Structure, Property, and Design of Nonbinary Regular Cycle Codes

Jie Huang, Shengli Zhou, *Member, IEEE,* and Peter Willett, *Fellow, IEEE*

Abstract—In this paper, we study nonbinary regular LDPC cycle codes whose parity check matrix H has fixed column weight $j = 2$ and fixed row weight d. Through graph analysis, we show that the parity check matrix H of a regular cycle code can be put into an equivalent structure in the form of concatenation of row-permuted block-diagonal matrices if d is even, or, if d is odd and the code's associated graph contains at least one spanning subgraph that consists of disjoint edges. This equivalent structure of H enables: i) parallel processing in linear-time encoding; ii) considerable resource reduction on the code storage for encoding and decoding; and iii) parallel processing in sequential belief-propagation decoding, which increases the throughput without compromising performance or complexity. On the code's structure design, we propose a novel design methodology based on the equivalent structure of H. Finally, we present various numerical results on the code performance and the decoding complexity.

Index Terms—LDPC, regular cycle code, Galois field, graph theory, decoding algorithm, code design.

I. INTRODUCTION

BINARY low-density parity-check (LDPC) codes which are proposed by Gallager [2] are excellent error-correcting codes that achieve performance close to the benchmark predicted by the Shannon capacity [3]. The extension of LDPC codes to nonbinary Galois field GF(q) was first investigated empirically by Davey and MacKay over the binary-input AWGN channel [4]. Since then, nonbinary LDPC codes have been actively studied.

In this paper, we focus on the LDPC codes with column weight $j = 2$ in their parity check matrix **H**, termed as cycle codes [5]. Although the distance properties of binary cycle codes are not as good as the LDPC codes of column weight $j \geq 3$ [2], it has been shown in [6] that nonbinary cycle codes over GF(q) can achieve near-Shannon-limit performance as q increases. Further, numerical results in [6] demonstrate that nonbinary cycle codes can outperform other LDPC codes, including degree-distribution-optimized binary irregular LDPC codes. For high order fields $q \geq 64$, the best GF(q)-LDPC codes decoded by belief propagation (BP) should be *ultra sparse* [4], [7], with a good example being the cycle codes

that have $j = 2$. A Fast-Fourier-Transform based q-ary sum-product algorithm (FFT-QSPA) for decoding LDPC codes over binary extension fields has been proposed in [8], [9]. A universal linear-complexity encoding algorithm for any nonbinary cycle code is available in [10]. With the performance and implementation advantages, nonbinary cycle codes are very promising for practical applications.

In this paper we study a special class of cycle codes whose parity check matrix **H** has fixed column weight 2 and fixed row weight d. We call such LDPC code a regular cycle code. One popular representation of LDPC codes is based on Tanner-graph [11]. A more compact representation for cycle codes is an associated graph G with m vertices and n edges, where each vertex represents one check node corresponding to one row of **H**, and each edge represents one variable node corresponding to one column of **H**. If the row weight of **H** for a cycle code is fixed as d, i.e., the code is a regular cycle code, then each vertex of G is connected exactly to d edges. Such graph is d-regular [12]. For any d-regular cycle code, we show that its parity check matrix can be arranged through row and column permutations into an equivalent structure in the form of *concatenation of row-permuted block-diagonal matrices* if d is even, or, if d is odd and the code's associated graph G contains at least one spanning subgraph that consists of disjoint edges.

This equivalent structure leads to several promising properties. First, encoding for nonbinary regular cycle codes can be performed *in parallel* in linear time. Second, the storage requirement for **H** can be greatly reduced, which is useful for both encoding and decoding. In addition, this structure enables *parallel processing* in sequential BP decoding for nonbinary regular cycle codes, which improves the decoding throughput considerably without compromising performance or complexity.

The design of nonbinary regular cycle codes consists of code structure design and selection of nonzero entries of **H**. On the code structure design, one can design the code based on known graphs with large girth, such as the Ramanujan and Cayley graphs as done in [6], [13]. Or, one can rely on computer search based algorithms, such as the well-known progressive edge-growth (PEG) algorithm [14]. The research on the selection of nonzero entries has been widely pursued [13], [15]–[17]. In this paper, we propose a novel method that amounts to designing directly the interleavers in the equivalent structure of **H** developed herein.

Simulations are carried out to evaluate the performance and the decoding complexity of nonbinary regular cycle codes.

Our simulations show that nonbinary regular cycle codes constructed by the PEG algorithm can outperform binary degree-distribution-optimized LDPC codes. In addition, we show that very-high-rate (rate 8/9 and 15/16) nonbinary regular cycle codes with block length about several thousands of bits can approach the Shannon limit within 1 dB. Also, the proposed sequential BP decoding with parallel processing can reduce the total decoding complexity by about 30 percent with a slightly better performance than the standard BP decoding. Compared with codes constructed from know good graphs and by the PEG algorithm, regular cycle codes with widely varying rates constructed by the proposed method based on the equivalent structure can achieve similar performance.

Some of the structure and property results have been presented in our earlier work [13] for a special class of regular cycles codes constructed from Cayley graphs. The distinctions of this paper from [13] are as follows: i) the *group*-theoretic analysis adopted in [13] works only for codes based on Cayley graphs, hence the coverage of [13] is quite limited. In this paper, we rely on *graph*-theoretic analysis, and the obtained results are applicable to a general regular cycle code; and ii) the results on the sequential BP decoding with parallel processing and the structure design of regular cycle codes are not available in [13]. In addition, this paper provides extensive simulation results on the code performance, with some of them on very-high-rate codes.

The rest of the paper is organized as follows. Section II presents the equivalent structure of $\mathbf{H}$ for regular cycle codes using graph-theoretic analysis. Section III specifies the promising properties of the code structure presented in Section II. Utilizing the structure results in Section II, we present a novel method to design the code structure for nonbinary regular cycle codes in Section IV. Extensive simulation results are presented in Section V. We draw conclusions in Section VI.

II. STRUCTURE OF NONBINARY REGULAR CYCLE CODES

A cycle code is an LDPC code whose $m \times n$ parity check matrix $\mathbf{H}$ has weight $j = 2$ for each column. As such, it can be represented by an associated graph $G = (V, E)$ with m vertices $V = \{v_1, \ldots, v_m\}$ and n edges $E = \{e_1, \ldots, e_n\}$, where each vertex represents a check node corresponding to a row of $\mathbf{H}$, and each edge represents a variable node corresponding to a column of $\mathbf{H}$ [10]. See Figs. 1(a) and 1(b) for an illustration.

For a regular cycle code with fixed row weight d in $\mathbf{H}$, the graph G is d-regular in that each vertex is exactly linked to d edges [12]. Obviously we have $2n = dm$. When $\mathbf{H}$ is full row-rank with elements from $\mathrm{GF}(q)$, $\mathbf{H}$ defines a nonbinary regular cycle code of rate $r = (d - 2)/d$.

In this paper, we use graph theory to analyze regular cycle codes. We first introduce two necessary definitions from [12].
Definitions:
- k-**factor**: A k-regular spanning subgraph of G that contains all the vertices is called a k-factor of G.
- k-**factorable**: a graph G is k-factorable if there are edge-disjoint k-factors $G_1, G_2, \ldots, G_L$ such that $G = G_1 \cup G_2 \cdots \cup G_L$.

Obviously, a 1-factor is a spanning subgraph that consists of disjoint edges, while a 2-factor is a spanning subgraph that

consists of disjoint cycles. See illustrations in Figs. 1(c) and 1(d).

For a subgraph G' of G, let $\mathbf{H}_{G'}$ be the sub-matrix of $\mathbf{H}$ restricted to the rows and columns indexed by the vertices and edges of G' respectively, which can be obtained from $\mathbf{H}$ by deleting the rows and columns other than those corresponding to the vertices and edges of G' respectively. We call $\mathbf{H}_{G'}$ the sub-matrix of $\mathbf{H}$ associated with G'. Let us now introduce two sub-matrices of $\mathbf{H}$ associated with an edge and a cycle of the graph G. For each edge, the sub-matrix is

$$\tilde{\mathbf{h}}^e = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \tag{1}$$

where α and β correspond to those two nonzero entries of the column of $\mathbf{H}$ indexed by this edge.

For a length-k cycle C that consists of k consecutive edges $e_1, e_2, \ldots, e_k$, we can define a $k \times k$ matrix as

$$\tilde{\mathbf{H}}^c = \begin{bmatrix} \alpha_1 & 0 & 0 & \ldots & \beta_k \\ \beta_1 & \alpha_2 & 0 & \ldots & 0 \\ 0 & \beta_2 & \alpha_3 & \ldots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & \ldots & 0 & \beta_{k-1} & \alpha_k \end{bmatrix}, \tag{2}$$

where α_is and β_is correspond to those two nonzero entries of the column of $\mathbf{H}$ indexed by edge e_i.

For two matrices $\mathbf{H}_1$ and $\mathbf{H}_2$, if $\mathbf{H}_1$ can be transformed into $\mathbf{H}_2$ simply through row and column permutations, we deem $\mathbf{H}_1$ equivalent to $\mathbf{H}_2$ and denote this relationship as $\mathbf{H}_1 \cong \mathbf{H}_2$.

A. Main results

Our main results are the following two theorems.

Theorem 1 *For a cycle code, if its associated graph G is d-regular with $d = 2\nu$, then its parity check matrix $\mathbf{H}$ of size $m \times n$ has the equivalent form*

$$\mathbf{H} \cong [\bar{\mathbf{H}}_1, \mathbf{P}_2 \bar{\mathbf{H}}_2, \ldots, \mathbf{P}_\nu \bar{\mathbf{H}}_\nu], \tag{3}$$

where $\mathbf{P}_i$ is $m \times m$ permutation matrix, and $\bar{\mathbf{H}}_i$ is of size $m \times m$, $1 \leq i \leq \nu$. The matrix $\bar{\mathbf{H}}_i$ has an equivalent block-diagonal form

$$\bar{\mathbf{H}}_i \cong \mathrm{diag}(\tilde{\mathbf{H}}_{i,1}^c, \tilde{\mathbf{H}}_{i,2}^c, \ldots, \tilde{\mathbf{H}}_{i,L_i}^c), \tag{4}$$

where the matrix $\tilde{\mathbf{H}}_{i,l}^c$ has the form of (2) and is of size $k_{i,l} \times k_{i,l}$ that satisfies $m = \sum_{l=1}^{L_i} k_{i,l}$.

