Low Complexity Progressive Edge-Growth algorithm based on Chinese Remainder Theorem

Xueqin Jiang, Papa Ousmance Thiaw Diagne, Moon Ho Lee, *Senior Member, IEEE* and Wujun Xu

***Abstract*—Progressive edge-growth (PEG) algorithm construction builds the Tanner graph for an LDPC code by establishing edges between the symbol nodes and the check nodes in an edge-by-edge manner and maximizing the local girth in a greedy fashion. This approach is simple but the computational complexity of the PEG algorithm scale as $\mathcal{O}(nm)$, where n is the number of symbol nodes and m is the number of check nodes. We deal with this problem by first construct a base LDPC code of length n_1 with the PEG algorithm and then extend this LDPC code into an LDPC code of length n, where $n \gg n_1$, via the the chinese remainder theorem (CRT). This method increase the code length of an LDPC code generated with the PEG algorithm, without decreasing its girth. Due to the code length reducing in the PEG construction step, the computational complexity of the whole code construction process is reduced. Furthermore, the proposed algorithm have a potential advantage by storing a small parity-check matrix of a base code and extending it "on-the-fly" in hardware.**

***Index Terms*—LDPC codes, Progressive Edge-Growth (PEG) algorithm, Chinese Remainder Theorem (CRT), girth.**

I. Introduction

Chinese remainder theorem (CRT) [1][2][3] tells that a positive integer K can be uniquely reconstructed from its remainders modulo s positive integers, $L_1, L_2, \ldots, L_s$, if $K < \text{lcm}\{L_1, L_2, \ldots, L_s\}$ where lcm stands for the least common multiple, and furthermore it provides a simple reconstruction formula if all moduli L_i are co-prime.

Low-density parity-check (LDPC) codes, first proposed in the early 1960's [4] and re-discovered in 1996 [5], have recently attracted much attention due to their capacity-approaching performance. LDPC codes belong to the class of linear block codes, whose performance improves as the code length n becomes very large [8][9]. Based on the methods of construction, LDPC codes can be classified into two general categories: 1) structured codes; and 2) random (or random-like) codes. A method to extend the code length of quasi-cyclic (QC) LDPC codes, which is a class of structured LDPC codes via CRT was proposed in [10]. Codes result from this method have flexible code lengths, flexible code rates and, most importantly, large girths. Among the existing methods for the construction of random-like LDPC codes, one of the most successful approaches is progressive-edge-growth (PEG) algorithm proposed in [6][7]. The PEG algorithm is simple but not efficient. In the case of long codes, the construction of LDPC codes with this approach requires long time consumption. In the worst case, the computational complexity of the PEG algorithm scale as $\mathcal{O}(nm)$.

In this paper, motivated by the high computational complexity problem of the PEG algorithm and the simplicity of the extending method for QC LDPC codes via CRT, we combine the CRT with the PEG algorithm in the construction of random-like LDPC codes. Note that our proposed algorithm is not just extending the method used earlier in the construction of QC LDPC code to the random-like LDPC code. The proposed method extend a short base LDPC code, constructed by PEG algorithm, via CRT without reducing its girth. The simulation results show that the extended codes perform better than the base PEG LDPC codes and perform similar to long PEG LDPC codes of the same length with much less computational complexity.

The remainder of this paper is organized as follows. Section II reviews the PEG algorithm and the CRT. Section III proposes a method to combine the PEG algorithm and CRT. Lower bound on the girth of the PEG-CRT LDPC codes is also derived in this section. Section IV introduce an improved PEG-CRT algorithm. Section V presents examples of PEG-CRT LDPC codes and improved PEG-CRT LDPC codes and demonstrates their error-correcting performances. The complexity of the proposed algorithm is analysis in Section VI. Finally, Section VII concludes the paper.

