LETTER

Nonbinary Quasi-Cyclic LDPC Cycle Codes with Low-Density Systematic Quasi-Cyclic Generator Matrices*

Yang YANG[†a)], Chao CHEN[††], Jianjun MU[†], Jing WANG[†††], *Nonmembers*, Rong SUN[†], *Member*, *and* Xinmei WANG[†], *Nonmember*

SUMMARY In this letter, we propose an appealing class of nonbinary quasi-cyclic low-density parity-check (QC-LDPC) cycle codes. The parity-check matrix is carefully designed such that the corresponding generator matrix has some nice properties: 1) systematic, 2) quasi-cyclic, and 3) sparse, which allows a parallel encoding with low complexity. Simulation results show that the performance of the proposed encoding-aware LDPC codes is comparable to that of the progressive-edge-growth (PEG) constructed nonbinary LDPC cycle codes.

key words: *low-density parity-check (LDPC) code, nonbinary, quasi-cyclic (QC), low-density generator matrix, cycle code*

1. Introduction

Low-density parity-check (LDPC) codes, invented by Gallager in the early 1960s [1] and rediscovered by MacKay and Neal in 1996 [2], have been shown to be a class of capacity-approaching codes with iterative decoding [3]. Various constructions and decoding algorithms for binary LDPC codes have been extensively investigated in the literature. In contrast to binary LDPC codes, only a few results are available for nonbinary LDPC codes. Nonbinary LDPC codes over finite fields were first introduced by Davey and MacKay in 1998 [4]. Inspired by Richardson and Urbanke [3], Davey devised an efficient Fast Fourier Transform based q-ary sum-product decoding algorithm (FFT-QSPA) [5].

Simulations and analyses demonstrate that good LDPC codes over GF(q) ($q > 2$) tend to have a number of weight-2 columns in the parity-check matrix and the proportion of weight-2 columns increases, as q increases [6], [7]. If the parity-check matrix consists of exactly weight-2 columns, the code is called cycle code in the literature. Poulliat et al. proposed a method to design cycle codes using binary images [8].

Quasi-cyclic (QC)-LDPC codes are a class of structured LDPC codes that are implementation-friendly for both encoding and decoding [9]–[11]. Li et al. proposed an efficient encoding method for general QC-LDPC codes by finding out the generator matrix in systematic-circulant form [12]. This method facilitates parallel encoding and can be generalized to encode nonbinary codes [13]. However, the resultant generator matrix is in general dense [14], which makes the encoder quite resource-consuming especially for codes over high-order finite fields, a motivation for us to exploit appropriate code structure to make the generator matrix also sparse.

In this letter, we propose a special class of nonbinary QC-LDPC cycle codes over finite fields, which are featured by the following properties:

- *Ultra-sparse quasi-cyclic parity-check matrix.*
- *Sparse systematic quasi-cyclic generator matrix.*
- *Parallel encoding with low complexity.*

2. Nonbinary Quasi-Cyclic LDPC Codes

Binary QC-LDPC codes have been widely investigated in the literature. The code construction is usually described in two steps: 1) build a binary base matrix; 2) lift the base matrix with zero matrices and sparse circulant matrices such as circulant permutation matrices (CPMs).

2.1 Definition of Nonbinary QC-LDPC Codes

Nonbinary QC-LDPC codes can be seen as a generalization of binary QC-LDPC codes. To lift the base matrix, Lin et al. introduced a special type of CPM called α-multiplied CPM (α is a primitive field element) in which each row is the right cyclic-shift of the row above it multiplied by α and the first row is the right cyclic-shift of the last row multiplied by α [13]. Peng et al. generalized it to β-multiplied CPM (β is some definite power of α) [15]. In both constructions, the nonzero entries of a CPM are assigned with different nonzero field elements. In this letter, we consider another simple QC lifting method, in which each CPM is assigned with a single nonzero field element.

