EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

RESEARCH **Open Access**

Improved progressive edge-growth algorithm for fast encodable LDPC codes

Xueqin Jiang[1,2*], Moon Ho Lee[3] and Jinpeng Qi[1,2]

Abstract

The progressive edge-growth (PEG) algorithm is known to construct low-density parity-check (LDPC) codes at finite code lengths with large girths by establishing edges between symbol and check nodes in an edge-by-edge manner. The linear-encoding PEG (LPEG) algorithm, a simple variation of the PEG algorithm, can be applied to generate linear time encodable LDPC codes whose m parity bits $p_1, p_2, ..., p_m$ are computed recursively in m steps. In this article, we propose modifications of the LPEG algorithm to construct LDPC codes whose number of encoding steps is independent of the code length. The maximum degree of the symbol nodes in the Tanner graph is denoted by $d_s^{\max}$; The m parity bits of the proposed LDPC codes are divided into $d_s^{\max}$ subgroups and can be computed in only $d_s^{\max}$ steps. Since $d_s^{\max} \ll m$, the number of encoding steps can be significantly reduced. It has also been proved that the PEG codes and the codes proposed in this article have similar lower bound on girth. Simulation results showed that the proposed codes perform very well over the AWGN channel with an iterative decoding.

1 Introduction

Low-density parity-check (LDPC) codes, which were first proposed in the early 1960's [1] and re-discovered in 1996 [2], have recently attracted much attention due to their capacity-approaching performance and low decoding complexity. Since their re-discovery, there are many methods such as the message-passing decoding and the linear program decoding [3,4] that have been proposed for the decoding algorithm. Also many other methods have been proposed for the construction algorithm [5-8]. Among the existing methods, the most successful approach for the construction of LDPC codes is the progressive-edge-growth (PEG) algorithm [5,6]. The PEG construction builds up a Tanner graph, equivalent to a parity-check matrix, for an LDPC code in an edge-by-edge manner and maximizes the local girth at symbol nodes in a greedy algorithm. It is simple and flexible in that it can be applied in constructing codes of arbitrary length and rate. In addition, the PEG algorithm can be modified to construct linear time encodable LDPC codes. In this article, this modified algorithm is referred to as linear-encoding PEG (LPEG) algorithm. Initially, the parity-check matrix was used in decoding process for LDPC codes. But it can also be used for the encoding of LDPC codes [9]. LDPC codes with m parity bits constructed by the LPEG algorithm can be encoded with the parity-check matrix in m recursive steps [6]. Therefore, for a given code rate R, the number of encoding steps of LPEG codes grows linearly with the code length $n = m/(1\text{-}R)$.

The objective of this article is to reduce the number of encoding steps of the LPEG codes with negligible performance loss. To reduce the number of encoding steps, we modified the LPEG algorithm to obtain two new algorithms which are referred to as fast-encoding PEG (FPEG) algorithm and modified FPEG (MFPEG) algorithm, respectively. $d_s^{\max}$ is used to denote the maximum degree of symbol nodes. The number of encoding steps of the FPEG codes grows linearly with $d_s^{\max}$, but not the code length n. The number of encoding steps of the MFPEG codes grows linearly with $(d_s^{\max} + \left\lfloor \frac{m}{d_s^{\max}} \right\rfloor - 1)$. Since $d_s^{\max}$ is much smaller than m, the number of encoding steps of FPEG codes or MFEPG codes is lower than that of the LPEG codes. Moreover, to ensure that there is negligible performance loss, we proved that the PEG codes and our codes had similar lower bound on girth. This was

* Correspondence: xqjiang@dhu.edu.cn
[1]Donghua University, School of Information Science and Technology, Shanghai, China
Full list of author information is available at the end of the article

confirmed by providing examples of the proposed codes and comparing their performance with that of the multiple serially concatenated multiple parity-check (M-SC-MPC) code, which is a class of LDPC codes of efficient encoding [10].

The remainder of this article is organized as follows. Section 2 reviews the PEG algorithm and the LPEG algorithm. Section 3 proposes the FPEG algorithm and the MFPEG algorithm. A lower bound on the girth of the FPEG algorithm is also derived in this section. Section 4 presents two examples of FPEG codes and MFPEG codes and demonstrates their performances. Finally, Section 5 concludes the article.

2 Preliminary

2.1 Progressive edge-growth algorithm

An LDPC code is a linear block code defined by a sparse parity-check matrix H having dimension $m \times n$. A bipartite graph with m check nodes in one class and n symbol nodes in the other can be created using H as the integer-valued incidence matrix for the two classes. Such a graph is also called a Tanner graph [11]. Let $V_c = \{c_0, c_1, ..., c_{m-1}\}$ denote the set of check nodes and $V_s = \{s_0, s_1, ..., s_{n-1}\}$ denote the set of symbol nodes. E is the set of edges such that $E \subseteq V_c \times V_s$, with edge $(c_i, s_j) \in E$ if and only if $h_{i,j} \neq 0$, where $h_{i,j}$ denotes the entry of H at the ith row and jth column, $0 \leq i \leq m - 1$, $0 \leq j \leq n - 1$. The PEG algorithm for constructing a Tanner graph with n symbol nodes and m check nodes is described in Algorithm 1. In this algorithm, both symbol nodes and check nodes are ordered according to their degrees in a nondecreasing order. d_{s_j} is the degree of symbol node s_j, $N^l_{s_j}$, and $\bar{N}^l_{s_j}$ denote the set of all check nodes reached by a tree spreading from symbol node s_j with in depth l, and its complement, respectively [6]. It was proved that [4] the PEG algorithm constructs Tanner graphs having a large girth and the lower bound on the girth was proved to be

Algorithm 1. PEG algorithm

```
1: for $j = 0$ to $n - 1$ do
2:     for $k = 0$ to $d_{s_j} - 1$ do
3:         if $k = 0$ then
               $E^0_{s_j} \leftarrow$ edge $(c_i, s_j)$, where $E^0_{s_j}$ is the first edge incident to $s_j$ and $c_i$ is a check node such that it has the lowest check-node degree under the current graph setting $E_{s_0} \cup E_{s_1} \cup \cdots \cup E_{s_{j-1}}$.
4:         else
               expand a subgraph from $s_j$ up to depth $l$ under the current graph setting such that the cardinality of $N^l_{s_j}$ stops increasing but is less than $m$, or $\bar{N}^{l+1}_{s_j} \neq \emptyset$ but $\bar{N}^{l+1}_{s_j} \neq \emptyset$, then $E^k_{s_j} \leftarrow$ edge $(c_i, s_j)$, where $E^k_{s_j}$ is the $k$th edge incident on $s_j$ and $c_i$ is a check node picked from $\bar{N}^l_{s_j}$ having the lowest check-node degree.
5:         end if
6:     end for
7: end for
```

$$g_P \geq 2\left(\left\lfloor \frac{\log(md_c^{\max} - \frac{md_c^{\max}}{d_s^{\max}} - m + 1)}{\log[(d_s^{\max} - 1)(d_c^{\max} - 1)]} - 1 \right\rfloor + 2\right), (1)$$

where $d_c^{\max}$ and $d_s^{\max}$ were the maximum degrees of the check nodes and symbol nodes, respectively.