II. Preliminaries

A. Progressive Edge-Growth Algorithm

Let H denote a sparse parity-check matrix having dimension $m \times n$ and let $V_c = \{c_0, c_1, ..., c_{m-1}\}$ denote the set of check nodes and $V_s = \{s_0, s_1, ..., s_{n-1}\}$ denotes the set of symbol notes. 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 i-th row and j-th column, $0 \leq i \leq m-1$, $0 \leq j \leq n-1$. In this algorithm, both symbol nodes and check nodes are ordered according to their degrees in nondecreasing order. d_{s_j} is the degree of symbol node s_j, and $N_{s_j}^l$ and $\bar{N}_{s_j}^l$ 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]. PEG algorithm constructs Tanner graphs having large girths and the lower bound on the girth g_p is proved to be

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

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

B. Chinese Remainder Theorem

Let K be a positive integer, $L_1, L_2, \ldots, L_s$ be s moduli, and $r_1, r_2, \ldots, r_s$ be s remainders of K, i.e.,

$$K \equiv r_i \text{ mod } L_i, \tag{2}$$

where $0 \le r_i < L_i$ for $1 \le i \le s$. It is not hard to see that K can be uniquely reconstructed from its s remainders if and only if $0 \le K < \text{lcm}\{L_1, L_2, \ldots, L_s\}$. If all the moduli L_i are co-prime and $L = L_1 L_2 ... L_s$, then CRT has a simple formula

$$K = \sum_{i=1}^{s} r_i A_i \bar{L}_i \text{ mod } L, \tag{3}$$

where $\bar{L}_i = L/L_i$ and $A_i \bar{L}_i \equiv 1 \text{ mod } L_i$. If any pair moduli L_i have gcd M, then the CRT has a general form [3].

III. Combining of PEG algorithm and CRT

Let $L_1, L_2, \ldots, L_s$ be s distinct integers such that any pair L_i have gcd 1 and $L = L_1 L_2 \cdots L_s$. In this section, we will first introduce the PEG-CRT algorithm and then analyze some of its important properties.

A. PEG-CRT algorithm

Before introducing the PEG-CRT algorithm, let us first give an important lemma.

Lemma 3.1: Let $L_1, L_2, \ldots, L_s$ be s moduli, and r_{ij} be a remainder modulo L_j, $1 \le j \le s$. Given a non-negative integer r, $r < L_1$, there are $\bar{L}_1 = L_2 L_3 ... L_s$ different sequences

$$\begin{aligned} a_1 &= (r_{11}, r_{12}, \ldots, r_{1s}), \\ a_2 &= (r_{21}, r_{22}, \ldots, r_{2s}), \\ &\vdots \qquad \vdots \qquad \vdots \\ a_{\bar{L}_1} &= (r_{\bar{L}_1 1}, r_{\bar{L}_1 2}, \ldots, r_{\bar{L}_1 s}), \end{aligned} \tag{4}$$

where $r_{i1} = r$.

Proof: For a given r, since $0 \le r_{ij} \le (L_j - 1)$ for $1 \le i \le \bar{L}_1$, $2 \le j \le s$, there are $L_2 L_3 \cdots L_s$ different sequences $(r, r_{i2}, r_{i3}, \cdots, r_{is})$. Therefore, it is possible to have one-to-one mapping between $a_1, a_2, \ldots, a_{\bar{L}_1}$ and $L_2 L_3 \cdots L_s$ different sequences, where $\bar{L}_1 = L_2 L_3 \cdots L_s$. The proof is completed.

Let $m_1 = L_1$ and $m = L$, the PEG-CRT algorithm is given as follows:

- Step 1: All the entries of the $m \times n$ parity-check matrix H are set to be '0's.
- Step 2: Generate an LDPC code C_1, whose parity-check matrix H_1 is of size $m_1 \times n_1$, with the PEG algorithm.
- Step 3: Let $R = (n_1 - m_1)/n_1$ be the code rate of the base code. For each '1' entry in the r-th row and c-th column of H_1, where $0 \le r < L_1$ and $0 \le c < L_1/(1-R)$, generate $\bar{L}_1 = L/L_1$ sequences

$$\begin{aligned} a_1 &= (r_{11}, r_{12}, \ldots, r_{1s}), \\ a_2 &= (r_{21}, r_{22}, \ldots, r_{2s}), \\ &\vdots \qquad \vdots \qquad \vdots \\ a_{\bar{L}_1} &= (r_{\bar{L}_1 1}, r_{\bar{L}_1 2}, \ldots, r_{\bar{L}_1 s}), \end{aligned} \tag{5}$$

  where $r_{i1} = r$ and $0 \le r_{ij} \le (L_j - 1)$ for $1 \le i \le \bar{L}_1$, $2 \le j \le s$.