Proof of Theorem 1: If G is d-regular with $d = 2\nu$, $\nu > 0$, G is 2-factorable as can be inferred from Corollary 2.1.5 of [12, p.33] (also known as the Konig-Hall theorem in graph theory). Denote the ν edge-disjoint 2-factors of G by $G_1, G_2, \ldots, G_\nu$. Arrange the columns of $\mathbf{H}$ in such a pattern that the columns indexed by the edges of G_1 are placed in the first m columns, followed by the m columns indexed by the edges of G_2 until the m columns which are indexed by the edges of G_ν. This way, $\mathbf{H}$ is partitioned to ν sub-matrices of size $m \times m$ each. Arranged as $\mathbf{H} \cong [\mathbf{H}_{G_1}, \ldots, \mathbf{H}_{G_\nu}]$, where $\mathbf{H}_{G_i}$ is the sub-matrix of $\mathbf{H}$ associated with G_i.

Now we show that each $m \times m$ sub-matrix $\mathbf{H}_{G_i}$ has an equivalent block diagonal form as in (4). Each 2-factor G_i can be decomposed into a set of disjoint cycles. Suppose G_i consists of L_i disjoint cycles $C_{i,l}$, $1 \leq l \leq L_i$, where $C_{i,l}$ is of length $k_{i,l}$ that satisfy $m = \sum_{l=1}^{L_i} k_{i,l}$. Arrange the rows and columns of $\mathbf{H}_{G_i}$ in sequence of rows and columns indexed by $C_{i,1}, C_{i,2}, \ldots, C_{i,L_i}$, the resultant matrix will have a block-diagonal form $\operatorname{diag}(\tilde{\mathbf{H}}_{i,1}^c, \tilde{\mathbf{H}}_{i,2}^c, \ldots, \tilde{\mathbf{H}}_{i,L_i}^c)$, where $\tilde{\mathbf{H}}_{i,l}^c$ represents the matrix associated with $C_{i,l}$ and has a form as in (2). Thus we have $\mathbf{H}_{G_i} = \mathbf{P}_i \bar{\mathbf{H}}_i \mathbf{R}_i$, where $\bar{\mathbf{H}}_i$ is defined in (4), and $\mathbf{P}_i$ and $\mathbf{R}_i$ are permutation matrices, $1 \leq i \leq \nu$.

Therefore, the matrix $\mathbf{H}$ can be arranged to have an equivalent form $[\mathbf{P}_1 \bar{\mathbf{H}}_1 \mathbf{R}_1, \mathbf{P}_2 \bar{\mathbf{H}}_2 \mathbf{R}_2, \ldots, \mathbf{P}_\nu \bar{\mathbf{H}}_\nu \mathbf{R}_\nu]$, We can further permute the rows of $\mathbf{H}$ to let $\mathbf{P}_1$ be the identity matrix and permute the columns of $\mathbf{H}_{G_i}$ to let each $\mathbf{R}_i$ be the identity matrix. The resultant matrix would have a form like (3). This completes the proof. ∎

Theorem 2 *Consider a regular cycle $GF(q)$ code with $d = 2\nu+1$. If its associated graph G contains at least one 1-factor, then its parity check matrix $\mathbf{H}$ of size $m \times n$ has the equivalent form*

$$\mathbf{H} \cong [\bar{\mathbf{H}}_1, \mathbf{P}_2 \bar{\mathbf{H}}_2, \ldots, \mathbf{P}_\nu \bar{\mathbf{H}}_\nu, \mathbf{P}^e \bar{\mathbf{H}}^e] \quad (5)$$

where $\mathbf{P}_i s$ and $\mathbf{P}^e$ are permutation matrices, $\bar{\mathbf{H}}_i$ is an $m \times m$ block-diagonal matrix having the form as in (4), $i = 1, \ldots, \nu$, $\bar{\mathbf{H}}^e$ is an $m \times \frac{m}{2}$ matrix having an equivalent block-diagonal form as

$$\bar{\mathbf{H}}^e \cong \operatorname{diag}(\tilde{\mathbf{h}}_1^e, \tilde{\mathbf{h}}_2^e, \ldots, \tilde{\mathbf{h}}_{\frac{m}{2}}^e), \quad (6)$$

where $\tilde{\mathbf{h}}_i^e$ is a vector having the form as in (1).

Proof of Theorem 2: If G is d-regular with $d = 2\nu + 1$, $\nu > 0$ and G has a 1-factor M, let G' denote the graph obtained from G by deleting the edges in M. So G' is 2ν-regular. Arrange the columns of $\mathbf{H}$ in such a pattern that the columns indexed by the edges of G' are placed in the first νm columns, followed by the $m/2$ columns which are indexed by the edges of M. Arranged as $\mathbf{H} \cong [\mathbf{H}_{G'}, \mathbf{H}_M]$, where $\mathbf{H}_{G'}$ is the sub-matrix of $\mathbf{H}$ associated with G' and $\mathbf{H}_M$ is the sub-matrix of $\mathbf{H}$ associated with M.

Applying Theorem 1, the sub-matrix $\mathbf{H}_{G'}$ has a form in (3). Now we show the form of sub-matrix $\bar{\mathbf{H}}_M$. Since M is a 1-factor of G, M is an union of disjoint edges. Denote the edges of M by e_i, $1 \leq i \leq m/2$. Arrange the rows and columns of $\mathbf{H}_M$ in sequence of rows and columns indexed by $e_1, e_2, \ldots, e_{m/2}$, the resultant matrix will have the form in (6). Thus we have $\mathbf{H}_M = \mathbf{P}^e \bar{\mathbf{H}}^e \mathbf{R}^e$, where $\bar{\mathbf{H}}^e$ is defined in (6), and $\mathbf{P}^e$ and $\mathbf{R}^e$ are permutation matrices.

Therefore, the matrix $\mathbf{H}$ would have an equivalent form like $[\bar{\mathbf{H}}_1, \mathbf{P}_2 \bar{\mathbf{H}}_2, \ldots, \mathbf{P}_\nu \bar{\mathbf{H}}_\nu, \mathbf{P}^e \bar{\mathbf{H}}^e \mathbf{R}^e]$, where $\mathbf{P}^e$, $\mathbf{R}^e$ and $\mathbf{P}_i s$, $2 \leq i \leq \nu$, are permutation matrices. Furthermore, we can permute the columns of $\mathbf{H}_M$ to let $\mathbf{R}^e$ be the identity matrix. The resultant matrix would have a form like (5). This completes the proof. ∎

B. Further Results

In addition to the main results in Section II-A, we have some further results for 1-factorable graphs and bipartite graphs.

Theorem 3 *For a cycle code, if its associated graph G is d-regular and 1-factorable, then its parity check matrix $\mathbf{H}$ of size $m \times n$ has the equivalent form*

$$\mathbf{H} \cong [\bar{\mathbf{H}}_1^e, \mathbf{P}_2^e \bar{\mathbf{H}}_2^e, \ldots, \mathbf{P}_d^e \bar{\mathbf{H}}_d^e] \quad (7)$$

where $\bar{\mathbf{H}}_i^e$, $1 \leq i \leq d$, are $m \times \frac{m}{2}$ matrices having an equivalent form as in (6), and $\mathbf{P}_i^e$, $2 \leq i \leq d$, are permutation matrices.

Proof of Theorem 3: Denote the d 1-factors of G by $M_1, M_2, \ldots, M_d$. Arrange the columns of $\mathbf{H}$ in such a pattern that the columns indexed by edges of M_1 are placed in the first $m/2$ columns, followed by the $m/2$ columns indexed by the edges of M_2 until the $m/2$ columns which are indexed by the edges of M_d. Thus we have $\mathbf{H} \cong [\mathbf{H}_{M_1}, \mathbf{H}_{M_2}, \ldots, \mathbf{H}_{M_d}]$, where $\mathbf{H}_{M_i}$ is the sub-matrix of $\mathbf{H}$ associated with M_i. As we have shown in the proof of Theorem 2, the sub-matrix $\mathbf{H}_{M_i}$ will have an equivalent form as in (6), that is, $\mathbf{H}_{M_i} = \mathbf{P}_i^e \bar{\mathbf{H}}_i^e \mathbf{R}_i^e$, where $\bar{\mathbf{H}}_i^e$ has an equivalent form as in (6), $\mathbf{P}_i^e$ and $\mathbf{R}_i^e$ are permutation matrices. Therefore the matrix $\mathbf{H}$ would have an equivalent form like $[\mathbf{P}_1^e \bar{\mathbf{H}}_1^e \mathbf{R}_1^e, \mathbf{P}_2^e \bar{\mathbf{H}}_2^e \mathbf{R}_2^e, \ldots, \mathbf{P}_d^e \bar{\mathbf{H}}_d^e \mathbf{R}_d^e]$, where $\mathbf{P}_i^e$ and $\mathbf{R}_i^e$, $1 \leq i \leq d$ are permutation matrices. Furthermore, we can permute the rows of $\mathbf{H}$ to let $\mathbf{P}_1^e$ be the identity matrix and permute the columns of $\mathbf{H}_{M_i}$ to let each $\mathbf{R}_i^e$ be the identity matrix. The resultant matrix would have a form like (7). This completes the proof. ∎

We now present the results for bipartite graphs. A graph $G = (V, E)$ is called bipartite if V admits a partition into two classes X and Y such that every edge in G has exactly one end in X and one end in Y: vertices in the same partition class must not be adjacent. To this end, we define a q-ary permutation matrix $\mathbf{Q}$ as a matrix which contains one and only one nonzero element from $GF(q)$ in each row and each column. Any q-ary permutation matrix $\mathbf{Q}$ can be represented as $\mathbf{Q} = \mathbf{P}\mathbf{D} = \mathbf{D}'\mathbf{P}$, where $\mathbf{D}, \mathbf{D}'$ are diagonal matrices with nonzero elements from $GF(q)$ and $\mathbf{P}$ is a permutation matrix obtained from $\mathbf{Q}$ by replacing each nonzero element of $\mathbf{Q}$ by 1.