Mathematically, the parity-check matrix has a compact representation and can be defined by

$$\mathbf{H} = \begin{bmatrix} e_{0,0}\mathbf{P}^{q_{0,0}} & e_{0,1}\mathbf{P}^{q_{0,1}} & \ldots & e_{0,n-1}\mathbf{P}^{q_{0,n-1}} \\ e_{1,0}\mathbf{P}^{q_{1,0}} & e_{1,1}\mathbf{P}^{q_{1,1}} & \ldots & e_{1,n-1}\mathbf{P}^{q_{1,n-1}} \\ \vdots & \vdots & \ddots & \vdots \\ e_{m-1,0}\mathbf{P}^{q_{m-1,0}} & e_{m-1,1}\mathbf{P}^{q_{m-1,1}} & \ldots & e_{m-1,n-1}\mathbf{P}^{q_{m-1,n-1}} \end{bmatrix} \quad (1)$$

Manuscript received March 8, 2011.

[†] The authors are with the State Key Lab of ISN, Xidian University, Xi'an 710071, China.

[††] The author is with China Academy of Space Technology, Xi'an 710000, China.

[†††] The author is with the School of Information Engineering, Chang'an University, Xi'an 710064, China.

[*] This work is supported jointly by the 973 Program under Grant 2010CB328300, the National Natural Science Foundation of China under Grant 61001132 and 61040005.

a) E-mail: yyang@mail.xidian.edu.cn

DOI: 10.1587/transcom.E94.B.2620

$$(\mathbf{M}^{-1})_{i,j} = \begin{cases} \alpha^{-1} a_j^{-1} \left[\prod_{l=i}^{j-1} (a_l^{-1} b_l)\right] \mathbf{P}^{\left[\sum_{l=i}^{j-1} (t_l - s_l)\right] - s_j} & (i < j) \\ \alpha^{-1} a_j^{-1} \mathbf{P}^{-s_j} & (i = j) \\ \alpha^{-1} a_j^{-1} \left[\prod_{l=0}^{j-1} (a_l^{-1} b_l)\right] \left[\prod_{l=i}^{m'-1} (a_l^{-1} b_l)\right] \mathbf{P}^{\left[\sum_{l=0}^{j-1} (t_l - s_l)\right] + \left[\sum_{l=i}^{m'-1} (t_l - s_l)\right] - s_j} & (i > j) \end{cases} \tag{8}$$

where $e_{i,j} \in \mathrm{GF}(2^b)$, $q_{i,j} \in \{\infty, 0, 1, \ldots, L-1\}$. $\mathbf{P}^{q_{i,j}}$ is an $L \times L$ zero matrix if $q_{i,j} = \infty$; otherwise, it is an $L \times L$ CPM obtained by cyclically shifting the rows of the identity matrix $\mathbf{I}$ to the right by $q_{i,j}$ times. Thus, each CPM $e_{i,j}\mathbf{P}^{q_{i,j}}$ in $\mathbf{H}$ is specified by a field element $e_{i,j}$ and a shift factor $q_{i,j}$, and can be denoted by a 2-tuple $(e_{i,j}, q_{i,j})$.

As is proved in [12], if the rightmost m block-columns (i.e. the rightmost mL columns) of $\mathbf{H}$ are linearly independent, the corresponding generator matrix can be written as

$$\begin{aligned} \mathbf{G} &= \left[\, \mathbf{I} \,\middle|\, \mathbf{G}_{\mathrm{p}} \,\right] \\ &= \left[\begin{array}{cccc|cccc} \mathbf{I} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{G}_{0,0} & \mathbf{G}_{0,1} & \cdots & \mathbf{G}_{0,m-1} \\ \mathbf{0} & \mathbf{I} & \ddots & \vdots & \mathbf{G}_{1,0} & \mathbf{G}_{1,1} & \cdots & \mathbf{G}_{1,m-1} \\ \vdots & \ddots & \ddots & \mathbf{0} & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{I} & \mathbf{G}_{k-1,0} & \mathbf{G}_{k-1,1} & \ldots & \mathbf{G}_{k-1,m-1} \end{array}\right] \end{aligned} \tag{2}$$

where $k = n - m$; $\mathbf{I}$ and $\mathbf{G}_{\mathrm{p}}$ correspond to the systematic part and the parity part, respectively; $\mathbf{G}_{i,j}$ is an $L \times L$ circulant matrix, which usually has high-density.

2.2 Two Optimization Methods

Two optimization methods for code construction were proposed in the literature in terms of cycle property and minimum distance property.