$$a_k \ne a_l, \ \forall \ k \ne l. \tag{6}$$

  Note that the possibility for (6) to hold is proofed in lemma 3.1.
- Step 4: For $1 \le i \le \bar{L}_1$, compute new row indices K_i with entries $r_{i1}, r_{i2}, \ldots, r_{is}$ in a_i, respectively, by

$$\bar{K}_i = \sum_{j=2}^{s} r_{ij} A_j \bar{L}_j \text{ mod } L \tag{7}$$

  and

$$K_i = \left(\bar{K}_i + r_{i1} A_1 \bar{L}_1\right) \text{ mod } L, \tag{8}$$

  where $\bar{L}_j = L/L_j$ and $A_j \bar{L}_j \equiv 1 \text{ mod } L_j$.
- Step 5: With row index K_i and column index

$$J_i = c * \bar{L}_1 + i, \tag{9}$$

  where $0 \le i < \bar{L}_1$, set the entry in the K_i-th row and J_i-th column of H to be '1'.

B. Properties of PEG-CRT LDPC Codes

Consequently we expand each one '1' entry of H_1 into $\bar{L}_1$ '1' entries of H. When we expand each '1' entry in c-th column into $\bar{L}_1$ '1' entries, the c-th column is expanded into $\bar{L}_1$ columns. Finally, the resulting matrix is a new parity-check matrix H which has the same density as that of H_1. The corresponding code length is

$$\begin{aligned} n &= L/(1-R) \\ &= L_1 L_2 \cdots L_s/(1-R) \\ &= n_1 L_2 \cdots L_s. \end{aligned} \tag{10}$$

We further summarize the above PEG-CRT algorithm in **Algorithm 1**. Note that the proposed PEG-CRT algorithm can also be applied when all pair of L_i have gcd M.

Theorem 3.2: For Letting g_1 denote the girth of a short PEG LDPC code C_1 and g denotes the girth of an LDPC code C extended from C_1 via CRT, then we have

$$g \ge g_1. \tag{11}$$

Proof: Assume that in the parity-check matrix H_1, there is a '1', p_1 in a cycle whose length is g_1 and the row index of p_1 is r_1. Assume that there is another '1' p_2 out side of this cycle and the row index of p_2 is r_2. Also assume that K_1, K_2 are three new row indices generated from p_1, p_2 respectively via CRT. Since K_1, K_2 are uniquely reconstructed from

$$\begin{aligned} a_1 &= (r_1, r_{12}, \ldots, r_{1s}), \\ a_2 &= (r_2, r_{22}, \ldots, r_{2s}), \end{aligned} \tag{12}$$

via CRT, respectively, $K_2 \ne K_1$. Therefore, after expanding the parity-check matrix H_1 via CRT, the '1's in H corresponding to p_2 are not in the same row of the '1's corresponding to p_1 and therefore the '1's corresponding to K_1 and K_2 do not make new cycles shorter than g_1. Consequently, the girth is not reduced. The proof is completed.

Algorithm 1 PEG-CRT algorithm

Preprocessing (Carry out only once):

Assume m_1, n_1 be the number of rows and columns of the base matrix H_b and m, n be the number of rows and columns of the extended matrix H. Initially, all the entries of H are '0's. For a given degree distribution, generate a Tanner graph with $m_1 = L_1$ check nodes and $n_1 = L_1/(1-R)$ symbol nodes with PEG algorithm. Construct the corresponding $m_1 \times n_1$ parity-check matrix H_1. Generate $\bar{L}_1 = L/L_1$ sequences $a_i = (r_{i1}, r_{i2}, \ldots, r_{is})$. Compute $\bar{K}_1, \bar{K}_2, \ldots, \bar{K}_{\bar{L}_1}$ with (7).

Parity-Check Matrix Expanding:

for $j = 0$ to $n-1$ **do**

  for $k = 0$ to $d_{s_j} - 1$ **do**

    for $i = 0$ to $\bar{L}_1$ **do**

    $K_i = \left(\bar{K}_i + r_{i1} A_1 \bar{L}_1\right) \mod L$

    $J_i = c * \bar{L}_1 + i.$

    Set the entry in the K_ith row and J_ith column of H to be '1'.