2.2 Linear-encoding PEG algorithm

It is stated [12] (Corollary 4) that if a parity-check matrix H can be transformed into an upper or lower triangular matrix by row and column permutations, the corresponding LDPC code can be encoded in linear number of steps. Obviously, the PEG algorithm can also be tailored to construct a parity-check matrix H having upper triangular structure. The code word C and the parity-check matrix H are partitioned into $C = [p, d]$ and $H = [H^p, H^d]$, respectively, such that

$$[H^p, H^d]C^T = 0, \tag{2}$$

where the $m \times m$ component $H^p = h^p_{i,j}$ of the parity-check matrix is forced to have the upper
triangular form

$$H^p = \begin{pmatrix} 1 & h^p_{1,2} & \cdots & \cdots & h^p_{1,m} \\ 0 & 1 & & & \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ & & 0 & 1 & h^p_{m-1,m} \\ 0 & \cdots & 0 & 0 & 1 \end{pmatrix} \tag{3}$$

in which $h_{ij} = 1$ for $i = j$. Hence, the parity bits $p = [p_1, p_2, ..., p_m]$ can be computed according to

$$p_i = \left(\sum_{j=i+1}^{m} h^p_{i,j} p_j \oplus \sum_{j=1}^{n-m} h^d_{i,j} d_j\right) \tag{4}$$

where $d = \{d_i\}$ is the systematic part of the code, $H^d = \{h^d_{i,j}\}$ is the $m \times (n\text{-}m)$ component of the partitioned parity-check matrix H and $\oplus$ represents the summation over binary field, i.e., an XOR operation. From Equation (4), the m parity bits can be computed from p_m to p_1 serially in m steps. Therefore, the number of encoding steps is

$$T_L = m. \tag{5}$$

Accordingly, the symbol node set V_s in the Tanner graph is partitioned into redundant subset V_s^p and the information subset V_s^d, which contain the first m symbol nodes and the other n - m symbol nodes, respectively. The edges of the symbol nodes are then established by means of the LPEG algorithm which constructs an upper triangular pattern. As the procedure of establishing the edges of n - m information bits follows the construction of edges of V_s^p and is exactly the same as the PEG algorithm described in Algorithm 1, only the LPEG algorithm for constructing edges of V_s^p is shown in Algorithm 2.

Algorithm 2. LPEG algorithm for Establishing Edges of V_s^p

```
1:for $j = 0$ to $m - 1$ do
2:    for $k = 0$ to $d_{s_j} - 1$ do
3:        if $k = 0$ then
              $E_{s_j}^0 \leftarrow \text{edge}\,(c_j, s_j)$, where $E_{s_j}^0$ is the first edge incident to $s_j$. This edge corresponds to the "1" in the diagonal line of matrix $H^p$.
4:        else
              expand a subgraph from $s_j$ up to depth $l$ under the current graph setting such that $\bar{N}_{s_j}^l \cap \{c_0, c_1, \ldots, c_{j-1}\} \neq \emptyset$ but $\bar{N}_{s_j}^{l+1} \cap \{c_0, c_1, \ldots, c_{j-1}\} = \emptyset$, or the cardinality of $N_{s_j}^l$ stops increasing, then $E_{s_j}^k \leftarrow \text{edge}\,(c_i, s_j)$, where $E_{s_j}^k$ is the kth edge incident to $s_j$ and $c_i$ is a check node picked from the set $\bar{N}_{s_j}^l \cap \{c_0, c_1, \ldots, c_{j-1}\}$ having the lowest check-node degree.
5:        end if
6:    end for
7: end for
```

Note that the first column of H^p corresponds to a degree-1 symbol node and the fraction of degree-1 symbol node is $1/n$. It was proved that the Tanner graph of an upper or lower triangular parity-check matrix could be equivalently transformed into a pseudo-tree and the corresponding LDPC codes could also be encoded in a linear number of steps by the label-and-decide algorithm [12].

2.3 M-SC-MPC codes

Let n_i, ki, and r_i denote the code length, information length and parity-check length of the ith component MPC code, respectively. The encoder of each component MPC code can be implemented as r_i SPC encoders as shown in Figure 1 [10]. The matrix cells of the ith encoder are filled in column-wise order from top left to bottom right. The first r_i - s_i cells, with $s_i = k \bmod r_i$, are unused. When the jth row is filled, j = 1, ..., r_i, the parity bit p_j is calculated by XORing the elements of the row, and its value is stored in the last column, at the same row. Each component MPC code can be seen as a shortened version of a binary cyclic code with length $N_i = \left\lceil \frac{n_i}{r_i} \right\rceil \cdot r_i \geq n_i$. We obtain a valid parity-check matrix H_i for the ith component code, consisting of a row of $\left\lceil \frac{n_i}{r_i} \right\rceil$ identity matrix with size $r_i \times r_i$. As the ith component code has length n_i, the cyclic code must be shortened. This implied eliminating the first N_i - n_i columns of H_i. H_i forms a block-row of the parity-check matrix H of the serially concatenated code. Consequently, H was in the lower triangular form, consisting of identity matrices and zero matrices. The serially concatenated code has information length k, parity-check length $m = \sum_{i=1}^{M} r_i$ and code length $n = k + m$.

