the edges whose values are previously computed, we obtain a new tree. This way we can compute all the edge values level by level until all the edges incident to c_0 are computed. We claim that the check equation corresponding to c_0 is then automatically satisfied. This is because the summation of all rows in the parity-check matrix of a cycle code equals zero, which means that if all the other $m-1$ checks are satisfied, then the remaining check is also satisfied. In other words, the vertex c_0 is a redundant check for the binary cycle code and removing it does not change the code structure. ■

The proof above shows that the encoding process of binary cycle codes is equivalent to solving the parity-check equations row by row sequentially with a re-arranged order of the rows in H. We refer to this encoding algorithm as sparse encoding. The codes that can be encoded using sparse encoding are called *sparse encodable* codes.

B. Parallel sparse encoding of nonbinary cycle codes

Unfortunately, nonbinary cycle codes are not sparse encodable in general. The proof of Theorem 4.1 shows that the root vertex c_0 must be redundant in order for the code to be sparse encodable. This is not necessarily true for nonbinary codes. Therefore, in order to realize sparse encoding for nonbinary cycle codes, one option is to change the code constraint associated with the root vertex c_0. Based on this idea, we propose a novel sparse encoding method for nonbinary QC LDPC cycle codes. Since this method utilizes the QC structure of the LDPC cycle code to facilitate parallel encoding, we refer to it as parallel sparse encoding. We will show that, starting from any base QC nonbinary LDPC cycle code, we can obtain a PSE code consisting of a QC subcode, modified from the base code, and a simple tree subcode. Simulation results in Section V demonstrate that the resulting parallel sparse code achieves a comparable performance to the base code with much reduced encoding time.

The parallel sparse encoding procedure. Assume that the parity-check matrix of the base QC code, H, with dimensions $m \times n$, is composed of permutation circulant submatrices of dimensions $\zeta \times \zeta$. We first show that the normal graph $N(H)$ consists of ζ disjoint spanning trees that are isomorphic. We build the spanning trees using the ζ vertices corresponding to the first ζ rows of H as roots. Suppose that we have formed ζ disjoint isomorphic trees each with k levels. Then at the $(k+1)$-th level, we first add all the edges that are incident to the leaves to each of the ζ trees. If cycles appear, we will then remove some of these newly added edges while keeping all the new leaves reachable. We say that a vertex in $N(H)$ connects to a circulant submatrix of H if part of its corresponding row belongs to that circulant submatrix. The cycles of H can be categorized as two types: type-I cycle paths that do not cross the vertices connecting to the same circulant submatrix more than twice, and type-II cycle paths that go through several vertices connecting to the same circulant submatrix before returning to the starting vertex. For type-I cycles, we simply remove the same set of edges (under the isomorphism) from each tree. For type-II cycles, we remove the edges that connect different trees to break the connected graph into ζ disjoint trees. Due to the QC structure of the code, the isomorphism among the ζ trees is still kept after adding the new edges. Therefore, we obtain ζ disjoint trees of $(k+1)$ levels. The trees grow in this way until all the vertices have been reached. After the spanning trees are built, we let the edges not included in the trees be the information symbols. Subsequently, we can perform sparse encoding as described in Theorem 4.1 over each disjoint tree in parallel.

The encoding procedure above leads to a modified version of the base QC code. Note that the check equations corresponding to the ζ root vertices are not satisfied after the values

of all edges incident to these vertices are computed. To offer additional protection on the symbols corresponding to these edges, we add a simple tree code on top of the QC code. Hence, the resulting code is a combination of a QC code and a small tree code. Next, we illustrate the parallel sparse encoding method through an example.