    end for

  end for

end for

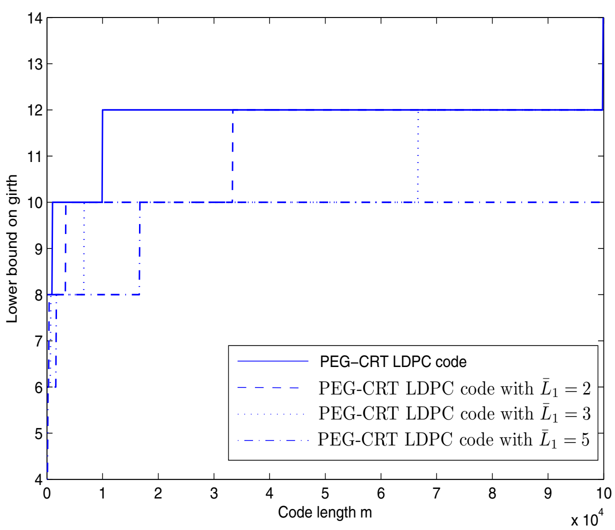

Fig. 1. Lower bounds on girth of PEG codes and PEG-CRT codes with $d_s^{max} = 3$, $d_c^{max} = 6$.

PEG-CRT algorithm constructs Tanner graphs having large girths and the lower bound on the girth g_c is proved to be

$$g_c \geq 2\left(\left\lfloor \frac{\log(\bar{m} d_c^{max} - \frac{\bar{m} d_c^{max}}{d_s^{max}} - \bar{m} + 1)}{\log[(d_s^{max}-1)(d_c^{max}-1)]} - 1 \right\rfloor + 2\right), \tag{13}$$

where $\bar{m} = m/L_1$. From (1), proof is immediate and omitted. Fig. 1 depicts the lower bounds on PEG Tanner graphs and PEG-CRT Tanner graphs for regular $d_s^{max} = 3$, $d_c^{max} = 6$ codes with varying m and rate 1/2.

Note that if all the '1's of H_1 are replaced by the same vectors $a_1, a_2, \ldots, a_{\bar{L}_1}$, then the result parity-check matrix H consists of $\bar{L}_1$ separate submatrices, which means that the Tanner graph of H consists of $\bar{L}_1$ separate subgraphs. The extrinsic information will not be exchanged among the symbol nodes and check nodes among these subgraphs. The problem can be solved by two methods. The first method is to permute the row indices $K_1, K_2, \ldots, K_{\bar{L}_1}$ when expanding each '1' in H_1. It can be realized by randomly permute the orders of vectors $a_1, a_2, \ldots, a_{\bar{L}_1}$ in (5) or permute the orders of $K_1, K_2, \ldots, K_{\bar{L}_1}$. Hence, we have to compute $K_1, K_2, \ldots, K_{\bar{L}_1}$ for only once in a initialization step and randomly permute them when we expand each '1' entry of in the same row of H_1. The second method is to permute the order of column indices $J_1, J_2, \ldots, J_{\bar{L}_1}$. The LDPC codes of length L constructed by PEG-CRT algorithm correspond to $\bar{L}_1$ short LDPC code of length L_1 interleaved together.

IV. Improved PEG-CRT Algorithm

The LDPC codes of length L, which are not simply $\bar{L}_1$ LDPC codes of length L_1 interleaved together, will be introduced in this section. Improved PEG-CRT algorithm not only preserves all the good properties of PEG-CRT algorithm such as girth and complexity, but also has better error-correcting performance, which will be shown in next section.

We represent by "$\oplus$" the modulo $\bar{L}_1$ summation. For each '1' entry in the r-th row and c-th column of H_1, we first generate a random integer x, $0 \leq x \leq (\bar{L}_1 - 1)$. Then we generate $\bar{L}_1$ sequences

$$\begin{aligned} a_1 &= (r_{(1\oplus x)1}, r_{(1\oplus x)2}, \ldots, r_{(1\oplus x)s}), \\ a_2 &= (r_{(2\oplus x)1}, r_{(2\oplus x)2}, \ldots, r_{(2\oplus x)s}), \\ &\vdots \qquad \vdots \qquad \vdots \\ a_{\bar{L}_1} &= (r_{(\bar{L}_1\oplus x)1}, r_{(\bar{L}_1\oplus x)2}, \ldots, r_{(\bar{L}_1\oplus x)s}), \end{aligned} \tag{14}$$

and compute new row indices K_i with entries $(i \oplus x)1, r_{(i\oplus x)2}, \ldots, r_{(i\oplus x)s}$ in a_i by

$$\bar{K}_i = \sum_{j=2}^{s} r_{(i\oplus x)j} A_j \bar{L}_j \mod L \tag{15}$$

and

$$K_i = \left(\bar{K}_i + r_{(i\oplus x)1} A_1 \bar{L}_1\right) \mod L. \tag{16}$$