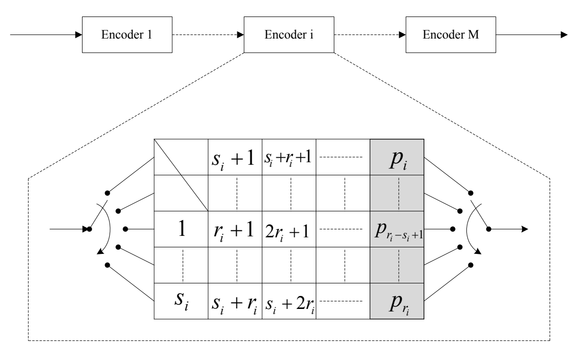

Figure 1 Structure of M-SC-MPC encoder. The encoder of each component MPC code can be implemented as r_i SPC encoders.

3 FPEG algorithm and MFPEG algorithm

In general, an ensemble of Tanner graphs is defined through degree distribution pairs. In the case of the symbol nodes, the degree distribution, from the edge perspective, is given by

$$\psi(x) = \sum_{i \geq 1}^{d_s^{\max}} \psi_i x^i, \qquad (6)$$

where ψ_i is the fraction of Tanner graph edges which emanate from degree-i symbol nodes. The fraction of degree-i symbol nodes, from the node perspective, is given by

$$\lambda_i = \frac{\psi_i / i}{\sum_j \psi_j / j}. \qquad (7)$$

Similarly, in the case of the check nodes, the degree distribution, from the edge perspective, is given by

$$\varphi(x) = \sum_{i\geq 1}^{d_c^{\max}} \varphi_i x^i, \tag{8}$$

where ϕ_i is the fraction of Tanner graph edges which emanate from degree-i check nodes. The fraction of degree-i check nodes, from the node perspective, is given by

$$\rho_i = \frac{\varphi_i/i}{\sum_j \varphi_j/j}. \tag{9}$$

In the following section, we introduce an FPEG algorithm and an MFPEG algorithm used to construct an upper triangular parity-check matrix.

3.1 FPEG algorithm

Example 1: Consider the parity-check matrix in (10),

$$H = [H^p, H^d] = \begin{pmatrix} H_1 \\ H_2 \\ H_3 \end{pmatrix} = \left(\begin{array}{c} 1\,0\,0\,1\,1\,0\,1\,0\,1\,0\,1\,0 \\ 0\,1\,1\,0\,0\,1\,0\,1\,0\,1\,0\,1 \\ \hline 0\,0\,1\,0\,1\,0\,1\,1\,1\,0\,0\,0 \\ 0\,0\,0\,1\,0\,1\,0\,0\,0\,1\,1\,1 \\ \hline 0\,0\,0\,0\,1\,0\,1\,1\,0\,1\,0\,0 \\ 0\,0\,0\,0\,0\,1\,0\,0\,1\,0\,1\,1 \end{array}\right). \tag{10}$$

This is a rate $R = 0.5$ and length $n = 12$ parity-check matrix that corresponds to an LDPC code $C = [p_1, p_2, ..., p_6, d_1, d_2, ..., d_6]$, where $d_1, d_2, ..., d_6$ are the information bits and $p_1, p_2, ..., p_6$ are the parity bits. This is by no means a good LDPC code but it is an example. The parity-check matrix H in (10) can be divided into three 2×12 submatrices H_1, H_2 and H_3. In each submatrix H_i, the rows do not have '1's in the same column. Then, the encoding of C includes the following three steps:

Step 1 Given the submatrix H_3 and the information bits $d_1, d_2, ..., d_6$, compute the parity bits p_5, p_6 by the parity-check equations

$$\begin{aligned} p_5 &= d_1 \oplus d_2 \oplus d_4, \\ p_6 &= d_3 \oplus d_5 \oplus d_6. \end{aligned} \tag{11}$$

Step 2 Given the submatrix H_2 and the information bits $d_1, d_2, ..., d_6$ and parity bits p_5, p_6, compute the check bits p_3, p_4 by the parity-check equations

$$\begin{aligned} p_3 &= d_1 \oplus d_2 \oplus d_3 \oplus p_5, \\ p_4 &= d_4 \oplus d_5 \oplus d_6 \oplus p_6. \end{aligned} \tag{12}$$

Step 3 Given the submatrix H_1 and the information bits $d_1, d_2, ..., d_6$ and check bits p_3, p_4, p_5, p_6, compute the parity bits p_1, p_2 by the parity-check equations

$$\begin{aligned} p_1 &= d_1 \oplus d_3 \oplus d_5 \oplus p_4 \oplus p_5, \\ p_2 &= d_2 \oplus d_4 \oplus d_6 \oplus p_3 \oplus p_6. \end{aligned} \tag{13}$$

It is easy to see that, in general, the number of encoding steps equals the number of submatrices, M. The parity-check matrix H should satisfy the following three conditions:

(A) The parity-check matrix H should contain an upper triangular pattern.

(B) The r_i rows in each submatrix H_i should not have '1's in the same column.

(C) The number of submatrices should not be smaller than the maximum symbol-node degree $\left(M \geq d_s^{\max}\right)$.

Condition (A) guarantees that the corresponding codes are linear time encodable [12]. Condition (B) guarantees that the r_i parity-check equations in submatrix H_i can be used to generate r_i parity bits simultaneously [10,13,14] while condition (C) is a necessary condition for condition (B). Observing these three conditions, the Tanner graph of the parity-check matrix H can be constructed with the proposed FPEG algorithm.

Divide all the check nodes $\{c_0, c_1, ..., c_{m-1}\}$ into M separate check node groups $G_1, G_2, ..., G_M$ where

$$G_i = \left\{c_{\sum_{j=1}^{i-1} r_j}, \ldots, c_{\sum_{j=1}^{i} r_j - 1}\right\} \tag{14}$$

and $r_1, r_2, ..., r_M$ are M positive integers such that

$$r_1 + r_2 + \cdots + r_M = m. \tag{15}$$

Given a symbol-node-degree distribution, the FPEG algorithm for establishing edges of V_s^p and V_s^d is given in Algorithm 3. The symbol $SC \backslash G_i$ was used to denote the check nodes contained in the set of selectable check (SC) nodes, but not in the check node group G_i, where SC denote the set of check nodes available for the next round of spreading. The number of encoding steps of the FPEG codes, equaling the number of submatrices H_i, is