Example 4.1: Consider a base QC LDPC cycle code with $n = 18, m = 12$, and $\zeta = 3$. The parity check matrix H is defined by the mother matrix H_m and the cyclic shift matrix $\mathbf{P}$ in (8). Namely, H is obtained by replacing each zero in H_m by a $\zeta \times \zeta$ zero matrix, and replacing each 1 located in the i-th row and j-th column of H_m by a cyclic permutation matrix obtained by shifting a $\zeta \times \zeta$ identity matrix to the right by P_{ij} positions. Here P_{ij} is the (i, j)-th element of $\mathbf{P}$.

$$\mathbf{H}_m = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \mathbf{P} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{pmatrix} \tag{8}$$

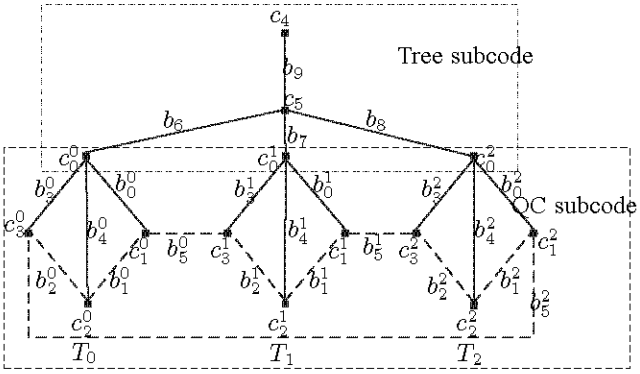

Fig. 2. Normal graph representation of a PSE code

Fig. 2 shows the normal graph of the PSE code based on the QC code defined by H. The top half of the graph corresponds to a two-level tree subcode, and the bottom half corresponds to the QC code. Note that $\{c_i^0, c_i^1, c_i^2\}$ and $\{b_i^0, b_i^1, b_i^2\}$ correspond to equivalent (under the isomorphism) rows and columns. Following the parallel sparse procedure described above, we can identify $\zeta = 3$ disjoint isomorphic spanning trees. The i-th tree consists of vertices $\{c_0^i, c_1^i, c_2^i, c_3^i\}$ with c_0^i as the root and edges $\{b_0^i, b_3^i, b_4^i\}$. In order to form disjoint trees, the remaining edges in the graph, represented by the dash lines, are removed to eliminate cycles. Specifically, $\{b_1^i, b_2^i\}$ are removed to eliminate the type-I cycles $c_0^i \to c_2^i \to c_3^i \to c_0^i$ and $c_0^i \to c_1^i \to c_2^i \to c_0^i$; and b_5^i is removed to eliminate the type-II cycle $c_0^0 \to c_1^0 \to c_3^1 \to c_0^1 \to c_1^1 \to c_3^2 \to c_0^2 \to c_1^2 \to c_3^0 \to c_0^0$. These 9 removed edges correspond to information symbols, from which the values of coded symbols can be computed. Note that without the tree subcode, the three root vertices $\{c_0^0, c_0^1, c_0^2\}$ are not necessarily satisfied. Hence, we add four additional coded symbols $\{b_6, b_7, b_8, b_9\}$ and checks $\{c_4, c_5\}$ to offer stronger protection. The values of these additional coded symbols are also computed from the bottom to the top. The resulting PSE code has a rate of $9/22 = 0.409$.

C. Encoding complexity of sparse encodable codes

As discussed above, the sparse encoding solves row equations of H sequentially. Its overall complexity is md_c multiplications and $m(d_c - 1)$ additions over GF(q), where d_c is the degree of check nodes. Hence, the encoding process requires $m[d_c T_1 + (d_c - 1)T_2]$ clock cycles, where T_1, T_2 are the clock cycles required for a multiplication and an addition over GF(q), respectively. For parallel sparse encoding, the overall encoding time is further reduced to $(m/\zeta)[d_c T_1 + (d_c - 1)T_2]$ for the encoding of QC subcode, plus the encoding time of the tree subcode, which is typically much smaller than the encoding time of the QC subcode. Compared to the generator matrix based encoding scheme, which has a complexity of $O(n^2)$, the parallel sparse encoding schemes achieve significant complexity saving since H is a sparse matrix. In [3], Lin *et al.* proposes an encoder for QC LDPC codes which utilizes the QC structure to reduce the density of the generator matrix with encoding time of $\zeta[((n-m)/\zeta + 1)T_1 + ((n-m)/\zeta)T_2]$ clock cycles. In general, since d_c is typically much smaller than ζ, when the code rate is not too low, we will have $d_c < (n/m - 1)\zeta$ so that the proposed parallel sparse encoder has a much lower complexity and shorter encoding time than the encoder in [3]. For instance, for a PSE code constructed from a rate $1/2$ QC GF(16) cycle code with $d_c = 4$, $n = 600$, and $\zeta = 15$, the parallel sparse encoder saves about 60% in complexity and encoding time compared to the encoder in [3]. In [13], a modified PEG algorithm is proposed to construct sparse encodable codes. However, since the codes constructed in [13] are not QC, parallel encoding is not applicable which results in a higher encoding complexity than the proposed parallel sparse encoder.