Equivalently, we can use $K_{(1\oplus x)}, K_{(2\oplus x)}, \ldots, K_{(\bar{L}_1\oplus x)}$ as the row indices with column indices $J_1, J_2, \ldots, J_{\bar{L}_1}$. Or use $J_{(1\oplus x)}, J_{(2\oplus x)}, \ldots, J_{(\bar{L}_1\oplus x)}$ as column indices with row indices $K_1, K_2, \ldots, K_{\bar{L}_1}$ (cyclic shift the $\bar{L}_1$ columns to the left by x positions). The improved PEG-CRT algorithm is described in **Algorithm 2**. Note that we decompose the formula (3) into (7) and (8) (or (14) and (15)) to lower the computational complexity. Since (7) (or (14)) is a common part in the proposed algorithm and which can be put into preprocessing step.

Given the degree distribution of the PEG LDPC code C_1 and the corresponding PEG-CRT LDPC code C, compared to the PEG-CRT algorithm, the improved PEG-CRT algorithm need one random x generator and $\bar{L}_1 s$ $\oplus$ summation. The

Algorithm 2 Improved PEG-CRT algorithm

Preprocessing (Carry out only once):
The same as the preprocessing in **Algorithm 1**.
Parity-Check Matrix Expanding:
for $j = 0$ to $n - 1$ **do**
  for $k = 0$ to $d_{s_j} - 1$ **do**
    Generate a random integer x.
    for $i = 0$ to $\bar{L}_1$ **do**
    $K_i = \left(\bar{K}_i + r_{i1} A_1 \bar{L}_1\right) \mod L$
    $J_i = c * \bar{L}_1 + (i \oplus x)$. (Cyclic shift the $\bar{L}_1$ columns to the left by x positions)
    Set the entry in the K_i-th row and J_i-th column of H to be '1'.
    end for
  end for
end for

increase of complexity is neglectable when the density of the parity-check matrix is very low.

V. Code Performance

In this section, we provide two examples of the proposed codes and compare them with the PEG LDPC codes [6] in the Matlab simulations.

Example 1: In this example, we study the error correcting performance of PEG-CRT LDPC codes by means of simulations. For comparison purposes, we use a rate-1/2 PEG LDPC code of length 504 in [6] (example in Fig.5), which is based on a Tanner graph with uniform degree 3 for each symbol nodes. We first construct a rate-1/2 PEG LDPC code of length $n_1 = 502$ ($L_1 = 251$ is a prime number). Based on this short PEG code, PEG-CRT LDPC codes of length 3012 ($L_2 = 2$, $L_3 = 3$ and $n_1 L_2 L_3 = 3012$) and 7530 ($L_2 = 3$, $L_3 = 5$ and $n_1 L_2 L_3 = 7530$), are constructed, respectively. With the same symbol-node-degree distribution, we also construct rate-1/2 PEG LDPC codes of length 3012 and 7530 respectively for comparison. 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 belief propagation algorithm and the maximum iteration number is set to be 80. Fig.2 compares the BERs for the six codes.

We collect at least 100 frame errors per simulation point. The proposed PEG-CRT LDPC codes perform much better than the original short base PEG code and perform similar to PEG LDPC codes of the same length. However, it will be shown in the next section that compared to long PEG LDPC codes, the improved PEG-CRT LDPC codes have much lower complexities.

Example 2: In this example, we investigate the performance of symbol-node-degree distributions as given in [8 Table II]. We first construct a rate-1/2 PEG LDPC code of length 502 ($L_1 = 251$) with symbol-node-degree distribution and maximum symbol-node degree 15. Based on this short PEG code, improved PEG-CRT LDPC codes of length 3012 ($L_2 = 2$, $L_3 = 3$) is constructed. With the same symbol-node degree distribution, we also construct rate-1/2 PEG LDPC codes of length 504 and 3012, respectively for comparison. The improved PEG-CRT LDPC codes perform much better than the original short PEG code and perform similar to PEG LDPC codes of the same length. Fig.3 compares the BERs for these codes.