Theorem 4 *For a cycle code, if its associated graph G is d-regular and bipartite, then its parity check matrix $\mathbf{H}$ of size $m \times n$ has the equivalent form*

$$\mathbf{H} \cong \begin{bmatrix} \mathbf{D}_1 & \mathbf{D}_2 & \cdots & \mathbf{D}_d \\ \mathbf{D}_1' & \mathbf{P}_2' \mathbf{D}_2' & \cdots & \mathbf{P}_d' \mathbf{D}_d' \end{bmatrix} \quad (8)$$

where $\mathbf{D}_i$ and $\mathbf{D}_i'$, $1 \leq i \leq d$, are diagonal matrices of size $\frac{m}{2} \times \frac{m}{2}$ with nonzero elements from $GF(q)$, and $\mathbf{P}_i'$, $2 \leq i \leq d$, are permutation matrices of size $\frac{m}{2} \times \frac{m}{2}$.

Proof of Theorem 4: Let $V = X \cup Y$ be a bipartition of G. Since G is d-regular and bipartite, G is 1-factorable which is a direct consequence of the well-known Hall's *marriage theorem* when applied to regular bipartite graphs;

see e.g., Theorem 2.1.2 of [12]. Denote the d 1-factors of G by $M_1, M_2, \ldots, M_d$. As shown in the proof of Theorem 3 we have $\mathbf{H} \cong [\mathbf{H}_{M_1}, \mathbf{H}_{M_2}, \ldots, \mathbf{H}_{M_d}]$, where $\mathbf{H}_{M_i}$ is the sub-matrix of $\mathbf{H}$ associated with M_i. Arrange the rows of $\mathbf{H}$ in such a pattern that the rows corresponding to vertices in X are placed in the upper $m/2$ rows and the rows corresponding to vertices in Y are placed in the lower $m/2$ rows. Since the $\frac{m}{2}$ edges in M_i are disjoint and each edge of M_i has one end in X and one end in Y, so $\mathbf{H}_{M_i}$ will have a form like $\mathbf{H}_{M_i} = \begin{bmatrix} \mathbf{Q}_i \\ \mathbf{Q}_i' \end{bmatrix}$, where $\mathbf{Q}_i$ and $\mathbf{Q}_i'$ are q-ary permutation matrices of size $\frac{m}{2} \times \frac{m}{2}$. The matrix $\mathbf{H}$ will have a form like $\begin{bmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 & \ldots & \mathbf{Q}_d \\ \mathbf{Q}_1' & \mathbf{Q}_2' & \ldots & \mathbf{Q}_d' \end{bmatrix}$. We can permute the upper (lower, respectively) $\frac{m}{2}$ rows of $\mathbf{H}$ to make $\mathbf{Q}_1$ ($\mathbf{Q}_1'$, respectively) diagonal. Further, we can permute the columns of $\mathbf{H}_{M_i}$ to make $\mathbf{Q}_i$, $2 \le i \le d$, diagonal. Since any q-ary permutation matrix $\mathbf{Q}$ can be represented as $\mathbf{Q} = \mathbf{PD}$ where $\mathbf{P}$ is a permutation matrix and $\mathbf{D}$ is a diagonal matrix with nonzero elements from $\mathrm{GF}(q)$, so the resultant matrix will have a form like (8). This completes the proof. ∎

To summarize, we have the following results for a regular cycle code with associated graph G.

1) If G is d-regular with $d = 2\nu$, $\nu > 0$, we can apply Theorem 1. If G is also 1-factorable, then we can apply Theorem 3.
2) If G is d-regular with $d = 2\nu + 1$, $\nu > 0$, and G has at least one 1-factor, we can apply Theorem 2. If G is also 1-factorable, then we can apply Theorem 3.
3) If G is d-regular and bipartite (which implies that G is 1-factorable), we can further apply Theorem 3 and Theorem 4.

III. PROPERTIES OF NONBINARY REGULAR CYCLE CODES

Based on the structures presented in Section II, we next describe several appealing properties of nonbinary regular cycle codes on the encoding, storage requirements and decoding aspects.

- First, encoding of nonbinary regular cycle codes can be done in linear time in parallel similar to [13]. This provides a lot of flexibility in the implementation of efficient encoders which is quite desirable especially when the codeword length is large. In contrast, the universal linear-time encoding algorithm presented in [10] can work only in *a serial manner*.
- Second, the storage cost for $\mathbf{H}$ matrix contains two parts. One part corresponds to the nonzero entries of $\mathbf{H}$. The other part corresponds to the structural information for $\mathbf{H}$ which we term as the structural storage cost. Compared with a general nonbinary cycle code having the same code rate and length, the amount of reduction in structural storage cost for a nonbinary regular cycle code can be more than 50 percent [13].
- Third, parallel processing can be applied in sequential BP decoding for the nonbinary regular cycle codes which increases throughput without compromising the performance and complexity.

The descriptions on the encoding and storage advantages can be found in [13], although they were presented for the special class of codes based on Cayley graphs. We next present the decoding aspect for a general regular cycle code in details.

A. Overview on Belief Propagation Updating Schedules

Iterative decoding based on belief propagation (BP) [3], [18] has received significant attention recently, mostly due to its near-Shannon-limit error performance for the decoding of LDPC codes [2] and turbo codes [19]. It works on the code's Tanner-graph [11] or factor graph [18] in an iterative manner through exchange of soft information. As for LDPC codes, there exist two kinds of processing unit: variable node processing units and check node processing units corresponding to variable nodes and check nodes respectively, and two kinds of message are exchanged between variable nodes and check nodes during iterations: variable-to-check messages and check-to-variable messages; see details in e.g., [3],[4], [8], [9] and [20]. Here we consider three different updating schedules for BP decoding of LDPC codes.

- *Parallel updating:* Each iteration contains a horizontal step followed by a vertical step. At the horizontal step, all check nodes update in parallel the output check-to-variable messages using the input variable-to-check messages. At the vertical step, all variable nodes update in parallel the output variable-to-check messages using the input check-to-variable messages. The updating schedule for standard BP is inherently fully parallel.
- *Sequential updating:* Most recently, a sequential version of the standard BP is proposed to speed up the convergence of BP decoding which is termed as shuffled BP in reference [20] and sequential updating schedule in reference [21]. The updating schedule for sequential BP is totally sequential. In each iteration, the horizontal step and vertical step process jointly, but in a column-by-column manner. It has been shown through simulations that the average number of iterations of the sequential BP algorithm can be about half that of the parallel BP algorithm, where parallel BP and sequential BP decoding achieve similar error performance [20]–[22]. The complexity per iteration for both algorithms is similar, resulting in a lower total complexity for the sequential BP algorithm [20], [21].
- *Partially parallel updating:* To decrease the decoding delay of the sequential BP and preserve the parallelism advantages of the parallel BP, a partially parallel decoding scheme named "group shuffled BP" is developed in [20]. In the group shuffled BP algorithm, the columns of $\mathbf{H}$ are divided into a number of groups. In each group, the updating of messages is processed in parallel, but the processing of groups remains sequential. Group shuffled BP (partially parallel BP) algorithm offers better throughput/complexity tradeoffs in the implementation of efficient decoders.

With respect to the sequential BP algorithm, if there are consecutive columns of $\mathbf{H}$ which are orthogonal to each other, i.e., no two columns intersect at a common row, then the updating for these columns can be carried out simultaneously.
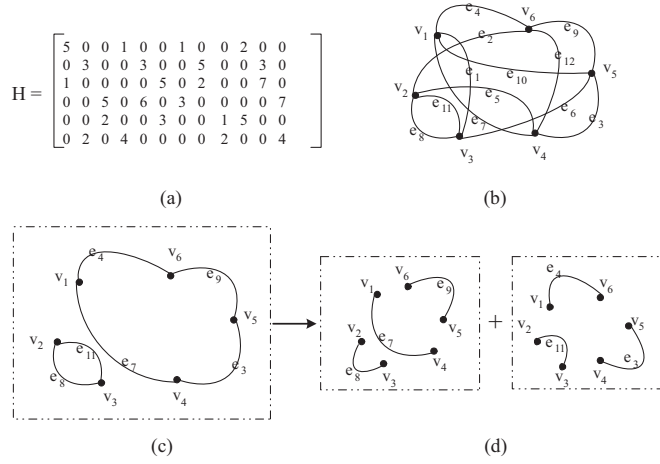
(a)   (b)

(c)   (d)

Fig. 1. Illustration of the concepts of associated graph, 2-factor and 1-factor. (a) a check matrix over GF(8) with column weight $j = 2$ and row weight $d = 4$; (b) the associated graph; (c) one 2-factor of the associated graph; and (d) two 1-factors split from the 2-factor in (c).

(a)   (b)

Fig. 2. Group of edges for sequential BP decoding with parallel processing.

By performing updating for consecutive orthogonal columns simultaneously, we can improve the throughput of sequential BP algorithm without any penalty in error performance or total decoding complexity. We denote this algorithm as sequential BP decoding with parallel processing. This is analogous in principle to a partially parallel BP algorithm where the columns in each group are orthogonal.

B. Sequential BP With Parallel Processing for Nonbinary Regular Cycle Codes

For a nonbinary cycle code, a collection of columns of $\mathbf{H}$ are orthogonal if and only if their corresponding edges in its associated graph G are independent. With the structures presented in Section II, it is easy to find orthogonal columns for nonbinary regular cycle codes.

We first present the following facts.