Theorem 1 ([9]). *Let $\mathbf{P}^{q_1} \to \mathbf{P}^{q_2} \to \cdots \to \mathbf{P}^{q_{2l}} \to \mathbf{P}^{q_1}$ be the chain corresponding to a length-2l block-cycle in $\mathbf{H}$, if r is the least positive integer such that*

$$r \cdot \sum_{i=1}^{2l} (-1)^{i-1} q_i \equiv 0 \pmod{L} \tag{3}$$

then the block-cycle leads to L length-2lr cycles in $\mathbf{H}$.

From Theorem 1, we can optimize the shift factors $\{q_i\}$ to eliminate short cycles in $\mathbf{H}$.

Theorem 2 ([8], [15]). *Let $(e_1, e_2, \cdots, e_{2l}) \in \mathrm{GF}(q)$ be the nonzero field elements assigned to a length-2l cycle in $\mathbf{H}$, then the cycle provides $(q-1)$ weight-l codewords if and only if*

$$\prod_{i=1}^{l} e_{2i-1} e_{2i}^{-1} = 1. \tag{4}$$

From Theorem 2, we can optimize the nonzero field elements assigned to the nonzero entries in $\mathbf{H}$ to avoid low-weight codewords and thus improve the minimum distance.

3. Code Structure and Encoding Circuit

In this section, we exploit a special code structure which enables every $\mathbf{G}_{i,j}$ in $\mathbf{G}$ of (2) to be a CPM, thus resulting in a low-density generator matrix.

3.1 A Useful Matrix

Consider the following matrix which will be used in our code construction

$$\mathbf{M} = \begin{bmatrix} a_0\mathbf{P}^{s_0} & b_0\mathbf{P}^{t_0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & a_1\mathbf{P}^{s_1} & b_1\mathbf{P}^{t_1} & \ddots & \vdots \\ \vdots & \mathbf{0} & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \vdots & \ddots & a_{m'-2}\mathbf{P}^{s_{m'-2}} & b_{m'-2}\mathbf{P}^{t_{m'-2}} \\ b_{m'-1}\mathbf{P}^{t_{m'-1}} & \mathbf{0} & \cdots & \mathbf{0} & a_{m'-1}\mathbf{P}^{s_{m'-1}} \end{bmatrix} \tag{5}$$

where $a_i, b_i \in \mathrm{GF}(2^b)$, $a_i, b_i \neq 0$; $\mathbf{P}^{s_i}, \mathbf{P}^{t_i}$ are $L \times L$ CPMs. We give a sufficient condition for $\mathbf{M}$ to be non-singular, under which the inverse of $\mathbf{M}$ is composed of CPMs.

Proposition. $\mathbf{M}$ *is non-singular if*

$$\sum_{i=0}^{m'-1} (t_i - s_i) \equiv 0 \pmod{L} \quad \textit{and} \quad \prod_{i=0}^{m'-1} (a_i^{-1} b_i) \neq 1. \tag{6}$$

Proof. Using elementary row operations, $\mathbf{M}$ is transformed to an equivalent matrix $\mathbf{M}'$.

$$\mathbf{M}' = \begin{bmatrix} a_0\mathbf{P}^{s_0} & b_0\mathbf{P}^{t_0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & a_1\mathbf{P}^{s_1} & b_1\mathbf{P}^{t_1} & \ddots & \vdots \\ \vdots & \mathbf{0} & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \vdots & \ddots & a_{m'-2}\mathbf{P}^{s_{m'-2}} & b_{m'-2}\mathbf{P}^{t_{m'-2}} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{D} \end{bmatrix} \tag{7}$$

where $\mathbf{D} = a_{m'-1}\mathbf{P}^{s_{m'-1}} \left[\mathbf{I} + \left(\prod_{i=0}^{m'-1} a_i^{-1} b_i\right)\left(\mathbf{P}^{\sum_{i=0}^{m'-1}(t_i - s_i)}\right)\right]$. Since $\det(\mathbf{M}) = \det(\mathbf{M}') = \left[\prod_{i=0}^{m'-2} \det(a_i\mathbf{P}^{s_i})\right] \det(\mathbf{D})$ and $\det(a_i\mathbf{P}^{s_i}) \neq 0$, where $\det(\cdot)$ denotes the matrix determinant, $\det(\mathbf{M}) \neq 0$ if and only if $\det(\mathbf{D}) \neq 0$. If $\sum_{i=0}^{m'-1}(t_i - s_i) \equiv 0$ (mod L) and $\prod_{i=0}^{m'-1}(a_i^{-1} b_i) \neq 1$, $\det(\mathbf{D}) \neq 0$, and therefore $\det(\mathbf{M}) \neq 0$. □