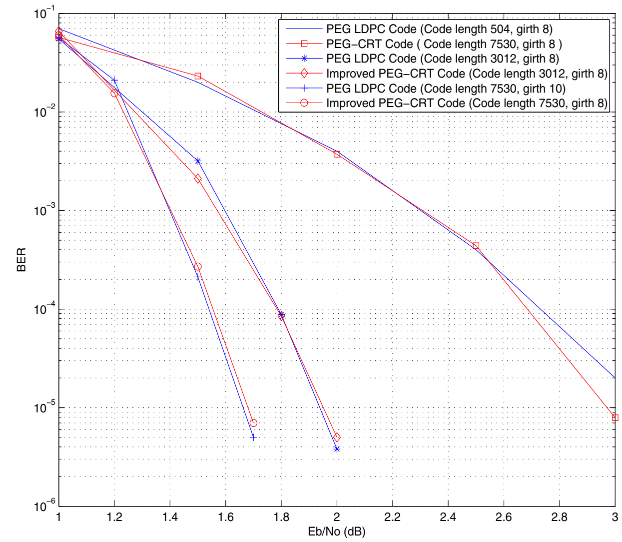

Fig. 2. BER performance of PEG codes and proposed PEG-CRT codes

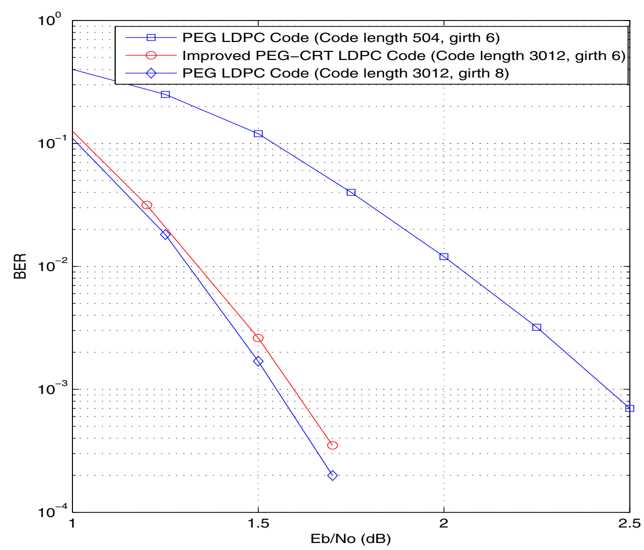

Fig. 3. BER performance of PEG codes and proposed PEG-CRT codes

VI. Complexity Analysis and Comparison

The computational complexity of the PEG algorithm primarily depends on the computational load in obtaining the set $N_{s_j}^l$ or $\bar{N}_{s_j}^l$ which depends on the degree sequences D_s and D_c as well as on the depth l. In a sparse graph, the elements of D_s and D_c are small numbers irrespectively of

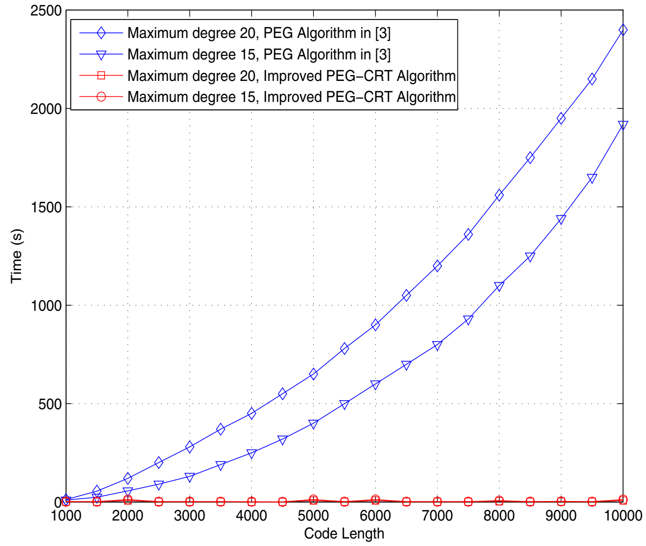

Fig. 4. Construction time comparison of PEG algorithm and PEG-CRT algorithm.