- The columns of $\mathbf{H}$ corresponding to edges of a 1-factor of G are orthogonal.
- If every component of a 2-factor is an even cycle, we call it an even 2-factor. If a 2-factor is even, its edges can be partitioned into two orthogonal groups. For example, the 2-factor illustrated in Fig. 1(c) is even which contains one length-2 cycle $C_1 = v_2 e_8 v_3 e_{11} v_2$ and one length-4 cycle $C_2 = v_1 e_7 v_4 e_3 v_5 e_9 v_6 e_4 v_1$. Its edges can be partitioned into two orthogonal groups $\{e_8, e_7, e_9\}$ and $\{e_{11}, e_3, e_4\}$, as illustrated in Fig. 1(d).
- If a 2-factor is not even, its edges can be partitioned into three orthogonal groups. For example, as for a 2-factor illustrated in Fig. 2(a) which contains one length-4 cycle $C_1 = v_1 e_1 v_2 e_2 v_3 e_3 v_4 e_4 v_1$ and one length-5 cycle $C_2 = v_5 e_5 v_6 e_6 v_7 e_7 v_8 e_8 v_9 e_9 v_5$, its edges can be partitioned into three orthogonal groups $\{e_1, e_3, e_5, e_7\}, \{e_2, e_4, e_6, e_8\}$ and $\{e_9\}$ as illustrated in Fig. 2(b).

Based on the aforementioned facts, we have the following results for d-regular nonbinary cycle codes.

1) For a d-regular graph G with $d = 2\nu$, it has ν edge-disjoint 2-factors; if the number of even 2-factors is t,
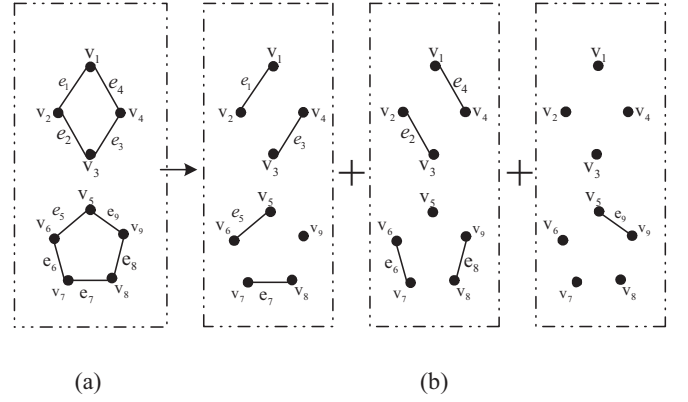
then edges of G can be partitioned into $3\nu - t = \frac{3}{2}d - t$ orthogonal groups, $0 \le t \le \frac{d}{2}$.

2) For a d-regular graph G with $d = 2\nu + 1$, if it contains at least one 1-factor, then it can be decomposed into $\nu + 1$ edge-disjoint components which consist of one 1-factor and ν 2-factors; denote the number of even 2-factors as t, then edges of G can be partitioned into $3\nu - t + 1 = \frac{3}{2}d - t - \frac{1}{2}$ orthogonal groups, $0 \le t \le \frac{d-1}{2}$.

3) If the d-regular graph G is 1-factorable, then its edges can be partitioned into d orthogonal groups.

By running updating for columns in each orthogonal group simultaneously, we can greatly improve the throughput of sequential BP decoding algorithm for nonbinary regular cycle codes by a factor at least $\frac{2n}{3d}$, relative to the sequential BP decoding working in a column by column manner. Note that n is usually large while d is usually small. Hence, the throughput improvement is significant, which is very appealing in the implementation of efficient decoders. We underscore that the performance and complexity advantages of sequential BP decoding are not compromised.

IV. DESIGN OF NONBINARY REGULAR CYCLE CODES

Now we look into the design issues of nonbinary regular cycle codes. Two steps are needed to design nonbinary regular cycle codes. The first step is to design the code structure that specifies the locations of nonzero entries in the check matrix. The code structure is reflected by the associated graph, which is desired to have good properties such as large girth, small diameter and good expansion property [23], [24]. The second step is to determine the nonzero entries of the parity check matrix.

On the first step, two existing methods are available to find good associated graphs.

1) One can utilize known regular graphs with good properties, such as the Ramanujan graphs [25]. Reference [6] first utilized this kind of promising graphs to construct cycle codes. Later, reference [13] showed through simulations that these cycle codes can achieve performance within 1 dB away from the corresponding Shannon limits, including codes of rate $1/2$, $2/3$ and $3/4$. However, good known graphs may be very limited in the number of code choices.

2) One can resort to computer search algorithms. Computer search based algorithms have been widely adopted to construct LDPC codes [14], [26]. Among them the progressive edge-growth (PEG) [14] algorithm has been shown efficient and feasible for constructing LDPC codes with short code lengths and high rates as well as LDPC codes with long code lengths. We can also use the PEG algorithm to construct regular cycle codes, including bipartite regular cycle codes and regular cycle codes which can be decomposed using Theorem 2 from Section II [1].

On the second step, references [13], [15]–[17] have addressed the issue for the selection of nonzero entries for cycle codes. Essentially, resolvable cycles [10] with short length correspond to low-weight codewords, which may induce undetected errors during the decoding process [13], [16], [17]. Therefore, to lower the error floor, it becomes desirable to make all cycles irresolvable, especially those with short lengths.

In this paper, we propose a novel design method based on the structure results presented in Section II.

A. The Proposed Code Structure Design

Based on the structures presented in Section II, we propose to construct good regular associated graphs through carefully designing interleavers. Note that references [27], [28] have also proposed other LDPC code construction methods based on interleaver design utilizing different code structure representations. Reference [27] designed the edge-interleaver for an arbitrary LDPC code whereas reference [28] designed an LDPC code with its Tanner graph compromising of an upper tree and a lower tree which are connected through an interleaver. We could utilize Theorems 1, 2, 3 and 4 to construct nonbinary regular cycle codes. Next we will illustrate the code design using Theorem 1.

Following Theorem 1, for a regular cycle code with $d = 2\nu$, its parity check matrix has an equivalent form as shown in (3). Correspondingly, its associated graph G can be decomposed into ν components — G_1 to G_ν, where each G_u is a 2-factor corresponding to $\bar{\mathbf{H}}_u$, $1 \leq u \leq \nu$. Considering two vertices i and j in G, let $\mathcal{D}^u(i, j)$ denote the distance between vertices i and j in G_u — the associated graph of $\bar{\mathbf{H}}_u$. $\mathcal{D}^u(i, j)$ is defined as the minimum length of a path traversing from i to j in G_u. Note that each G_u is comprised of disjoint cycles. Define $\mathcal{D}^u(i, j)$ to be infinite if vertices i and j are not connected in G_u, i.e., when they belong to different cycles. If they are connected in G_u, i.e., when they belong to the same cycle, there are exactly two paths traversing from i to j in G_u. In this case, $\mathcal{D}^u(i, j)$ is the minimum length of the two paths.

1) Code Design for Rate 1/2: For rate-1/2 code, $d = 4$ and $n = 2m$. According to Theorem 1, the parity check matrix $\mathbf{H}$ can be partitioned as $[\bar{\mathbf{H}}_1, \mathbf{P}_2\bar{\mathbf{H}}_2]$. Hence, the associated graph

(a). Type I constraint   (b). Type II constraint   (c). Type III constraint

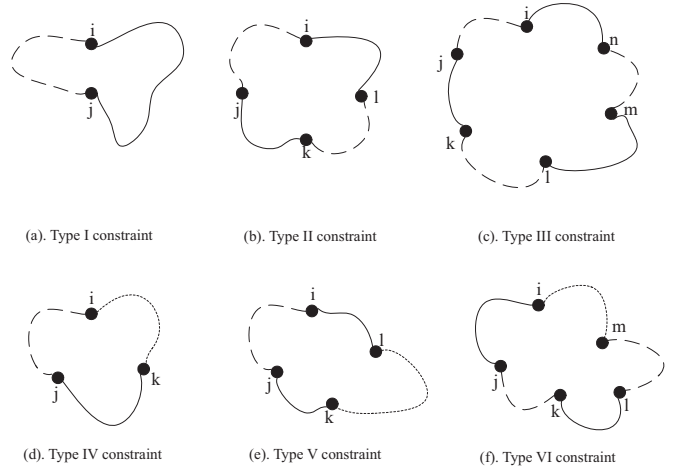(d). Type IV constraint   (e). Type V constraint   (f). Type VI constraint

Fig. 3. Illustration of different types of constraints to break cycles in the code's associated graph, where black circles represent vertices in the associated graph and different types of lines represent paths in different G_u—the associated graph of $\bar{\mathbf{H}}_u$.

G can be decomposed into two components, G_1 corresponding to $\bar{\mathbf{H}}_1$ and G_2 corresponding to $\bar{\mathbf{H}}_2$. G_1 and G_2 share the same set of vertices of G whereas having disjoint edge sets. Once G_1 and G_2 are given, the design problem reduces to the design of two interleavers π_1 and π_2. In particular π_1 can be chosen to be the identity interleaver and π_2 plays the role of $\mathbf{P}_2$.

Except for cycles contained purely in G_1 or G_2, cycles in G can also be formed by traversing through paths in G_1 and G_2 with shared vertices alternatively. Accordingly, the formation of cycles in G can be categorized depending on the number of different paths in G_1 and G_2 traversed by the cycle. The first three types illustrated in Fig. 3 represent possible formation of cycles with two components involved (G_1 and G_2) where different line types denote paths in different components. The type I cycle shown in Fig. 3 (a) is formed by a path in G_1 followed by a path in G_2 where the two paths share the same two end points. If the target girth g of the associated graph G is larger than 2, than the following constraint has to be satisfied for any two different vertices i and j.

$$\mathcal{D}^1(\pi_1(i), \pi_1(j)) + \mathcal{D}^2(\pi_2(j), \pi_2(i)) \geq g \quad (9)$$

The type II cycle shown in Fig. 3 (b) is formed by two paths in G_1 and two paths in G_2. If the target girth g is larger than 4, than the following constraint has to be satisfied for any four different vertices i, j, k and l.

$$\mathcal{D}^1(\pi_1(i), \pi_1(j)) + \mathcal{D}^2(\pi_2(j), \pi_2(k)) + \\ \mathcal{D}^1(\pi_1(k), \pi_1(l)) + \mathcal{D}^2(\pi_2(l), \pi_2(i)) \geq g \quad (10)$$

The type III cycle shown in Fig. 3 (c) is formed by three paths in G_1 and three paths in G_2. If the target girth g is larger than 6, than the following constraint has to be satisfied for any six different vertices i, j, k, l, m and n.

$$\mathcal{D}^1(\pi_1(i), \pi_1(j)) + \mathcal{D}^2(\pi_2(j), \pi_2(k)) + \\ \mathcal{D}^1(\pi_1(k), \pi_1(l)) + \mathcal{D}^2(\pi_2(l), \pi_2(m)) + \\ \mathcal{D}^1(\pi_1(m), \pi_1(n)) + \mathcal{D}^2(\pi_2(n), \pi_2(i)) \geq g \quad (11)$$

---

[1] Although the original PEG algorithm aims to construct a bipartite Tanner graph, the same principle of PEG can be adopted to construct associated graphs for regular and bipartite regular cycle codes, including regular cycle codes which can be decomposed using Theorem 2. Specifically, if $d = 2\nu + 1$ and m is even, we can first establish $\frac{m}{2}$ disjoint edges in G, then apply the PEG algorithm to obtain a $2\nu + 1$-regular graph G. In this way, the obtained G can be decomposed using Theorem 2.