$$T_F = M \geq d_s^{\max}. \tag{16}$$

Algorithm 3. FPEG algorithm for establishing edges of V_s^p and V_s^d

```
1: for $j = 0$ to $n - 1$ do
2:     if $j < m$ then
3:         $SC = \{c_0, c_1, ..., c_{j-1}\}$.
4:     else
```

```
5:          $SC = \{c_0, c_1, ..., c_{n-1}\}$.
6:      end if
7:      for $k = 0$ to $d_{s_j} - 1$ do
8:          if $k = 0$ then
9:              if $j < m$ then
                    $E^0_{s_j} \leftarrow$ edge $(c_j, s_j)$, where $E^0_{s_j}$ is the first edge incident to $s_j$. This edge corresponds to the "1" in the diagonal line of matrix $H^p$.
10:             else
                    $E^0_{s_j} \leftarrow$ edge $(c_j, s_j)$, where $E^0_{s_j}$ is the first edge incident to $s_j$ and $c_i$ is a check node such that it has the lowest check-node degree under the current graph setting $E_{s_0} \cup E_{s_1} \cup \cdots \cup E_{s_{j-1}}$.
11:             end if
12:         else
                expand a subgraph from $s_j$ up to depth $l$ under the current graph setting such that $\bar{N}^l_{s_j} \cap SC \neq \emptyset$ but $\bar{N}^{l+1}_{s_j} \cap SC = \emptyset$, or the cardinality of $N^l_{s_j}$ stops increasing, then $E^k_{s_j} \leftarrow$ edge $(c_i, s_j)$, where $E^k_{s_j}$ is the $k$th edge incident to $s_j$ and $c_i$ is a check node picked from the set $\bar{N}^l_{s_j} \cap SC$ having the lowest check-node degree.
13:         end if
14:         Find out which check node group $G_i$ includes $c_i$. $SC \leftarrow SC \backslash G_i$.
15:     end for
16: end for
```

Similar to the PEG algorithm, the check-node degrees are made as uniform as possible by the FPEG algorithm. Notice that the FPEG algorithm is not always valid for any given symbol-node-degree distribution. Since the column weight of H_i is at most one and the columns of H are ordered according to their weights in a nondecreasing order. The weight of first r_1 columns of H^p is at most one, that of the next r_2 columns is at most two, likewise the weight of the last r_M columns is at most M. In other words, the number of columns with weight less than or equal to i should be larger than or equal to $\sum_{j=1}^{i} r_j$. Therefore, the FPEG algorithm is valid if and only if

$$n \sum_{j=1}^{i} \lambda_j \geq \sum_{j=1}^{i} r_j, \tag{17}$$

for $i \in \{1, 2, ..., M\}$. When the condition (17) is satisfied, it was proved that, given a symbol-node-degree distribution, for large code lengths, the probability of failing to construct an approximate upper or lower triangular parity-check matrix was negligible [15] (Theorem 1).

Theorem 3.1 *The lower bound on girth of the Tanner graph constructed by the FPEG algorithm is*

$$g_F \geq 2\left(\left\lfloor \frac{\log(Jd_c^{\max} - \frac{Jd_c^{\max}}{d_s^{\max}} - J + 1)}{\log[(d_s^{\max} - 1)(d_c^{\max} - 1)]} - 1 \right\rfloor + 2\right), \tag{18}$$

where $\lfloor \cdot \rfloor$ *denotes the flooring operation,* $J = r_{\min}(d_s^{\max} - 1) + 1$ *and* $r_{\min}$ *is the minimum of* $\{r_1, r_2, ..., r_M\}$.

Proof: The proof of (18) is an adaptation of the proof of equation (1) reported in [6]. For a given symbol node s_j, define its neighborhood in G_k within depth l, G^l_{k,s_j}, as the set consisting of check nodes in G_k reached by a subgraph spreading from symbol node s_j. Its complementary set, $\bar{G}^l_{k,s_j}$, is defined as $G_k \backslash G^l_{k,v_j}$. Consider a depth-$l$ subgraph of an irregular Tanner graph which spreads from any symbol node s_j, $s_j \in V_s$, such that $G^l_{k,s_j} \subset G_k$ and $G^{l+1}_{k,s_j} = G_k$. By definition the depth-0 subgraph contains at most $d_s^{\max}$ check nodes, one of them is in G_k. Each of the $d_s^{\max}$ check nodes gives rise to at most $(d_s^{\max} - 1)(d_c^{\max} - 1)$ check nodes in the next round of spreading. Thus, there are at most $d_s^{\max}(d_s^{\max} - 1)(d_c^{\max} - 1)$ check nodes at depth 1, and $d_s^{\max}(d_c^{\max} - 1)$ check nodes of them are in G_k. Similarly, there are at most $d_s^{\max}(d_s^{\max} - 1)^l(d_c^{\max} - 1)^l$ check nodes at depth l and $d_s^{\max}(d_s^{\max} - 1)^{l-1}(d_c^{\max} - 1)^l$ check nodes of them are in G_k. Let l' be the largest integer such that

$$\begin{aligned} &1 + d_s^{\max}(d_c^{\max} - 1) + \cdots \\ &+ d_s^{\max}(d_s^{\max} - 1)^{l'-1}(d_c^{\max} - 1)^{l'} < r_{\min}. \end{aligned} \tag{19}$$

From (19), it is easy to see that

$$\begin{aligned} &d_s^{\max} + d_s^{\max}(d_s^{\max} - 1)(d_c^{\max} - 1) + \cdots \\ &+ d_s^{\max}(d_s^{\max} - 1)^{l'}(d_c^{\max} - 1)^{l'} \\ &< r_{\min}(d_s^{\max} - 1) + 1, \end{aligned} \tag{20}$$

which can be simplified to

$$\begin{aligned} &\frac{d_s^{\max}[(d_s^{\max} - 1)^{l'+1}(d_c^{\max} - 1)^{l'+1} - 1]}{(d_s^{\max} - 1)(d_c^{\max} - 1) - 1} \\ &< r_{\min}(d_s^{\max} - 1) + 1. \end{aligned} \tag{21}$$