n, and l grows at most logarithmically with m. In the worst case, the computational complexity of the PEG algorithm scale as $\mathcal{O}(nm)$. The computational complexity of the PEG-CRT algorithm includes two parts. The first part is the complexity of the PEG algorithm used to construct the short LDPC codes, which does not grow as the size of H grow. The second part is the complexity of the CRT computation, which only depends on the number of '1's in H_1. Therefore, the complexity of PEG-CRT algorithm only depends on H_1, and does not change while the size of H is growing. To compare the time complexities of the PEG algorithm and improved PEG-CRT algorithm, we constructed two groups of rate-1/2 LDPC codes with two different degree distributions. These two degree distributions, with maximum symbol-node degrees of 15 and 20, are all from table II in [8]. A comparison of the construction time by the same computer is shown in Fig.4. The curves of the PEG algorithm increase much faster than our proposed improved PEG-CRT algorithm. The complexity of our proposed algorithm is almost not changing because the complexity of our algorithm mainly depends on the construction of the small Tanner graph whose size does not change.

The row and column indices of nonzero entries in a parity check matrix will be pre-computed and stored in the practical implementation. From this point of view, the proposed algorithm have a potential advantage by storing the small parity check matrix of a base code and extending it "on-the-fly" [11][12] in hardware, which may reduce the storage space.

VII. Conclusion

The PEG algorithm is simple but the computational complexity of the PEG algorithm scale as $\mathcal{O}(nm)$. We deal with this problem by first construct a short LDPC code with the PEG algorithm and then extend the short PEG LDPC code into a long LDPC code via the the CRT. Due to the code length reducing in the PEG algorithm, the computational complexity is reduced. Furthermore, the storage space of the decoder may be reduced by storing a smaller parity-check matrix H_b. The proposed method increase the code length of a short PEG LDPC code without decreasing the girth. The complexity of this proposed algorithm is analyzed. Simulation results show that the proposed PEG-CRT LDPC codes perform much better than the original short PEG code and perform similar to PEG LDPC codes of the same length. But compared to long PEG LDPC codes, the improved PEG-CRT LDPC codes have much lower complexities.

Acknowledgment

This work was supported by World Class University R32-2008-000-20014-0, Fundamental Research 2010-0020942 NRF, EU-FP KIAT, Korea. This work is also supported in part by Yangtze River Delta Development Joint-Program (10595812200) and Fundamental Research Funds for the Central University (12D10417).

References

[1] J. H. McClellan and C. M. Rader, *Number in Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1979.
[2] C. Ding, D. Pei and A. Salomaa, *Chinese Remainder Theorem: Applications in Computing, Coding, Cryptography*. Singapore: World Scientific, 1999.
[3] W. Wang and X.-G. Xia, "A Closed-Form Robust Chinese Remainder Theorem and Its Performance Analysis," *IEEE. Trans. on Signal Processing*, vol. 58, pp. 5655-5666, no. 11, Nov. 2010.
[4] R. G. Gallager,"Low density parity check codes," *IEEE Trans. Inform. Theory*, vol. IT-8, no. 1, pp. 21-28, Jan. 1962.
[5] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *IEE Electron. Lett*, vol. 32, no. 18, pp. 1645-1646, Aug. 1996.
[6] X. Y. Hu, E. Eleftheriou, and D. M. Arnold, "Progressive edge-growth Tanner graphs," *IEEE Global Telecommunications Conf.(GLOBECOM)*, San Antonio, TX, Nov. 2001.
[7] Xueqin Jiang, Mi Sung Lee and Moon Ho Lee, "Modified Progressive Edge-Growth Algorithm for Fast Encoding LDPC Codes," *IEEE VTC2010 fall*, Ottawa, Canada, Sep. 2010.
[8] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of Capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 619-637, Feb. 2001.
[9] S. Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045dB of the Shannon limit," *IEEE Commun. Lett.*, vol. 5, pp. 58-60, Feb. 2001.
[10] Xueqin Jiang and Moon Ho Lee, "Large Girth Quasi-Cyclic LDPC Codes based on the Chinese Remainder Theorem," *IEEE Commun. Lett.*, vol. 13, no. 5, pp.342-344, May. 2009.
[11] D. Oh and K. Parhi,"Low-Complexity Switch Network for Reconfigurable LDPC Decoders,"*IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 1, pp. 85-94, Jan. 2010.
[12] D. Bao, B. Xiang, R. Shen, A. Pan, Y. Chen and X. Zeng,"Programmable Architecture for Flexi-Mode QC-LDPC Decoder Supporting Wireless LAN/MAN Applications and Beyond,"*IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 57, no. 1, pp. 125-138, Jan. 2010.