By performing rather cumbersome elementary row operations on the augmented matrix $[\mathbf{M} \,|\, \mathbf{I}]$ until it reaches reduced row echelon form $\left[\mathbf{I} \,|\, \mathbf{M}^{-1}\right]$, the inverse matrix $\mathbf{M}^{-1}$

can be obtained as in (8) where $\alpha = 1 + \prod_{i=0}^{m'-1}(a_i^{-1}b_i)$. It can be easily seen that $\mathbf{M}^{-1}$ is composed of CPMs.

When carefully examining the conditions in (6), we note that $\mathbf{M}$ provides L length-$2m'$ cycles and these cycles do not lead to low-weight codewords, according to Theorem 1 and Theorem 2.

3.2 Code Structure

The parity-check matrix of our proposed code is defined as

$$\mathbf{H} = \left[\, \mathbf{H}_\mathrm{s} \,\middle|\, \mathbf{H}_\mathrm{p} \,\right] = \left[\begin{array}{c|cc} \mathbf{H}_{\mathrm{s}_1} & \mathbf{H}_{\mathrm{p}_1} & \mathbf{0} \\ \mathbf{H}_{\mathrm{s}_2} & \mathbf{0} & \mathbf{H}_{\mathrm{p}_2} \end{array}\right] \tag{9}$$

where $\mathbf{H}_\mathrm{s}$ and $\mathbf{H}_\mathrm{p}$ correspond to the systematic part and the parity part, respectively; $\mathbf{H}_{\mathrm{p}_1}$ and $\mathbf{H}_{\mathrm{p}_2}$ are both in the form of matrix $\mathbf{M}$; $\mathbf{H}_{\mathrm{s}_1}$ and $\mathbf{H}_{\mathrm{s}_2}$ are both composed of $L \times L$ CPMs and $L \times L$ zero matrices.

The corresponding generator matrix is defined as

$$\mathbf{G} = \left[\, \mathbf{I} \,\middle|\, \mathbf{G}_\mathrm{p} \,\right] = \left[\, \mathbf{I} \,\middle|\, \mathbf{G}_{\mathrm{p}_1} \;\; \mathbf{G}_{\mathrm{p}_2} \,\right]. \tag{10}$$

Then by $\mathbf{H}\mathbf{G}^\mathrm{T} = \mathbf{0}$, we have

$$\mathbf{G}_{\mathrm{p}_1}^\mathrm{T} = \mathbf{H}_{\mathrm{p}_1}^{-1}\mathbf{H}_{\mathrm{s}_1} \quad \text{and} \quad \mathbf{G}_{\mathrm{p}_2}^\mathrm{T} = \mathbf{H}_{\mathrm{p}_2}^{-1}\mathbf{H}_{\mathrm{s}_2}. \tag{11}$$

Theorem 3. *Let* $\mathbf{H}$ *be given in* (9), *if the column-weights of* $\mathbf{H}_{\mathrm{s}_1}$ *and* $\mathbf{H}_{\mathrm{s}_2}$ *are both one, then there exists a corresponding* $\mathbf{G}$ *in the form of* (2) *with each* $\mathbf{G}_{i,j}$ *being a CPM.*

Proof. If $\mathbf{H}_{\mathrm{s}_1}$ and $\mathbf{H}_{\mathrm{s}_2}$ satisfy the condition in Theorem 3, then there is only one $L \times L$ CPM in each block-column of $\mathbf{H}_{\mathrm{s}_1}$ and $\mathbf{H}_{\mathrm{s}_2}$. And since $\mathbf{H}_{\mathrm{p}_1}^{-1}$ and $\mathbf{H}_{\mathrm{p}_2}^{-1}$ are both composed of $L \times L$ CPMs as shown in (8), it can be easily checked from (10)–(11) that $\mathbf{G}$ has the form of (2) and each $\mathbf{G}_{i,j}$ is an $L \times L$ CPM. □

Remark. The parity-check matrix constructed according to Theorem 3 has a uniform column-weight of two, thus giving a nonbinary cycle code. And most interestingly, the parity part of the corresponding generator matrix is composed of CPMs, which makes the generator matrix much sparser than an ordinary one.