Let t be the solution of the equation

$$\begin{aligned} &\frac{d_s^{\max}[(d_s^{\max} - 1)^{t+1}(d_c^{\max} - 1)^{t+1} - 1]}{(d_s^{\max} - 1)(d_c^{\max} - 1) - 1} \\ &= r_{\min}(d_s^{\max} - 1) + 1 \end{aligned} \tag{22}$$

that is,

$$t = \frac{\log(Jd_c^{\max} - \frac{Jd_c^{\max}}{d_s^{\max}} - J + 1)}{\log[(d_s^{\max} - 1)(d_c^{\max} - 1)]} - 1, \tag{23}$$

where

$$J = r_{\min}(d_s^{\max} - 1) + 1. \tag{24}$$

Then $l \geq l' = \lfloor t \rfloor$ and

$$\begin{aligned} g_F &\geq 2(\lfloor t \rfloor + 2) \\ &= 2\left(\left\lfloor \frac{\log(Jd_c^{\max} - \frac{Jd_c^{\max}}{d_s^{\max}} - J + 1)}{\log[(d_s^{\max} - 1)(d_c^{\max} - 1)]} - 1 \right\rfloor + 2\right). \end{aligned} \tag{25}$$

The proof is completed.

Figure 2 depicts the lower bounds on PEG Tanner graphs and the lower bounds on FPEG Tanner graphs for regular $d_s^{\max} = 3$, $d_c^{\max} = 6$ codes and $d_s^{\max} = 4$, $d_c^{\max} = 8$ codes with varying m and $R = 0.5$. Note that due to the flooring operation in (1) and (25), there are almost vertical transitions shown in Figure 2. A step size of 100 was used when varying the value of m. It can be seen that, for the same $d_s^{\max}$ and $d_c^{\max}$, the PEG Tanner graph and the FPEG Tanner graph have similar lower bound on girth. It can also be seen that for small values of m, the girth can be 4. To avoid girth 4, equation below is used,

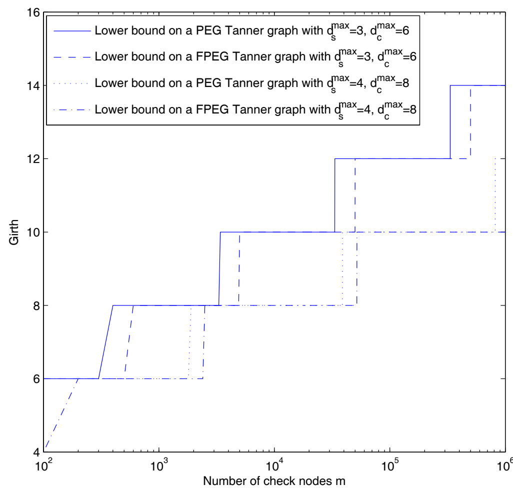

Figure 2 Lower bounds on two PEG Tanner graphs and two FPEG Tanner graphs. This figure depicts the lower bounds on PEG Tanner graphs and the lower bounds on FPEG Tanner graphs for regular $d_s^{\max} = 3$, $d_c^{\max} = 6$ codes and $d_s^{\max} = 4$, $d_c^{\max} = 8$ codes with varying m and $R = 0.5$.

$$2\left(\left\lfloor \frac{\log(Jd_c^{\max} - \frac{Jd_c^{\max}}{d_s^{\max}} - J + 1)}{\log[(d_s^{\max} - 1)(d_c^{\max} - 1)]} - 1 \right\rfloor + 2\right) \geq 6, \tag{26}$$

which can be simplified to

$$t = \frac{\log(Jd_c^{\max} - \frac{Jd_c^{\max}}{d_s^{\max}} - J + 1)}{\log[(d_s^{\max} - 1)(d_c^{\max} - 1)]} - 1 \geq 1, \tag{27}$$

that is

$$1 + d_s^{\max}(d_c^{\max} - 1) < r_{\min}. \tag{28}$$

Assume $r_{\min} = \left\lfloor \frac{m}{d_s^{\max}} \right\rfloor$, then

$$\begin{aligned} 1 + d_s^{\max}(d_c^{\max} - 1) &< \left\lfloor \frac{m}{d_s^{\max}} \right\rfloor \\ d_c^{\max} &< \left\lfloor \frac{\frac{m}{d_s^{\max}} - 1}{d_s^{\max}} + 1 \right\rfloor. \end{aligned} \tag{29}$$

Hereafter in this article, for a given m, it was assumed $M = d_s^{\max}$ for two reasons: First, from (16) it can be seen that the number of encoding steps is least when $M = d_s^{\max}$. Second, from (18) it is easy to see that J grows linearly with $r_{\min}$ and larger J value leads to larger lower bound on g_F. It is also easy to see that the maximum

$$r_{\min} = \left\lfloor \frac{m}{M} \right\rfloor \tag{30}$$

is achieved when $M = d_s^{\max}$. Therefore, the maximum g_F is achieved when $M = d_s^{\max}$.

Note that there are degree-1 symbol nodes in the corresponding FPEG Tanner graph and the fraction of degree-1 symbol nodes is r_1/n. The existence of degree-1 symbol nodes is a necessary condition for a linear-encoding algorithm such as the label-and-decide algorithm and the LPEG algorithm [5,12]. However, it was stated that the outbound extrinsic messages of degree-1 nodes would not be updated during the iterative decoding process [16,17]. Consequently, the degree-1 symbol nodes would cause many problems such as mismatching of extrinsic information transfer (EXIT) functions and the halting of mutual information evolution. In the following section, we will introduce a modified FPEG algorithm, which construct LDPC codes with only one degree-1 symbol node.

3.2 MFPEG algorithm

A simple modification of the FPEG algorithm can be applied to construct LDPC codes which have only one degree-1 symbol node. This modified algorithm is called

MFPEG algorithm in this article. The MFPEG algorithm is almost the same as the Algorithm 3 except for line 14. Therefore, we only describe the modified part in Algorithm 4.