This can be generalized to any g. Besides, the same set of constraints have to be held when the roles of π_1 and π_2 exchange. To achieve a target girth g, the lengths of cycles contained purely in G_1 and G_2 have to be at least g as well. With these three constraints being satisfied, the maximum girth that can be achieved for rate-1/2 codes is 8.

2) Code Design for Rate 2/3: For rate-2/3 code, $d = 6$ and $n = 3m$. According to Theorem 1, the parity check matrix $\mathbf{H}$ can be partitioned as $[\bar{\mathbf{H}}_1, \mathbf{P}_2\bar{\mathbf{H}}_2, \mathbf{P}_3\bar{\mathbf{H}}_3]$. Hence, the associated graph G can be decomposed into three components, G_1, G_2, and G_3, corresponding to $\bar{\mathbf{H}}_1$, $\bar{\mathbf{H}}_2$, and $\bar{\mathbf{H}}_3$, respectively. G_1, G_2 and G_3 share the same set of vertices of G whereas having disjoint edge sets. Once G_1, G_2 and G_3 are given, the design problem reduces to the design of three interleavers π_1, π_2 and π_3. In particular π_1 can be chosen to be the identity interleaver, π_2 and π_3 play the role of $\mathbf{P}_2$ and $\mathbf{P}_3$, respectively.

Except for those constraints with two components involved, we also need to consider constraints with three components involved. The fourth, fifth and sixth types of cycles illustrated in Fig. 3 (d), (e) and (f) represent possible formation of cycles with three components involved. The type IV cycle shown in Fig. 3 (d) is formed by a path in G_1 followed by a path in G_2 and a path in G_3. If the target girth g of the associated graph is larger than 3, than the constraint corresponding to Fig. 3 (d) is

$$\mathcal{D}^1(\pi_1(i), \pi_1(j)) + \mathcal{D}^2(\pi_2(j), \pi_2(k)) + \\ \mathcal{D}^3(\pi_3(k), \pi_3(i)) \geq g \quad (12)$$

Similarly, if the target girth g is larger than 4, than the constraint corresponding to Fig. 3 (e) is

$$\mathcal{D}^1(\pi_1(i), \pi_1(j)) + \mathcal{D}^2(\pi_2(j), \pi_2(k)) + \\ \mathcal{D}^3(\pi_3(k), \pi_3(l)) + \mathcal{D}^2(\pi_2(l), \pi_2(i)) \geq g \quad (13)$$

If the target girth g is larger than 5, than the constraint corresponding to Fig. 3 (f) is

$$\mathcal{D}^1(\pi_1(i), \pi_1(j)) + \mathcal{D}^2(\pi_2(j), \pi_2(k)) + \\ \mathcal{D}^1(\pi_1(k), \pi_1(l)) + \mathcal{D}^2(\pi_2(l), \pi_2(m)) + \\ \mathcal{D}^3(\pi_3(m), \pi_3(i)) \geq g \quad (14)$$

This can be generalized to any g. Besides, the same set of constraints have to be held when the roles of π_1, π_2 and π_3 exchange. To achieve a target girth g, the lengths of cycles contained in one or two components of $\{G_1, G_2, G_3\}$ have to be at least g as well. With all the six constraints being satisfied, the maximum girth that can be achieved for rate-2/3 codes is 6.

3) Code Design for Rate Higher Than 2/3: The same principle can be generalized to code design for any rate $r = \frac{\nu-1}{\nu}$ with $d = 2\nu$. Take the code design of rate 3/4 as an example. If the target girth g of the associated graph is larger than 4, the seventh constraint with four components involved (not shown in Fig. 3) can be represented as

$$\mathcal{D}^1(\pi_1(i), \pi_1(j)) + \mathcal{D}^2(\pi_2(j), \pi_2(k)) + \\ \mathcal{D}^3(\pi_3(k), \pi_3(l)) + \mathcal{D}^4(\pi_4(l), \pi_4(i)) \geq g \quad (15)$$

This can be generalized to any g. Besides, the same set of constraints have to be held when the roles of π_1, π_2, π_3 and

π_4 exchange. To achieve a target girth g, the lengths of cycles contained in one, two or three components of $\{G_1, G_2, G_3, G_4\}$ have to be at least g as well. With all the seven constraints being satisfied, the maximum girth that can be achieved for rate-3/4 codes is 5.

It is feasible to consider higher rate code design and to include more than four components. With moderate block length in bits (less than $10,000$), as the code rate increases, the achievable maximum girth decreases, most likely the constraints with more than four components will be satisfied. So it is reasonable to consider at most four components. This makes the algorithm feasible and efficient for even higher code rates.

4) Computer Search of Interleavers: Given the constraints above, we do trial-and-test computer search to find the interleavers, which is similar to the computer search of S-random interleavers proposed by Dolinar and Divsalar [29] and its variants [30]–[32].

As an example, the following algorithm is used to find interleavers for regular cycle codes of rate-2/3.

Algorithm 1

> INPUTS: the target girth g, the length of the interleavers N, the maximum trial-and-test number $t_{\max}$ and the structures of G_1, G_2 and G_3.
> OUTPUTS: the three interleavers for a rate-2/3 code.
> Set $i = 1$, $t = 1$.
> LOOP WHILE $i \leq N$
> 1) Generate randomly the i-th entries for the three interleavers. Make sure that the generated entry for each interleaver has not been used before.
> 2) Check whether the corresponding girth constraints in eqns (9)–(14) are satisfied or not
>   a) If satisfied, store the entries and set $i = i + 1$, $t = 1$.
>   b) If not satisfied, check whether t is smaller than $t_{\max}$ or not.
>   i) If t is smaller than $t_{\max}$, set $t = t + 1$.
>   ii) Otherwise, set $g = g - 1$, $t = 1$.
> END OF LOOP

The structures of G_1, G_2 and G_3 can be determined by hand to make sure that the lengths of cycles contained purely in G_1, G_2 or G_3 are no less than the target girth. Furthermore, more than one runs of **Algorithm 1** may be needed before obtaining good interleavers.

V. SIMULATION RESULTS

In this section, we perform simulations to evaluate the performance of nonbinary regular cycle codes. In all simulations the codewords are transmitted over the binary AWGN channel with inputs of ± 1 and additive noise of variance σ^2. If one communicates using a code of rate r then it is conventional to describe the signal-to-noise ratio (SNR) by $E_b/N_0 = 1/(2r\sigma^2)$ and to report this number in decibels as $10 \log_{10} E_b/N_0$. Unless other stated, we run simulations until more than 40 block errors have been observed or up to one million block decodings for each SNR and the maximum number of iterations for decoding is set to be 80 for both binary and nonbinary LDPC codes.

TABLE I
CYCLE DISTRIBUTION SPECTRUM OF CYCLE CODES IN FIG. 4 WITH
LENGTH 1008 BITS.

|	Galois Field	Girth	Number of cycles of different lengths					
			4	5	6	7	8	9
Irregular	GF(64)	6	0	0	54	235	493	1214
Regular	GF(64)	5	0	3	47	227	481	1172
Bipartite	GF(64)	4	1	0	112	0	976	0
Irregular	GF(256)	5	0	5	74	221	495	1189
Regular	GF(256)	5	0	7	72	212	478	1177

Fig. 4. Performance comparison of regular/bipartite regular cycle codes with irregular cycle codes, including codes with optimized nonzero entries. Also included are those of a binary degree-distribution-optimized irregular LDPC code and the MacKay's (3,6) regular code—"504.504.3.504"; The code length is 1008 bits.

For codes with optimized nonzero entries, we adopt the following two steps to choose the nonzero entries to make as many cycles irresolvable as possible, especially those having short lengths: i) given the code's associated graph, we find out all the short-length cycles of concern through computer search, e.g., using depth-first search algorithms; ii) we choose appropriate nonzero entries to render as many short-length cycles irresolvable as possible. Note that the sub-matrix associated with a length-k cycle is equivalent to $\tilde{\mathbf{H}}^c$ as shown in (2). The cycle is irresolvable iff $\tilde{\mathbf{H}}^c$ is full-rank, i.e., $\prod_{i=1}^{k} \alpha_i^{-1}\beta_i \neq 1$. Define $\gamma_i = \alpha_i^{-1}\beta_i$ to be the *gain* of the edge e_i. The second step can be done in a trial-and-test manner to find appropriate γ_i. Given γ_i for an edge e_i, the value α_i is randomly generated and $\beta_i = \alpha_i\gamma_i$. Note that this procedure applies to any cycle code.

A. Performance of Regular Cycle Codes

We first investigate the performance of regular cycle codes. We have constructed regular and irregular cycle codes over GF(64) and GF(256) where the code rate is $1/2$ and the codeword length is 1008 bits. We have also constructed a bipartite regular cycle code over GF(64). The parity check matrices of irregular, regular and bipartite regular nonbinary cycle codes are constructed by the PEG method. Table I lists the girth parameter and the cycle distribution spectrum for these constructed cycle codes. It can be seen from Table I that the number of cycles increases exponentially as the length of cycle increases. For example, the number of cycles of length 6, 7, 8 and 9 are 54, 235, 493 and 1214, respectively for the irregular cycle code over GF(64). It can also be seen from Table I that all the cycle codes defined over the same Galois field, including irregular, regular/bipartite regular cycle codes, have similar cycle distribution spectrum though their associated graphs may have different girths. Therefore we would expect that they achieve similar performance.