3.3 Encoding Circuit

Encoding is the process of determining the parity sequence $\mathbf{v}$ given the information sequence $\mathbf{u}$. To encode, the information sequence $\mathbf{u}$ is divided into k sections, $\mathbf{u} = (\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \cdots, \mathbf{u}^{(k-1)})$, and each section consists of L consecutive components of $\mathbf{u}$. Likewise, the parity sequence $\mathbf{v}$ is divided into m sections, $\mathbf{v} = (\mathbf{v}^{(0)}, \mathbf{v}^{(1)}, \cdots, \mathbf{v}^{(m-1)})$, and each section consists of L consecutive components of $\mathbf{v}$.

Then from (2), we have

$$\mathbf{v}^{(j)} = \mathbf{u}^{(0)}\mathbf{G}_{0,j} + \mathbf{u}^{(1)}\mathbf{G}_{1,j} + \cdots + \mathbf{u}^{(k-1)}\mathbf{G}_{k-1,j}. \tag{12}$$

The jth parity section $\mathbf{v}^{(j)}$ can be computed by a shift-register-multiplier-adder (SRMA) circuit, as shown in Fig. 1. The tapping point in the ith cyclic shift register is determined by the shift factor of the CPM $\mathbf{G}_{i,j}$, and $g_{i,j}$ is the non-zero field element specifying $\mathbf{G}_{i,j}$.

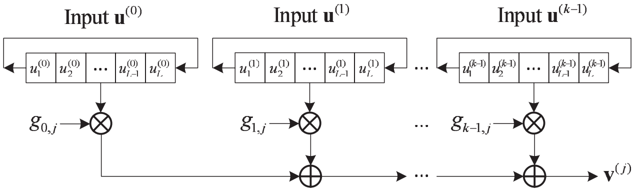
Input $\mathbf{u}^{(0)}$
Input $\mathbf{u}^{(1)}$
Input $\mathbf{u}^{(k-1)}$
$u_1^{(0)}$ $u_2^{(0)}$ ··· $u_{L-1}^{(0)}$ $u_L^{(0)}$
$u_1^{(1)}$ $u_2^{(1)}$ ··· $u_{L-1}^{(1)}$ $u_L^{(1)}$
$u_1^{(k-1)}$ $u_2^{(k-1)}$ ··· $u_{L-1}^{(k-1)}$ $u_L^{(k-1)}$
$g_{0,j}$
$g_{1,j}$
$g_{k-1,j}$
···
$\mathbf{v}^{(j)}$

Fig. 1 SRMA circuit.

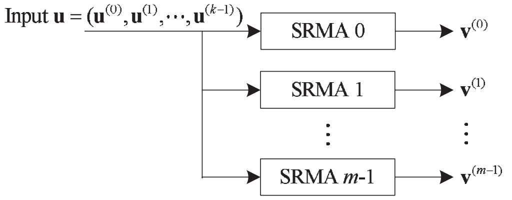
Input $\mathbf{u} = (\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \cdots, \mathbf{u}^{(k-1)})$
SRMA 0
SRMA 1
SRMA m-1
$\mathbf{v}^{(0)}$
$\mathbf{v}^{(1)}$
$\mathbf{v}^{(m-1)}$

Fig. 2 SRMA based encoder.

The entire encoder is obtained by parallel concatenating m SRMA circuits, as shown in Fig. 2. Note that the m SRMA circuits share the same set of k cyclic shift registers. After $\mathbf{u}^{(0)}, \mathbf{u}^{(1)}, \cdots, \mathbf{u}^{(k-1)}$ are loaded into the k cyclic shift registers, the m SRMA circuits generate $\mathbf{v}^{(0)}, \mathbf{v}^{(1)}, \cdots, \mathbf{v}^{(m-1)}$ in parallel in L clock-cycles. Thanks to the low-density characteristic of the proposed generator matrix, only km field elements are needed to be stored and only km two-input finite field multipliers plus $(k-1)m$ two-input finite field adders are needed to be implemented in the encoder, much fewer than the logic and storage requirements of encoders based on ordinary generator matrices. These resource savings due to the sparse generator matrix may compensate for the resource-consuming computations over high-order finite fields.