Algorithm 4. MFPEG algorithm

```
1: Find out which check node group $G_i$ includes $c_i$.
2: if $i$ = 1 (the first check node group $G_1$) then
    $SC \leftarrow SC.$
3: else
    $SC \leftarrow SC \backslash G_i.$
4: end if
```

In a parity-check matrix corresponding to a Tanner graph generated by the MFPEG algorithm, the r_1 rows of the submatrix H_1 can have '1's in the same column. Consequently,

$$\{p_1, p_2, \ldots, p_{r_1}\}$$

should be computed serially, from p_{r_1} to p_1, in r_1 steps by the parity-check equations in H_1 and the parity bits

$$\left\{p_{\sum_{j=1}^{i-1} r_j+1}, p_{\sum_{j=1}^{i-1} r_j+2}, \ldots, p_{\sum_{j=1}^{i} r_j}\right\}$$

for $i = 2, 3, \ldots, d_s^{\max}$ are computed, in parallel, in one step by the parity-check equations in H_i. Totally, the number of encoding steps of the MFPEG codes is

$$\begin{aligned} T_M &= (d_s^{\max} - 1) + r_1 \\ &\geq (d_s^{\max} - 1) + r_{\min} \\ &\geq \left(d_s^{\max} + \left\lfloor \frac{m}{d_s^{\max}} \right\rfloor - 1\right), \end{aligned} \tag{31}$$

where the equality is achieved when r_1 is the minimum of $\{r_1, r_2, ..., r_M\}$. In fact, the MFPEG algorithm has loosened the condition (A) and is a combination of the LPEG algorithm and the FPEG algorithm. Therefore, for a given degree distribution pair, we have the following equalities:

$$g_F \leq g_M \leq g_L, \tag{32}$$

where g_M is the lower bound on girth of the MFPEG Tanner graph.

Note that, since both of LPEG codes and MFPEG codes have only one degree-1 symbol node (FPEG codes have r_1 degree-1 symbol nodes), they may have the same symbol-node-degree distribution. As shown in Figure 3, the encoder of the proposed FPEG and MFPEG codes can be implemented as r_i SPC encoder and one quasi-random interleaver, thus increasing complexity, compared to MPC encoder.

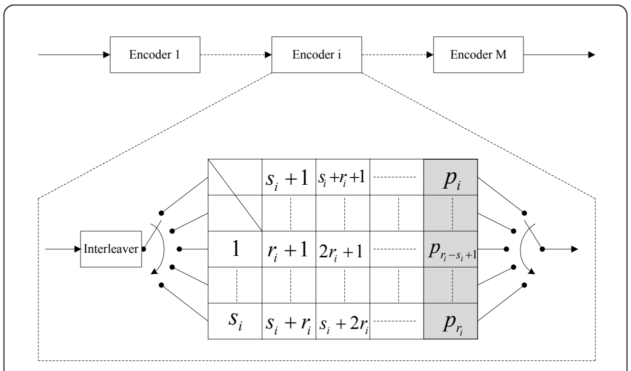

Figure 3 Structure of the proposed encoder. The encoder of the proposed FPEG and MFPEG codes can be implemented as r_i SPC encoder and one quasi-random interleaver.

4 Examples and simulation results

In this section, we provide two examples of the proposed codes and compare them with the M-SC-MPC codes [10] and LPEG codes. In this article, we denote LPEG codes, FPEG codes and MFPEG codes with M submatrices by M-LPEG codes, M-FPEG codes, and M-MFPEG codes respectively. In the first example it is shown that, In comparison to the M-SC-MPC codes which have the same number of encoding steps as FPEG codes, FPEG codes have better error correcting performance. In the second example, it is shown that for a given symbol-node-degree distribution, FPEG codes and MFPEG codes have similar error correcting performance but less encoding steps. In computing the error correcting performance, in terms of the bit error rate (BER), we assume BPSK transmission over the AWGN channel. The decoding algorithm used here is the log-likelihood sum-product algorithm and the maximum iteration number is set to be 50.

Example 2: An M-SC-MPC code consists of M MPC encoders and offers a flexible code rate and code length with low encoding complexity [10]. The encoding process of an M-SC-MPC code includes M steps, where M is also the maximum symbol-node degree, $d_s^{\max}$. Clearly, for the same maximum symbol-node degree, the number of encoding steps of M-SC-MPC codes is the same as those of FPEG codes. Therefore, we will compare the performance of FPEG codes and M-SC-MPC codes, provided that the number of encoding steps and the symbol-node-degree distributions are the same.

For the rate 3/4, in [10], the authors considered the following M-SC-MPC codes:

- 4-SC-MPC: n = 1196, r_1 = 59, r_2 = 73, r_3 = 78, r_4 = 89;
- 5-SC-MPC: n = 1268, r_1 = 53, r_2 = 55, r_3 = 59, r_4 = 67, r_5 = 83;

Table 1 Symbol-node-degree distributions of the M-SC-MPC codes

Code	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6
4-SC-MPC	0.0744	0.0652	0.0610	0.7993		
5-SC-MPC	0.0655	0.0528	0.0465	0.0434	0.7918	
6-SC-MPC	0.0510	0.0440	0.0410	0.0390	0.0382	0.7874

- 6-SC-MPC: n = 1204, r_1 = 45, r_2 = 46, r_3 = 47, r_4 = 49, r_5 = 53, r_6 = 61.

The symbol-node degree distribution and check-node degree distributions of these codes from node perspective are shown in Tables 1 and 2, respectively. Using the same symbol-node-degree distribution, we construct the M-FPEG codes with the FPEG algorithm for M = 4, 5, 6. The BER performance comparisons of these codes are given in Figure 4. It is shown that an M-FPEG code performs better than that of the corresponding M-SC-MPC code. At BER = 10^{-4}, the advantage of the M-FPEG codes against the M-SC-MPC codes are about 0.1, 0.7, and 0.3 dB, respectively, for M = 4, 5, 6.

Example 3: We construct M-LPEG codes, M-FPEG codes and M-MFPEG codes with the LPEG algorithm, FPEG algorithm and MFPEG algorithm, respectively, for $M = 4, 5, 6, (d_s^{\max} = 4, 5, 6)$. The code length is n = 1000, the parity-check length is m = 500 and the code rate is R = 0.5. For the purpose of performance comparison, the optimal degree distributions given in [[18], Table I] are used in this example. The fraction of Tanner graph edges which emanate from degree-1 symbol nodes of M-LPEG codes, M-FPEG codes and M-MFPEG codes can be obtained from that of the degree-i symbol nodes by the following formula:

$$\psi_1^* = \psi_i - \psi_i^*, \tag{33}$$

where ψ_i is the original fraction of Tanner graph edges which emanate from degree-i symbol nodes and ψ_i^* is the new fraction used in this example. Note that if the number of degree-2 symbol nodes of the Tanner graph of H is larger than or equal to n - m, any combination of n - m degree-2 columns forms cycles. Thus we can improve the error floor performance by reducing the fraction