V. Numerical Results

In this section, we examine the performance of LDPC cycle codes in MIMO channels. We first compare the performance of the proposed nonbinary QC cycle codes constructed from the CE algorithm with a code constructed from the PEG algorithm, which is not QC, and with a QC code constructed from the QPP algorithm [14]. Then we provide a performance comparison between nonbinary coded systems employing the PSE codes and binary coded systems employing an optimized binary LDPC code or a QC code in the 802.16e standards.

We have constructed three codes. The first code, labeled as CE I, is constructed by applying CE only once to find good shift coefficients. The second code, labeled as CE II, is constructed by applying CE twice to find both good shift coefficients and nonzero elements. Both codes have rate 1/2 and code length 600 symbols. From CE II, we construct a PSE code with 624 coded symbols and 315 information symbols. It consists a QC subcode and a 4-level tree subcode. We label it as CE III.

Fig. 3 presents the bit-error-rate (BER) and block-error-rate (BLER) performance curves of CE I, CE II codes for a MIMO channel with two transmit and two receive antennas. The channel matrix has i.i.d. Rayleigh entries and is independent over time. The 16 QAM modulation is used. Performance curves of the nonbinary QC code constructed from the QPP

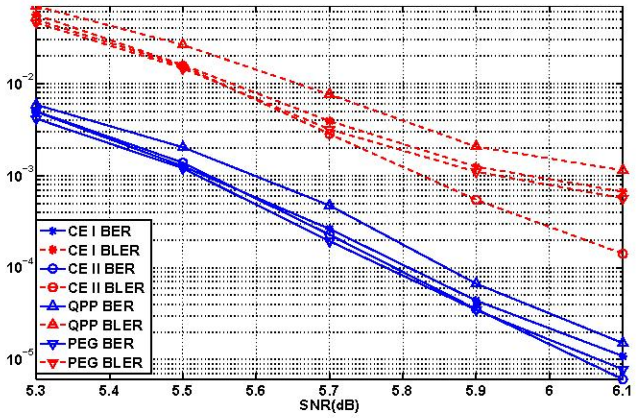

Fig. 3. Performance comparisons of LDPC cycles codes over GF(16) based on CE, QPP, and PEG constructions

algorithm and the nonbinary code constructed from the PEG algorithm are also shown. To ensure the accuracy of numerical results, we collect at least 100 error blocks for each point in the performance curve. Fig. 3 shows that the CE I code performs closely to the PEG code. Both codes have relatively high error floor. In contrast, the CE II code has a significantly lower error floor and it achieves over 0.2 dB gain in BLER at high SNR. The QC code constructed using the QPP algorithm has the worst performance and the highest error floor. This justifies the effectiveness of our construction algorithm.

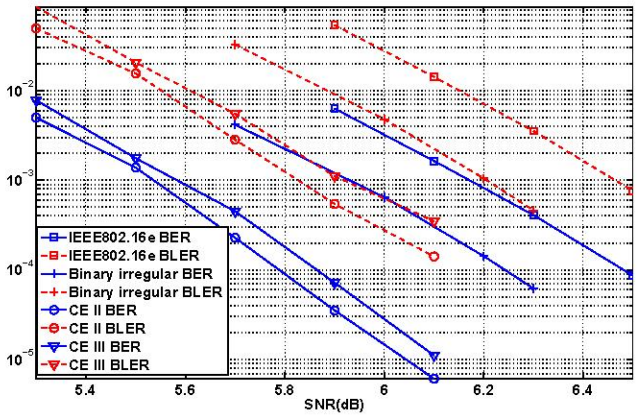

Fig. 4. Performance comparisons nonbinary and binary coded system

In Fig. 4, we compare the performance of nonbinary and binary LDPC coded systems. For the binary coded system we use an irregular code whose degree distribution is optimized for MIMO channel using EXIT chart [5]. Since irregular codes are usually harder to implement in practice, we also consider a QC code which is included in an IEEE802.16e standard that has lower encoding and decoding complexity. Both codes have a code length of 2304. Fig. 4 shows that, while the CE III code has a much lower encoding complexity as discussed in Section IV-C, it has only about 0.1dB performance degradation compared to the CE II code. When comparing with binary coded system, at BER $= 10^{-4}$, the CE III code performs about 0.38 dB and 0.6 dB better than the best optimized irregular code and the IEEE802.16 code respectively.