4. Simulation Results

In this section, using the proposed structure, we construct a QC-LDPC cycle code over GF(64) of length 168 and rate-1/2. The parity-check matrix is defined as

$$\mathbf{H}_\mathrm{s} = \begin{bmatrix}
(32,6) & & & & & & (15,2) & & & & & \\
 & (22,4) & & & & & & (36,2) & & & & \\
 & & (62,2) & & & & & & (31,1) & & & \\
 & & & (28,5) & & & & & & (1,3) & & \\
 & & & & (28,5) & & & & & & (15,1) & \\
 & & & & & (32,1) & & & & & & (56,5) \\
(44,0) & & & & & & (1,4) & & & & & \\
 & (38,6) & & & & & & (1,5) & & & & \\
 & & (43,6) & & & & & & (4,1) & & & \\
 & & & (21,2) & & & & & & (5,1) & & \\
 & & & & (35,1) & & & & & & (2,3) & \\
 & & & & & (7,4) & & & & & & (5,5)
\end{bmatrix}$$

and

$$\mathbf{H}_\mathrm{p} = \begin{bmatrix}
(38,2) & (58,3) & & & & & & & & & & \\
& (42,4) & (22,6) & & & & & & & & & \\
& & (40,4) & (34,2) & & & & & & & & \\
& & & (60,5) & (12,5) & & & & & & & \\
& & & & (57,0) & (62,3) & & & & & & \\
(56,5) & & & & & (44,2) & & & & & & \\
& & & & & & (11,6) & (15,6) & & & & \\
& & & & & & & (39,5) & (47,6) & & & \\
& & & & & & & & (40,0) & (30,3) & & \\
& & & & & & & & & (7,2) & (21,2) & \\
& & & & & & & & & & (55,5) & (35,0) \\
& & & & & & (13,6) & & & & & (61,5)
\end{bmatrix}.$$

According to (8), The parity part of the corresponding generator matrix is

$$\mathbf{G}_\mathrm{p} = \begin{bmatrix}
(38,3) & (41,4) & (47,6) & (51,4) & (58,4) & (25,0) & (63,6) & (27,6) & (25,0) & (29,3) & (40,3) & (39,5) \\
(44,6) & (5,0) & (57,2) & (52,0) & (33,0) & (4,3) & (24,6) & (7,6) & (61,0) & (46,3) & (56,3) & (49,5) \\
(19,6) & (23,0) & (12,2) & (56,0) & (29,0) & (45,3) & (32,0) & (53,0) & (56,1) & (43,4) & (34,4) & (62,6) \\
(46,6) & (51,0) & (39,2) & (44,0) & (38,0) & (60,3) & (33,3) & (19,3) & (41,4) & (30,0) & (30,0) & (43,2) \\
(8,1) & (29,2) & (63,4) & (13,2) & (57,2) & (37,5) & (51,0) & (39,0) & (23,1) & (52,4) & (17,4) & (30,6) \\
(1,4) & (27,5) & (46,0) & (25,5) & (62,5) & (57,1) & (57,2) & (20,2) & (59,3) & (38,6) & (23,6) & (41,1) \\
(51,0) & (22,1) & (42,3) & (55,1) & (46,1) & (63,4) & (59,2) & (2,2) & (12,3) & (16,6) & (29,6) & (48,1) \\
(11,1) & (48,2) & (63,4) & (13,2) & (57,2) & (1,5) & (46,0) & (24,0) & (58,1) & (27,4) & (5,4) & (45,6) \\
(40,0) & (42,1) & (6,3) & (28,1) & (47,1) & (55,4) & (30,5) & (22,5) & (55,6) & (4,2) & (54,2) & (12,4) \\
(14,1) & (4,2) & (30,4) & (24,2) & (21,2) & (46,5) & (53,4) & (54,4) & (50,5) & (58,1) & (58,1) & (35,3) \\
(31,5) & (37,6) & (13,1) & (27,6) & (52,6) & (59,2) & (60,5) & (44,5) & (45,6) & (47,2) & (20,2) & (24,4) \\
(17,0) & (33,1) & (19,3) & (3,1) & (15,1) & (59,4) & (50,1) & (1,1) & (6,2) & (8,5) & (47,5) & (24,0)
\end{bmatrix}.$$

The size of each CPM is 7×7. We use the primitive polynomial $P(x) = x^6 + x + 1$ over GF(2) to construct GF(64). Field elements are represented by integer notation. For comparison, we also construct a progressive-edge-growth (PEG) LDPC cycle code with the same length, rate, and field order. Shift factors in the parity-check matrix are optimized according to Theorem 1 for our proposed code. For both codes, nonzero field elements in the parity-check matrices are optimized according to Theorem 2.