$$\lambda_2 = \frac{\psi_2/2}{\sum_j \psi_j/j}, \tag{34}$$

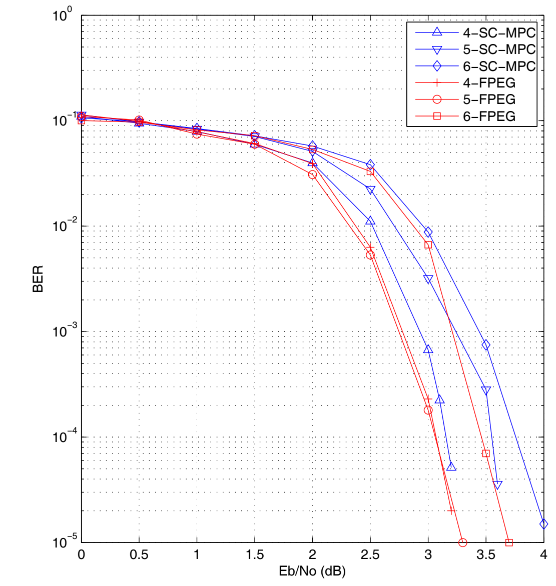

Figure 4 BER performance of FPEG codes in comparison with M-SC-MPC codes. The BER performance comparisons of these codes are given in Example 2.

if the fraction of degree-2 symbol nodes is higher than the optimal value given in [19]. Therefore, for the original degree distributions ψ_i given in [[18], Table I], we obtained the degree-1 symbol nodes from the degree-2 symbol nodes by

$$\psi_1^* = \psi_2 - \psi_2^*. \tag{35}$$

The original node degree distributions are given in the Table 3 and in the simulation of this example, the degree distributions from edge perspective are given in Table 4.

Note that the symbol-node-degree distributions from node perspective for the LPEG, FPEG, and MFPEG algorithms can be calculated from edge distribution with the formula (7) and the check-node-degree distribution is not needed as the check-node degrees are made as uniform as possible by the LPEG algorithm [6] and the proposed FPEG and MFPEG algorithms. From (1) and (18), it is clear that the maximum check-node degree $d_c^{\max}$ is necessary to derive the lower bound on girth. However, the value $d_c^{\max}$ can be obtained easily in the

Table 2 Check-node-degree distributions of M-SC-MPC codes

Code	ρ_{13}	ρ_{14}	ρ_{15}	ρ_{16}	ρ_{17}	ρ_{18}	ρ_{19}	ρ_{20}	ρ_{21}	ρ_{22}	ρ_{23}
4-SC-MPC	0.1672	0.5618	0.0736	0.1572	0.0401						
5-SC-MPC			0.1893	0.0726	0.0662	0.1640	0.4637	0.0442			
6-SC-MPC							0.0532	0.1495	0.2757	0.4585	0.0631

Table 3 Original symbol-node-degree distribution in [18]

d_s^{max}	ψ_2	ψ_3	ψ_4	ψ_5	ψ_6
4	0.3835	0.0424	0.5741		
5	0.3266	0.1196	0.1840	0.3699	
6	0.3324	0.2463	0.1102		0.3111

Table 4 New symbol-node-degree distribution

Code	ψ_1^*	ψ_2^*	ψ_3	ψ_4	ψ_5	ψ_6
4-LPEG	0.0010	0.3825	0.0424	0.5741		
5-LPEG	0.0010	0.3455	0.1196	0.1840	0.3699	
6-LPEG	0.0010	0.3314	0.2463	0.1102		0.3111
4-FPEG	0.1250	0.2585	0.0424	0.5741		
5-FPEG	0.1000	0.2466	0.1196	0.1839	0.3699	
6-FPEG	0.0830	0.2594	0.2463	0.1102		0.3111
4-MFPEG	0.0010	0.3825	0.0424	0.5741		
5-MFPEG	0.0010	0.3455	0.1196	0.1840	0.3699	
6-MFPEG	0.0010	0.3314	0.2463	0.1102		0.3111

simulation program. The lower bound on girth, girth and average cycle length of the simulated codes are given in Table 5 and their number of encoding steps are given in Table 6. From Equations (5), (16), (31), and Table 6, it can be seen that $T_F < T_M < T_L$ for these given symbol-node degree distributions. The simulation results are shown in Figure 5. It is shown that, compared to the LPEG codes, the FPEG codes perform better in the waterfall region but worse in the error floor region. However, the MFPEG codes perform similarly to the LPEG codes.

5 Conclusion

In this article, we introduced the FPEG algorithm and the MFPEG algorithm for generating fast encodable LDPC codes. The number of encoding steps of the FPEG codes grows linearly with $d_s^{\max}$, not the code length n. The number of encoding steps of the MFPEG codes grows linearly with $\left(d_s^{\max} + \left\lfloor \frac{m}{d_s^{\max}} \right\rfloor - 1\right)$. Moreover, we derived a lower bound on girth of the FPEG codes which is shown to be similar to that of PEG codes. By examples and simulations, it is shown that compared to the M-SC-MPC codes the FPEG codes have the same number of encoding steps but better error correcting performance, and compared to the LPEG codes the FPEG codes have similar error correcting performance but less encoding steps. Variants of M-SC-MPC codes, in which the degree of freedom is exploited, have been proposed in [20,21]. Considering these variants of M-SC-MPC codes in the design of fast-encodable LDPC codes is an open issue for future research.