Fig. 4 shows the performance comparison of these cycles codes with *randomly* generated nonzero entries for their parity check matrices. Also plotted are the performance of a binary irregular rate-$1/2$ LDPC code constructed by the PEG algorithm and that of a rate-$1/2$ MacKay's regular-(3,6) code — "504.504.3.504", both having a code length of 1008 bits and decoded by standard BP. The binary irregular code has a density-evolution-optimized degree distribution pair achieving an impressive iterative decoding threshold of 0.3347 dB, from Table II in [33], i.e. the symbol-node edge distribution is $0.23802x + 0.20997x^2 + 0.03492x^3 + 0.12015x^4 + 0.01587x^6 +$
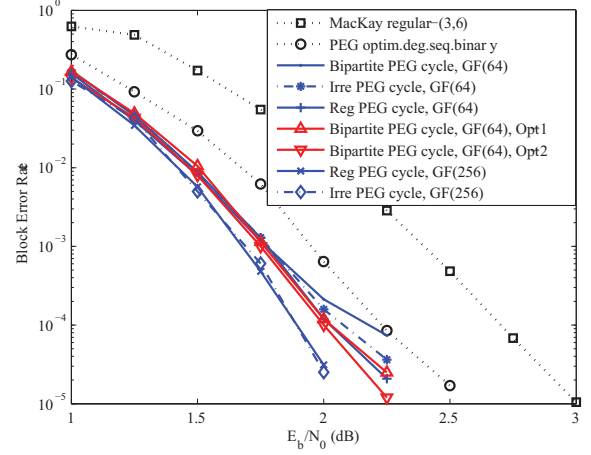
$0.00480x^{13} + 0.37627x^{14}$ and the check-node edge distribution is $0.98013x^7 + 0.01987x^8$. It can be seen from Fig. 4 that regular and irregular cycle codes over GF(256) have similar performance throughout the whole range of testing SNR, both outperforming their binary irregular counterpart by about 0.5 dB at BLER of 10^{-5}. It can also be seen from Fig. 4 that the performance of cycle codes over GF(64) starts to diverge at BLER of 10^{-4}. Among them the irregular and bipartite regular cycle codes over GF(64) start to demonstrate error floors at E_b/N_0 of 2.25 dB. These error floors come from the contribution of undetected errors of weight 6 with *randomly* generated nonzero entries. For example, for the bipartite regular cycle code over GF(64) with randomly generated nonzero entries, there are 25 undetected errors out of 40 errors at E_b/N_0 of 2.25 dB, out of which 24 are of weight 6 and 1 is of weight 8.

This error floor can be effectively lowered by optimizing the nonzero entries in the check matrix. We have constructed two optimized codes for the bipartite regular cycle code over GF(64). For the 'Opt-1' code, all cycles of length 4 and 6 are rendered irresolvable. For the 'Opt-2' code, all cycles of length 4, 6, and 8 are rendered irresolvable. Their performance curves are also shown in Fig. 4. It can be seen from Fig. 4 that the 'Opt-2' code can achieve 0.25 dB gain against the binary irregular counterpart at BLER of 10^{-5}. Besides, the code's error floor has been effectively alleviated by optimizing the nonzero entries. Actually, for the 'Opt-1' code, there are only 5 undetected errors of weight at least 8 out of 25 errors at E_b/N_0 of 2.25 dB. For the 'Opt-2' code, there are 10 undetected errors of weight at least 10 out of 24 errors at E_b/N_0 of 2.25 dB.

B. Performance of Very-High-Rate Regular Cycle Codes

We now investigate the performance of nonbinary regular cycle codes of very high rates, e.g., $r = 8/9$ and $r = 15/16$. We have constructed regular cycle codes of rate $8/9$ with lengths around 2000 and 4000 bits and regular cycle codes

TABLE II
CYCLE DISTRIBUTION SPECTRUM OF REGULAR CYCLE CODES OF RATE
8/9 AND 15/16

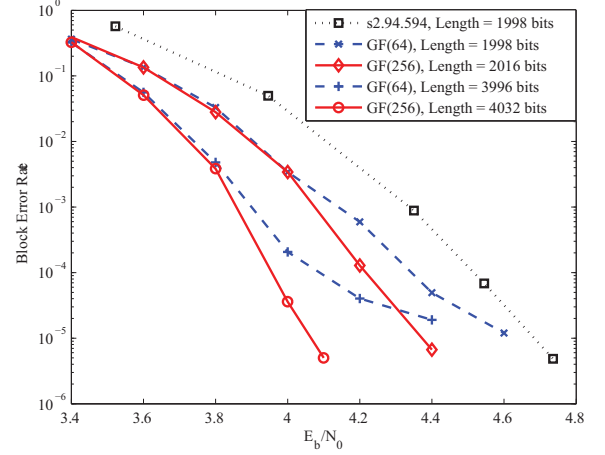| Rate	Galois Field	Length	Girth	Number of cycles		
				length-2	length-3	length-4
8/9	GF(64)	1998	2	1	866	11089
8/9	GF(256)	2016	2	1	916	11213
8/9	GF(64)	3996	3	0	499	11738
8/9	GF(256)	4032	3	0	714	11270
15/16	GF(64)	4608	2	1	5277	120453
15/16	GF(256)	4352	2	5	5430	122256
15/16	GF(64)	9216	3	0	4782	119121
15/16	GF(256)	8704	2	1	5065	119381

Fig. 5. Performance of regular cycle codes of rate 8/9=0.8889; MacKay's code s2.94.594 of rate 8/9 is also included; The Shannon capacity for rate 8/9 is about 3.1 dB.
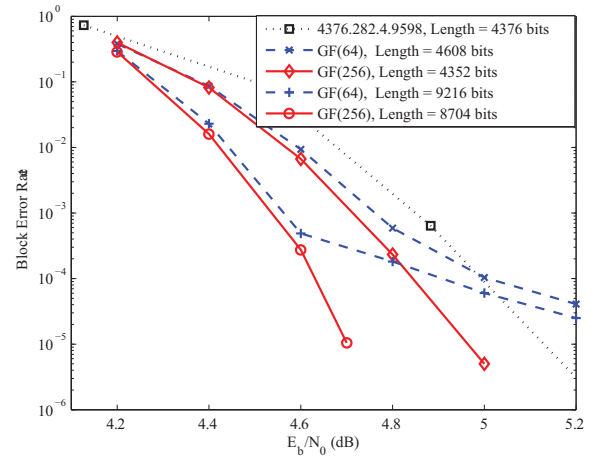

Fig. 6. Performance of regular cycle codes of rate 15/16 = 0.9375; MacKay's code 4376.282.4.9598 of rate 0.936 is also included; The Shannon capacity for rate 15/16 is about 3.9 dB.

of rate 15/16 with lengths around 4500 and 9000 bits. Their parity check matrices are constructed by the PEG method. Table II lists their lengths, girths and their cycle distribution spectrums. Note that the number of length-4 cycles for all rate 8/9 (15/16, resp.) codes considered are more than $10,000$ ($100,000$, resp.).

Fig. 5 shows the simulation results of the constructed regular cycle codes of rate-8/9 decoded by standard BP algorithm. Also included is that of MacKay's 's2.94.594' code of length 1998 bits [34], [35]. The nonzero entries for all the cycle codes are optimized. For all codes of rate 8/9, all cycles of length 2 and 3 are rendered irresolvable. For codes over GF(64), undetected errors play a significant contribution at BLER about 10^{-5}. Actually the two codes over GF(64) start to show error floors at BLER about 10^{-5} which is due to the contribution of a large portion of undetected errors. For example, there exist 14 undetected errors of weight no less than 4 out of 24 errors for the code of length 1998 over GF(64) at BLER of 2×10^{-5} (4.6 dB). This becomes even worse with block length doubled. There are 20 undetected errors of weight no less than 4 out of 24 errors for the code of length 3996 over GF(64) at BLER of 2×10^{-5} (4.4 dB). For codes over GF(256), undetected errors start to contribute at BLER about 10^{-5}. For example, there exist 4 undetected errors of weight no less than 4 out of 10 errors for the code of length 4032 over GF(256) at BLER of 6×10^{-6} (4.1 dB). Nevertheless, it can been seen from Fig. 5 that the rate-8/9 regular cycle code over GF(256) with block length 2016 bits achieves about 0.3 dB gain against MacKay's 's2.94.594' code at BLER of 10^{-5}. Besides, rate-8/9 regular cycle code over GF(256) with length about 4000 bits is within 1 dB away from the corresponding Shannon limit at BLER of 10^{-5}.

Fig. 6 shows the simulation results of the constructed regular cycle codes of rate-15/16 decoded by standard BP algorithm. Also included is that of MacKay's '4376.282.4.9598' code of length 4376 bits [34], [35]. The nonzero entries for all the cycle codes are optimized. For all the codes of rate 15/16, all cycles of length 2 are rendered irresolvable and only part of cycles of length 3 are rendered irresolvable. The same observations as for rate-8/9 hold. Besides, we can seen from Fig. 6 that the error floors for codes of rate-15/16 over GF(64) show up even earlier than that of rate-8/9 over GF(64) as shown in Fig. 5. This is due to the exponential increase of the number of short-length cycles as the code

rate increases. Nevertheless, it can be seen from Fig. 6 that the rate-15/16 regular cycle code over GF(256) with block length 4352 bits achieves about 0.2 dB gain against MacKay's '4376.282.4.9598' code at BLER of 10^{-5}. Besides, rate-15/16 regular cycle code over GF(256) with length about 9000 bits is within 1 dB away from the corresponding Shannon limit at BLER of 10^{-5}.