VI. Conclusion

In this paper, we propose a class of nonbinary LDPC cycle codes for MIMO channels which demonstrates superior performance than the best optimized binary LDPC code. By exploiting the QC structure of nonbinary cycle codes, a novel parallel sparse encoding method is developed to facilitate parallel implementation in addition to linear-time encoding. The cycle elimination algorithm is applied to the code construction to remove the short cycles and to choose the nonzero elements in the parity-check matrix such that the number of low weight codewords can be reduced. The codes constructed using the proposed approach achieve lower error floors than those constructed using the PEG algorithm and the QPP algorithm. In [15], we also show that the proposed nonbinary coded system has a lower receiver complexity than that of the binary coded system. Therefore, we conclude that the proposed PSE nonbinary codes are good candidates for MIMO channels in both performance and complexity.

References

[1] S. Hakimi and J. Bredeson, "Graph theoretic error-correcting codes," *IEEE Trans. Inform. Theory*, vol. 14, no. 4, pp. 584–591, Jul 1968.

[2] X.-Y. Hu and E. Eleftheriou, "Binary representation of cycle Tanner-graph GF(2/sup b/) codes," in *Proc. IEEE Inter. Conf. on Communication (ICC'04)*, vol. 1, June 20-24 2004, pp. 528–532.

[3] Z. Li, L. Chen, L. Zeng, S. Lin, and W. H. Fong, "Efficient encoding of quasi-cyclic low-density parity-check codes," *IEEE Trans. Commun.*, vol. 54, no. 1, pp. 71–81, Jan. 2006.

[4] S. Lin, S. Song, L. Lan, L. Zeng, and Y. Y. Tai, "Constructions of nonbinary quasi-cyclic LDPC codes: a finite field approach," in *Proc. Information Theory and Applications (ITA) Workshop*, UCSD, 2006.

[5] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Trans. Commun.*, vol. 52, pp. 670–678, Apr. 2004.

[6] *IEEE Std 802.16e-2005*, approved Dec 2005, pub. Feb 2006.

[7] J. Forney, G.D., "Codes on graphs: normal realizations," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 520–548, Feb 2001.

[8] G. Royle, "Cages of higher valency," http://people.csse.uwa.edu.au/gordon/cages/allcages.html.

[9] S. Myung, K. Yang, and J. Kim, "Quasi-cyclic LDPC codes for fast encoding," *IEEE Trans. Inform. Theory*, vol. 51, no. 8, pp. 2894–2901, Aug. 2005.

[10] L. Yang, H. Liu, and C.-J. Shi, "Code construction and FPGA implementation of a low-error-floor multi-rate low-density parity-check code decoder," *IEEE Trans. Circuits and Systems I*, vol. 53, no. 4, pp. 892–904, April 2006.

[11] C. Poulliat, M. Fossorier, and D. Declercq, "Using binary images of non binary LDPC codes to improve overall performance," in *4th International Symposium on Turbo-codes and related topics*, Munich, Germany, April 2006.

[12] R. Diestel, *Graph theory*, 3rd ed., S. Axler and K. Ribet, Eds. Springer, 2005.

[13] X. Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth Tanner graphs," *IEEE Trans. Inform. Theory*, vol. 51, pp. 386–398, Jan. 2005.

[14] R. Peng and R.-R. Chen, "Application of nonbinary LDPC codes for communication over fading channels using higher order modulations," in *Proc. IEEE Global Telecommunications Conference (GLOBECOM'06)*, Nov. 2006.

[15] ——, "Application of nonbinary LDPC cycle codes to MIMO Channels," submitted to *IEEE Trans. Wireless Comm.*