The simulation scenario is as follows. The codes are modulated using BPSK and transmitted over the AWGN channel. FFT-QSPA is employed as the decoding algorithm and the maximum iteration number is set to be 80. For each simulation point, we run until at least 100 frame errors are detected. The BER and FER performances are shown in Fig. 3. It is seen that the proposed code has almost the same performance as the PEG constructed code in the low to moderate signal-to-noise ratio (SNR) region, and performs slightly better in the high SNR region.

5. Conclusion

We have proposed a special class of nonbinary QC-LDPC cycle codes with sparse systematic quasi-cyclic generator matrices that allow low-complexity parallel encoding. The encoder can be implemented through simple shift-register circuits with low logic and memory requirements. Simulation results show that the proposed codes have no performance degradation due to the constraint on their special structure, as compared to the PEG constructed nonbinary LDPC cycle codes.

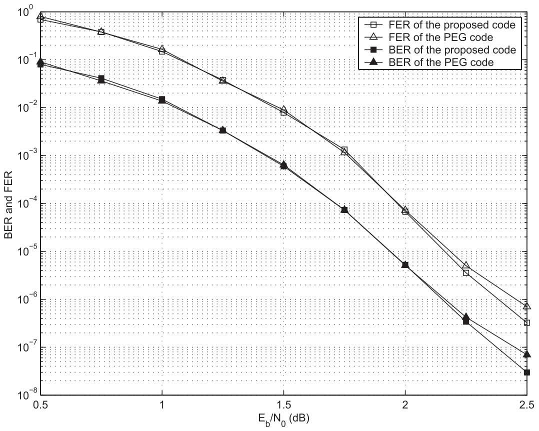

Fig. 3 Performance comparison between the proposed LDPC code and the PEG constructed LDPC code.

References

[1] R. Gallager, "Low-density parity-check codes," IEEE Trans. Inf. Theory, vol.IT-8, no.1, pp.21–28, 1962.

[2] D. MacKay and R. Neal, "Near shannon limit performance of low density parity check codes," IEE Electron. Lett., vol.32, no.18, pp.1645–1646, 1996.

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

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

[5] M. Davey, Error-Correction Using Low-Density Parity-Check Codes, Ph.D. Thesis, Univ. of Cambridge, Cambridge, U.K., Dec. 1999.

[6] M. Davey and D. MacKay, "Monte carlo simulations of infinite low density parity check codes over GF(q)," Proc. Int. Workshop on Optimal Codes and related Topics, pp.9–15, Bulgaria, June 1998.

[7] X.Y. Hu and E. Eleftheriou, "Binary representation of cycle tanner-graph GF(2^b) codes," Proc. IEEE Int. Conf. on Commun., pp.528–532, June 2004.

[8] C. Poulliat, M. Fossorier, and D. Declercq, "Design of regular $(2, d_c)$-LDPC codes over GF(q) using their binary images," IEEE Trans. Commun., vol.56, no.10, pp.1626–1635, 2008.

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

[10] IEEE Std 802.16e-2005, "IEEE standard for local and metropolitan area networks part 16: Air interface for fixed and mobile broadband wireless access systems amendment 2," Feb. 2006.

[11] M. Mansour and N. Shanbhag, "High-throughput LDPC decoders," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.11, no.6, pp.976–996, 2003.

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

[13] L. Zeng, L. Lan, Y. Tai, S. Song, S. Lin, and K. Abdel-Ghaffar, "Constructions of nonbinary quasi-cyclic LDPC codes: A finite field approach," IEEE Trans. Commun., vol.56, no.4, pp.545–554, 2008.

[14] K. Andrews, S. Dolinar, and J. Thorpe, "Encoders for block-circulant LDPC codes," Proc. IEEE Int. Symp. Inf. Theory, pp.2300–2304, Sept. 2005.

[15] R.H. Peng and R.R. Chen, "Design of nonbinary quasi-cyclic LDPC cycle codes," Proc. IEEE Inf. Theory Workshop, pp.13–18, Sept. 2007.