Compared with existing codes in literature, regular cycle codes show performance gain. Fig. 6 in [36] presented an irregular rate-0.9 LDPC code of length 4550 bits which can achieve BLER of 10^{-5} at about 4.6 dB, and is about 1.35 dB away from the capacity. Rate-8/9 regular cycle code over GF(256) of length 4032 bits shown in Fig. 5 can achieve BLER of 10^{-5} within 1 dB away from the capacity which is 0.35 dB closer than the one presented in [36].

C. Sequential BP Decoding With Parallel Processing

Now we investigate the comparison of different scheduling schemes for decoding of regular cycle codes. We have done
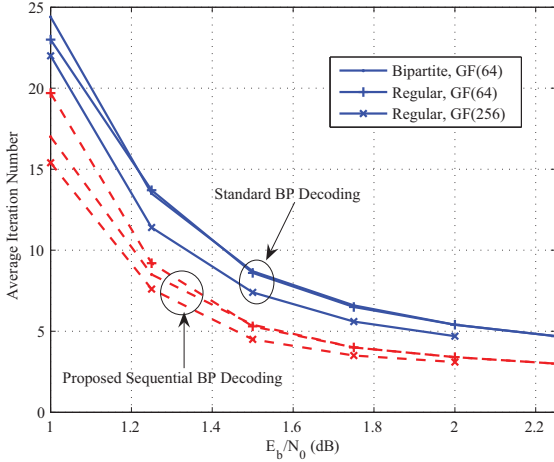
Fig. 7. Comparison on the average number of iterations of the proposed sequential BP decoding with parallel processing and that of standard BP decoding for those regular cycle codes in Fig.4.

TABLE III
CYCLE DISTRIBUTION SPECTRUM OF REGULAR CYCLE GF(64) CODES
OF RATE 1/2 DESIGNED BY THE PROPOSED METHOD

Method	Matrix Size	Girth	Number of resolvable cycles/cycles			
			length-7	length-8	length-9	length-10
Ramanujan	336×672	8	0/0	0/252	0/0	0/8064
PEG	336×672	7	0/29	0/443	23/1316	58/3251
Proposed 1	336×672	7	0/14	0/523	15/1389	48/3071
Proposed 2	336×672	7	0/40	0/602	12/1145	51/3243

simulations comparing the proposed sequential BP decoding with parallel processing and the standard BP decoding algorithms for those regular cycle codes shown in Fig. 4. The proposed sequential BP decoding with parallel processing achieves slightly better performance than the standard BP decoding at BLER above 10^{-5}. More importantly, Fig 7 shows the comparison on the average number of iterations between the proposed sequential BP decoding with parallel processing and standard BP decoding. It can be seen from Fig. 7 that the average number of iterations for the proposed sequential BP decoding is about 30 percent less than that of the standard BP decoding at high SNR, which translates into 30 percent reduction on the total decoding complexity. Moreover, the proposed sequential BP decoding with parallel processing enables a speedup on the throughput of sequential BP decoding by a factor at least $\frac{2n}{3d} = \frac{2 \times 126}{3 \times 4} = 21$ for the regular GF(256) code and at least $\frac{2n}{3d} = \frac{2 \times 168}{3 \times 4} = 28$ for the regular and bipartite regular GF(64) codes. Therefore we can conclude that compared with sequential and standard BP decoding, the proposed sequential BP decoding with parallel processing offers better tradeoff between the decoding complexity and throughput.

D. Code Construction Through Interleaver Design

We now investigate code design by the proposed method in Section IV.A which utilizes the equivalent structure. **Algorithm 1** presented in Section IV.A is used to search for good interleavers. After obtaining the code structure, the nonzero entries are further optimized. We have designed codes of rate 1/2, 2/3 and 3/4. The block lengths for rate 1/2, 2/3 and 3/4 are 672, 1008 and 1344 symbols over GF(64) respectively. Their corresponding associated graphs have 336 vertices. Table III, IV and V list the corresponding code parameters for rate 1/2, 2/3 and 3/4 respectively, including the size of the parity check matrix, the girth of the obtained associated graph, and the resolvable cycle/cycle distribution spectrum. Codes constructed by the PEG algorithm are included for comparison which are labelled as 'PEG'. Codes labelled as 'Ramanujan'

taken from [13] are also included for comparison which are constructed from known good graphs — the Ramanujan graphs. The 'Ramanujan' code of rate-1/2 is constructed from the Ramanujan graph $X^{3,7}$ [13] whereas the 'Ramanujan' code of rate-2/3 is constructed from the Ramanujan graph $X^{5,7}$ [13]. For each rate, we have constructed two codes by the proposed method. For the codes labelled as 'Proposed 1', each component G_u—the associated graph of $\bar{\mathbf{H}}_u$, is comprised of 1 length-336 cycle. Each component G_u for the code labelled as 'Proposed 2' of rate 1/2 is comprised of 42 length-8 cycles whereas each component G_u for the codes labelled as 'Proposed 2' of rate 2/3 and 3/4 is comprised of 56 length-6 cycles.

We can see from Table III, IV and V that the codes constructed from Ramanujan graphs always possess the largest girth and have the best resolvable cycle distribution spectrum. However, the differences on the resolvable cycle/cycle distribution spectrum for all the codes, including those constructed by the PEG algorithm and the proposed method are not significant. In particular, the resolvable cycle/cycle distribution spectrum for codes constructed by the PEG algorithm and the proposed method are comparable. This verifies the feasibility of the proposed method.

Fig. 8 shows the performance of all these codes. A binary irregular degree-distribution-optimized code of rate 1/2 and length 4032 is also included in Fig. 8. The degree distribution for this binary irregular code is the same as the one used in Section V.A. It can be seen from Fig. 8 that all the codes achieve similar performance at BLER above 10^{-5}. This also verifies the feasibility of the proposed method. Besides, codes of rate 1/2 constructed by the proposed method can outperform the binary irregular code. For all the codes constructed by the PEG method and the proposed method, undetected errors of small weight start to contribute at BLER about 10^{-4}. In particular, those codes of rate 2/3 and 3/4 start to experience an error floor at BLER of 10^{-5}. This error floor can be alleviated by moving to higher order Galois field, say GF(256). Nevertheless, the codes constructed by the proposed method of rate 1/2, 2/3 and 3/4 are about 1.3 dB, 1.1 dB and 1.0 dB away from the corresponding Shannon limits at BLER of 10^{-5} respectively.

Compared with existing codes in literature, such as those in [37], [38], regular cycle codes constructed by the proposed

TABLE IV

CYCLE DISTRIBUTION SPECTRUM OF REGULAR CYCLE GF(64) CODES
OF RATE 2/3 DESIGNED BY THE PROPOSED METHOD

Method	Matrix Size	Girth	Number of resolvable cycles/cycles			
			length-4	length-5	length-6	length-7
Ramanujan	336 × 1008	6	0/0	0/0	0/3360	0/0
PEG	336 × 1008	4	0/1	0/9	8/1263	81/6288
Proposed 1	336 × 1008	5	0/0	0/79	11/1570	62/5837
Proposed 2	336 × 1008	5	0/0	0/96	15/1685	93/5591

TABLE V

CYCLE DISTRIBUTION SPECTRUM OF REGULAR CYCLE GF(64) CODES
OF RATE 3/4 DESIGNED BY THE PROPOSED METHOD

Method	Matrix Size	Girth	Number of resolvable cycles/cycles		
			length-4	length-5	length-6
PEG	336 × 1344	4	0/4	12/1291	157/10740
Proposed 1	336 × 1344	4	0/15	10/1817	154/10350
Proposed 2	336 × 1344	4	0/13	10/1818	154/10506

method achieve performance gains. In reference [37], a rate-1/2 128-ary (2032, 1016) quasi-cyclic LDPC code has been constructed, which has block length of 14224 bits. This code achieves BLER of 10^{-5} at E_b/N_0 of 2.0 dB. Our rate-1/2 codes over GF(64) achieve BLER of 10^{-5} at E_b/N_0 of 1.5 dB, which is 0.5 dB better than the 128-ary (2032, 1016) quasi-cyclic LDPC code reported in [37], even though the block lengths of our codes are much smaller. In reference [38], a rate-0.7510 256-ary (1020, 766) quasi-cyclic LDPC codes has been constructed, which has block length of 8160 bits. This code achieves BLER of 10^{-5} at E_b/N_0 of 3.6 dB. Our rate-3/4 codes over GF(64) have similar block length and can achieve BLER of 10^{-5} at E_b/N_0 of 2.6 dB, which is 1 dB better than the 256-ary (1020, 766) quasi-cyclic LDPC code reported in [38].

Compared with other advanced binary irregular LDPC codes, such as those in [39], [40], our codes have similar performance. Figs. 1 and 2 in [39] presented a rate-1/2 code of length 5120 bits which can achieve BLER of 10^{-5} at about 1.5 dB. Fig. 8 in [40] presented a rate-1/2 code of length 5792 bits which can achieve BLER of 10^{-5} at about 1.5 dB. Fig. 8 in [40] also presented a rate-3/4 code of length 5792 bits which can achieve BLER of 10^{-5} at about 2.6 dB. All these codes have similar performance as our codes of the same rate shown in Fig. 8.

VI. CONCLUSIONS

In this paper we focused on a special class of nonbinary cycle codes—nonbinary regular cycle codes. Through graph-theoretic analysis, an equivalent structure has been derived for
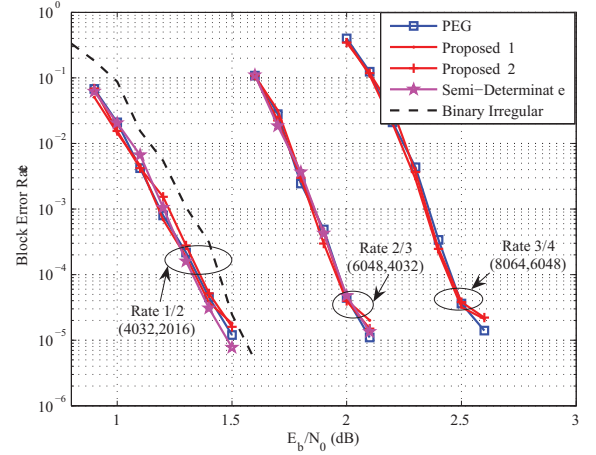

Fig. 8. Performance comparison of regular cycle codes designed using different methods. The Shannon capacities are 0.188, 1.084 and 1.628 dB for rate 1/2, 2/3 and 3/4 respectively. The 'Semi-Determinate' codes are constructed from Ramanujan graphs and taken from [13].

the parity check matrix $\mathbf{H}$. Encoding utilizing this structure can be performed in parallel in linear time. The storage requirements for $\mathbf{H}$ can also be reduced. In addition, decoding utilizing this structure enables parallel processing in sequential BP decoding, which considerably increases the decoding throughput without compromising performance or complexity. We also presented a novel method to design the code structure for nonbinary regular cycle codes. Extensive simulations confirm that nonbinary regular cycle codes have very good performance. Future research topics include code design utilizing the equivalent structure with deterministic interleavers, such as interleavers based on permutation polynomials over integer rings [41].