Table 5 Girth lower bound, girth and average cycle length for *m* = 500, *n* = 1000, and *R* = 0.5

Code	d_c^{max}	Girth	Lower bound	Average cycle length
4-LPEG	8	7	6	8.02
5-LPEG	6	8	6	7.99
6-LPEG	6	8	6	7.486
4-FPEG	6	7	6	7.49
5-FPEG	6	8	6	7.308
6-FPEG	6	8	6	7.126
4-MFPEG	6	7	6	8.142
5-MFPEG	6	8	6	7.876
6-MFPEG	6	8	6	7.532

Table 6 Number of encoding steps For *m* = 500, *n* = 1000, and *R* = 0.5

d_s^{max}	T_L (LPEG)	T_F (FPEG)	T_M (MFPEG)
4	500	4	128
5	500	5	104
6	500	6	88

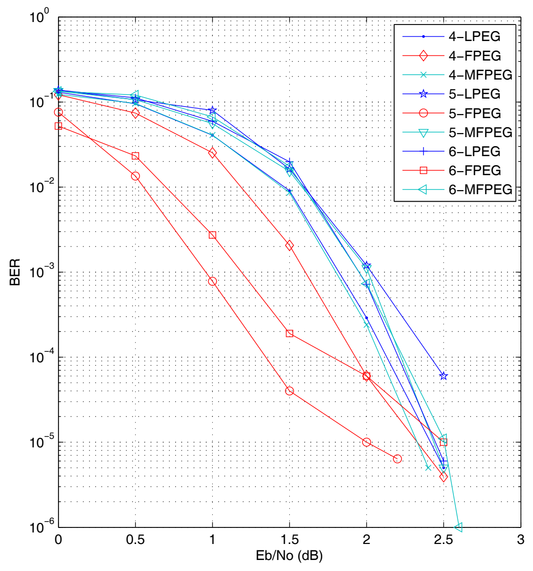

Figure 5 BER performance of FPEG codes and MFPEG codes in comparison with LPEG codes. The BER performance comparisons of these codes are given in Example 3.

Acknowledgements
This study was supported by World Class University (R32-2008-000-20014-0) NRF, Korea. It was also supported partly by the National Nature Science Foundation of China (61104154), Specialized Research Fund for Natural Science Foundation of Shanghai (10ZR1401600) and Fundamental Research Funds for the Central Universities (11D10417).

Author details
[1]Donghua University, School of Information Science and Technology, Shanghai, China [2]Donghua University, Engineering Research Center of Digitized Textile and Fashion Technology, Ministry of Education, Shanghai,

China [3]Chonbuk National University, Division of Electronics and Information Engineering, Jeonju, Korea

Competing interests
The authors declare that they have no competing interests.

Received: 14 May 2011 Accepted: 21 May 2012 Published: 21 May 2012

References

1. RG Gallager, Low density parity check codes. IEEE Trans Inf Theory. **IT-8**(1), 21–28 (1962)
2. DJC MacKay, RM Neal, Near Shannon limit performance of low density parity check codes. IEE Electron Lett. **32**(18), 1645–1646 (1996). doi:10.1049/el:19961141
3. J Feldman, MJ Wainwright, DR Karger, Using linear programming to decode binary linear codes. IEEE Trans Inf Theory. **51**(3), 954–972 (2005). doi:10.1109/TIT.2004.842696
4. MH Taghavi N, PH Siegel, Adaptive methods for linear programming decoding. IEEE Trans Inf Theory. **54**(12), 5396–5410 (2008)
5. XY Hu, E Eleftheriou, DM Arnold, Progressive edge-growth Tanner graphs, in *IEEE Global Telecommunications Conf. (GLOBECOM)*, San Antonio, TX, **2**, 995–1001 (2010)
6. XY Hu, E Eleftheriou, DM Arnold, Regular and irregular progressive edge-growth Tanner graphs. IEEE Trans Inf Theory. **51**(1), 386–398 (2005)
7. X Jiang, MH Lee, Large girth quasi-cyclic LDPC codes based on the chinese remainder theorem. IEEE Commun Lett. **13**(5), 342–344 (2009)
8. X Jiang, MH Lee, Large girth non-binary LDPC codes based on Euclidean geometries and finite fields. IEEE Signal Process Lett. **16**(6), 521–524 (2009)
9. D Haley, A Grant, J Buetefuer, Iterative encoding of low-density parity-check codes, in *IEEE Globecom 2002*, Taipei, Taiwan, Roc, **2**, 1289–1293 (2002)
10. M Baldi, G Cancellieri, A Carassai, F Chiaraluce, LDPC codes based on serially concatenated multiple parity-check codes. IEEE Commun Lett. **13**(2), 142–144 (2009)
11. RM Tanner, A recursive approach to low complexity codes. IEEE Trans Inf Theory. **IT-27**(6), 533–547 (1981)
12. J Lu, JMF Moura, Linear time encoding of LDPC codes. IEEE Trans Inf Theory. **56**(1), 233–249 (2010)
13. X Jiang, MH Lee, Semi-random LDPC codes with efficient encoding. IEE Electron Lett. **45**(24), 1259–1260 (2009). doi:10.1049/el.2009.2314
14. JSK Tee, DP Taylor, PA Martin, Multiple serial and parallel concatenated single parity-check codes. IEEE Trans Commun. **51**(10), 1666–1675 (2003). doi:10.1109/TCOMM.2003.818085
15. S Freundlich, D Burshtein, S Litsyn, Approximately lower triangular ensembles of LDPC codes with linear encoding complexity. IEEE Trans Inf Theory. **53**(4), 1484–1494 (2007)
16. G Yue, L Ping, X Wang, Generalized low-density parity-check codes based on Hadamard constraints. IEEE Trans Inf Theory. **53**(3), 1058–1079 (2007)
17. J Garcia-Frias, W Zhong, Approaching Shannon performance by iterative decoding of linear codes with low-density generator matrix. IEEE Commun Lett. **7**(6), 266–268 (2003)
18. T Richardson, MA Shokrollahi, R Urbanke, Design of capacityapproaching irregular low-density parity-check codes. IEEE Trans Inf Theory. **47**(2), 619–637 (2001). doi:10.1109/18.910578
19. W Weng, A Ramamoorthy, R Wesel, Lowering the error floors of irregular high-rate ldpc codes by graph conditioning, in *VTC*, Los Angeles, California, **4**, 2549–2553 (2004)
20. M Baldi, G Cancellieri, F Chiaraluce, A De Amicis, Design of permuted serially concatenated multiple parity-check codes, in *Proc SoftCOM*, Split, Croatia, **1**, 285–289 (2010)
21. M Baldi, G Cancellieri, F Chiaraluce, A De Amicis, Irregular M-SC-MPC codes for wireless applications, in *Proc 2010 European Wireless Conference, EW*, Lucca, Italy, **1**, 369–376 (2010)

doi:10.1186/1687-1499-2012-178
Cite this article as: Jiang *et al.*: Improved progressive edge-growth algorithm for fast encodable LDPC codes. *EURASIP Journal on Wireless Communications and Networking* 2012 **2012**:178.