REFERENCES

[1] J. Huang, S. Zhou, and P. Willett, "Structure of non-binary regular LDPC cycle codes," in *Proc. IEEE ICASSP*, Mar.–Apr., 2008.
[2] R. G. Gallager, *Low Density Parity Check Codes*. Cambridge, MA: MIT Press, 1963.
[3] D. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.
[4] M. C. Davey and D. MacKay, "Low-density parity-check codes over GF(q)," *IEEE Commun. Lett.*, vol. 2, pp. 165–167, June 1999.
[5] D. Jungnickel and S. A. Vanstone, "Graphical codes revisted," *IEEE Trans. Inform. Theory*, vol. 43, pp. 136–146, Jan. 1997.
[6] X.-Y. Hu and E. Eleftheriou, "Binary representation of cycle Tanner-graph GF(2^b) codes," in *Proc. IEEE International Conf. Commun.*, June 2004, vol. 27, no. 1, pp. 528–532.
[7] M. C. Davey, "Error-correction using low-density parity-check codes," dissertation, University of Cambridge, 1999.
[8] H. Song and J. R. Cruz, "Reduced-complexity decoding of Q-ary LDPC codes for magnetic recording," *IEEE Trans. Magn.*, vol. 39, pp. 1081–1087, Mar. 2003.
[9] L. Barnault and D. Declercq, "Fast decoding algorithm for LDPC codes over GF(2^q)," in *Proc. IEEE Inform. Theory Workshop*, 2003.
[10] J. Huang and J.-K. Zhu, "Linear time encoding of cycle GF(2^p) codes through graph analysis," *IEEE Commun. Lett.*, vol. 10, pp. 369–371, May 2006.
[11] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Transa. Inform. Theory*, vol. 27, pp. 533–547, Sep. 1981.
[12] D. Reinhard, *Graph Theory*, 2nd ed. Springer-Verlag, 2000.
[13] J. Huang, S. Zhou, J.-K. Zhu, and P. Willett, "Group-theoretic analysis of Cayley-graph-based cycle GF(2^p) codes," *IEEE Trans. Commun.*, vol. 57, no. 7, pp. 1560–1565, June 2009.

[14] X.-Y. Hu, E. Eleftheriou, and D.-M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Trans. Inform. Theory*, vol. 51, no. 1, pp. 386–398, Jan. 2005.

[15] D. J. C. MacKay, "Optimizing sparse graph codes over GF(q)," Aug. 2003. [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/

[16] C. Poulliat, M. P. Fossorier, and D. Declercq, "Using binary images of nonbinary LDPC codes to improve overall performance," in *Proc. 4th Int. Symp. Turbo Codes Related Topics*, Apr. 2006.

[17] D. Kimura, R. Pyndiah, and F. Guilloud, "Construction of parity-check matrices for non-binary LDPC codes," in *Proc. 4th Int. Symp. Turbo Codes and Related Topics*, Apr. 2006.

[18] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, pp. 498–519, Feb. 2001.

[19] C. Berrou and A. Glavieux, "Near-optimum error-correcting coding and decoding: Turbo-codes," *IEEE Trans. Commun.*, vol. 44, pp. 1261–1271, Oct. 1996.

[20] J.-T. Zhang and M. P. C. Fossorier, "Shuffled iterative decoding," *IEEE Trans. Commun.*, vol. 53, pp. 209–213, Feb. 2005.

[21] H. Kfir and I. Kanter, "Parallel versus sequential updating for belief propagation decoding," *Physica A: Statistical Mechanics Applications*, vol. 330, pp. 259–270, Dec. 2003.

[22] J.-T. Zhang and M. P.C.Fossorier, "Shuffled belief propagation decoding," in *Proc. 36th Asilomar Conf. Signals, Syst. Computers*, Nov. 2002, vol. 1, pp. 8–15.

[23] J. Rosenthal and P. O. Vontobel, "Constructions of LDPC codes using Ramanujan graphs and ideals from Margulis," in *Proc. 38th Annual Allerton Conf. Commun., Control, Computing*, 2000, pp. 248–257.

[24] M. Sipser and D. A. Spielman, "Expander codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 1710–1722, Nov. 1996.

[25] P. S. G. Davidoff and A. Valette, *Elementary Number Theory, Group Theory, and Ramanujan Graphs*. Cambridge University Press, 2002.

[26] Y. Mao and A. H. Banihashemi, "A heuristic search for good LDPC codes at short block lenghts," in *Proc. IEEE Intl. Conf. Commun.*, June 2001, vol. 1, pp. 41–44.

[27] O. Y. Takeshita, "A compact construction for LDPC codes using permutation polynomials," in *Proc. IEEE International Symp. Inform. Theory*, July 2006, pp. 79–82.

[28] J. Lu and J. M. F. Moura, "TS-LDPC codes: Turbo-structured codes with large girth," *IEEE Trans. Inform. Theory*, vol. 53, no. 3, pp. 1080–1094, Mar. 2007.

[29] S. Dolinar and D. Divsalar, "Weight distribution for turbo codes using random and nonrandom permutations," *TDA Progress Rep. 42-122*, Aug. 1995.

[30] S. N. Crozier, "New high-spread high-distance interleavers for turbo-codes," in *Proc. Biennial Symp. Commun.*, May 2000, pp. 3–7.

[31] L. Dinoi and S. Benedetto, "Design of fast-prunable S-random interleavers," *IEEE Trans. Wireless Commun.*, vol. 4, no. 5, pp. 2540–2548, Sep. 2005.

[32] C. Fragouli and R. D. Wesel, "Semi-random interleaver design criteria," in *Proc. IEEE Globecom*, Dec. 1999, vol. 5, pp. 2352–2356.

[33] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 619–637, Feb. 2001.

[34] D. J. C. MacKay and M. C. Davey, "Evaluation of Gallager codes for short block length and high rate applications," in *Codes, Systems and Graphical Models*, Springer-Verlag, pp. 113–130, 2000.

[35] D. J. C. MacKay and E. A. Ratzer, "Gallager codes for high rate applications," 2003. [Online]. Available: http://www.inference.phy.cam.ac.uk/mackay/

[36] M. Yang, W. E. Ryan, and Y. Li, "Design of efficiently encodable moderate-length high-rate irregular LDPC codes," *IEEE Trans. Commun.*, vol. 52, no. 4, pp. 564–571, Apr. 2004.

[37] B. Zhou, Y.-Y. Tai, L. Lan, S. Song, L. Zeng, and S. Lin, "Construction of high performance and efficiently encodable nonbinary quasi-cyclic LDPC codes," in *Proc. IEEE Globecom*, Nov.–Dec. 2006, pp. 1–6.

[38] L. Zeng, L. Lan, Y. Tai, B. Zhou, S. Lin, and K. Abdel-Ghaffar, "Construction of nonbinary cyclic, quasi-cyclic and regular LDPC codes: A finite geometry approach," *IEEE Trans. Commun.*, vol. 56, no. 3, pp. 378–387, Mar. 2008.

[39] T. Richardson and R. Urbanke, "Multi-edge type LDPC codes." [Online]. Available: http://lthcwww.epfl.ch/papers/multiedge.ps

[40] D. Divsalar and C. Jones, "Protograph LDPC codes with node degrees at least 3," in *Proc. IEEE GlobeCom*, Nov.–Dec. 2006, pp. 1–5.

[41] J. Sun and O. Y. Takeshita, "Interleavers for turbo codes using permutation polynomials over integer rings," *IEEE Trans. Inform. Theory*, vol. 51, pp. 101–119, Jan. 2005.
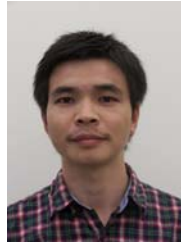
Jie Huang received the B.S. degree in 2001 and the Ph. D. degree in 2006, from the University of Science and Technology of China (USTC), Hefei, both in electrical engineering and information science. He is now a research assistant professor with the Department of Electrical and Computer Engineering at the University of Connecticut, Storrs.

His general research interests lie in the areas of communications and signal processing, specifically error control coding theory and coded modulation system design. His recent focus is on signal processing, channel coding, and network coding for underwater acoustic communications and underwater sensor networks.

Shengli Zhou (M'03) received the B.S. degree in 1995 and the M.Sc. degree in 1998, from the University of Science and Technology of China (USTC), Hefei, both in electrical engineering and information science. He received his Ph.D. degree in electrical engineering from the University of Minnesota (UMN), Minneapolis, in 2002. He has been an assistant professor with the Department of Electrical and Computer Engineering at the University of Connecticut (UCONN), Storrs, 2003-2009, and now is an associate professor.

His general research interests lie in the areas of wireless communications and signal processing. His recent focus is on underwater acoustic communications and networking. He received the 2007 ONR Young Investigator award and the 2007 Presidential Early Career Award for Scientists and Engineers (PECASE).

Peter Willett (F'03) received his BASc (Engineering Science) from the University of Toronto in 1982, and his PhD degree from Princeton University in 1986. Since then, he has been a faculty member at the University of Connecticut, and since 1998 has been a professor. His primary areas of research have been statistical signal processing, detection, communications, data fusion, and tracking.

He is editor-in-chief for the IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS. He has been a member of the IEEE AESS Board of Governors since 2003.