Design of Low-Rate Irregular LDPC Codes Using Trellis Search

Sang Hyun Lee, *Member, IEEE,* and Kwang Soon Kim, *Senior Member, IEEE*

***Abstract*—A simple design method using trellis search is proposed for good low-density parity-check (LDPC) codes with relatively low code rates. By applying a trellis search technique to the design of a pre-assigned part of the parity-check matrix that allows a simple encoding, we improve the distribution of cycles formed by the entries contained in the parity-check part of the parity-check matrix. In addition, we extend the proposed algorithm to a class of structured LDPC codes, which have been recently preferred in many practical applications. Simulation results show that the codes designed by the proposed method outperform those constructed by conventionally used greedy design algorithms.**

***Index Terms*—Channel coding, LDPC codes, trellis search.**

I. Introduction

RECENTLY, LDPC codes have been employed in a variety of practical applications including mobile communications with a link adaptation technique, in which a suite of various channel codes is used [1]. Although such codes have different code rates and codeword lengths, it is commonly preferred for a simple encoding that their corresponding parity-check matrices contain a lower triangular sub-matrix [2], with the main diagonal and the first sub-diagonal entries filled with nonzero entries. Henceforth, this sub-matrix structure is referred to as an echelon form[1]. Also, it usually consists of low-degree columns and is used for the parity-check part of the parity-check matrix.

In [3] and [4], it has been reported that a special echelon form for the parity-check part consisting of degree-two columns helps not only allowing a simple encoding, but also keeping the minimum distance and the smallest cycle length as large as possible for high-rate codes. However, it is generally preferred to optimize the degree distribution of a code for better performance by using several techniques such as density evolution [5]. The result of the optimization most likely leads to the occurrence of columns with degree of more than two in the parity-check part as the code rate decreases [6][7]. As given in [6], most optimized degree distributions for code rates less than 0.6 have less degree-two bit nodes than parity-check nodes. For instance, the optimized proportion of degree-two bit nodes for 1/3-rate codes is around 0.5, while it should have been more than 2/3 for the design based on [4]. Thus, for low-rate codes, which will be addressed mainly in this paper, the parity-check part is set as an echelon form for easy encoding, in which columns with degree more than two are very likely to be included. In such cases, the parity-check part does not turn out to be of trivial form as given in [4], and greedy methods, such as the progressive edge growth (PEG) algorithm [8], have been applied for the design in previous literature. The greedy construction methods usually consider only the girth as the performance measure, and their results are often far from the asymptotic results. Since the girth is a single parameter representing the global structure of the graph, it can hardly grasp local structures that may impair iterative decoding. Thus, it is expected that the performance may be improved by employing a performance measure which can characterize more local structures of the graph. To come closer to the expected optimal result, we will consider a ‘distribution’ of cycles as the performance measure for the code design.

In this paper, we focus on the following facts: once we adopt the echelon form as the parity part, most of edges are assigned in advance (along the two diagonals), and only a small portion of edges needs to be configured in the design procedure. Furthermore, the existence and the lengths of cycles formed by parity-check nodes can be straightforwardly examined by calculating the ‘distance’ between nonzero entries in the echelon form, not by spanning a search tree as done in conventional construction methods. In particular, for lower-rate codes mainly focused in this paper, most part of the parity-check matrix is occupied by the echelon form. Thus, the number of entries (corresponding to the message part in the parity-check matrix) designed by a greedy construction is reduced, and the overall decoding performance can also be controlled by the configuration in the echelon form. Since an implicit estimation of the decoding performance is available from the ‘distribution’ of cycles in the corresponding graph, the positions of configurable entries in the parity-check matrix can be determined so as to maximize the lengths of associated cycles. Note that it is already known that such an optimization problem cannot be solved in a polynomial time. Let us represent the column-by-column edge placement in the parity part of the parity-check matrix as a trellis search with a metric representing the distribution of cycles caused from the placement. Then, for the optimal design, the states for each

Paper approved by O. Milenkovic, the Editor for Coding Theory and Applications of the IEEE Communications Society. Manuscript received February 25, 2007; revised December 18, 2007 and June 26, 2008.

Initial work of this paper was presented at the International Symposium on Information Theory and Its Applications (ISITA), Seoul, Korea, 2006.

This research was supported by the Ministry of Knowledge Economy, Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Advancement) (IITA-2008-C1090-0803-0002).

S. H. Lee is with the Electronics and Telecommunications Research Institute, Daejeon, Korea (e-mail: sanghlee@ieee.org).

K. S. Kim is with with the Department of Electrical and Computer Engineering, Yonsei University, Seoul, Korea (e-mail: ks.kim@yonsei.ac.kr).

Digital Object Identifier 10.1109/TCOMM.2009.07.070088

[1] Although the common definition of the echelon form does not require the existence of the two nonzero diagonal entries, it is used in this paper for simplicity.

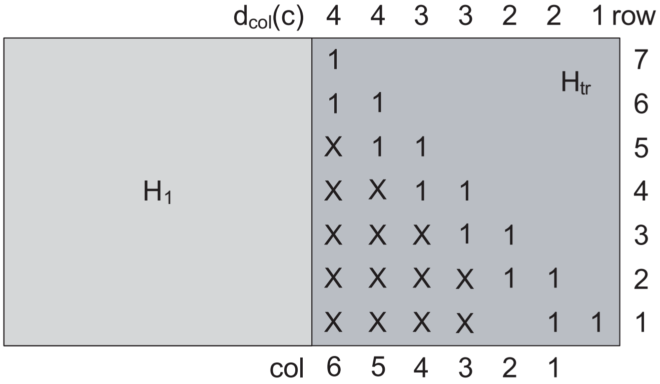


Fig. 1. The representation of the echelon form.

column in the trellis should be all combinations of possible previous edge placements, which implies that the complexity grows exponentially with the number of edges. Also, note that a PEG-like construction can be considered as a simple algorithm using the following two relaxations: i) for the states of each column, all combinations of possible edge placements in the column are considered ii) only one surviving path is greedily selected for the current column.

In this paper, we adopt the first relaxation for a polynomial-time complexity but do not use the second one. In this case, we can apply a trellis search to solve the relaxed optimization problem. We empirically observed that the solution of the relaxed problem is often found to be the same as that of the original problem in the case of relatively small dimensions. The initial work of this paper has been presented in [9], and the proposed algorithm is extended to the case of structured codes in this paper. The rest of this paper is organized as follows. Section II proposes the structure of the proposed parity-check matrix and the algorithm, Section III extends the idea of the proposed algorithm to a class of structured LDPC codes, Section IV studies the complexity of the proposed algorithm and presents some code design examples, and Section V concludes this paper.

II. The Proposed Algorithm

For an LDPC code of our interest, the parity-check matrix, $H = [H_1 \ H_{tr}]$, is partitioned into two sub-structures, where H_{tr} is an echelon form denoting the sub-matrix containing all columns associated with parity-check bits, and H_1 is a sub-matrix containing all columns associated with message bits. Note that H_{tr} needs not be a 'square' triangular matrix with the rightmost degree-one column but can be any 'rectangular' triangular matrix where the number of columns is less than or equal to that of rows. For the sake of easy understanding, we explain the proposed algorithm with a simple example. However, the following assertions are still valid for any parity-check matrix whose rightmost part is an echelon form. Fig. 1 shows a parity-check matrix with the echelon form of seven rows and columns. Here, $d_{col}(\mathbf{c})$ is a sequence of column degrees of the column vector $\mathbf{c}$ contained in the echelon form. Then, the column c with the degree of at least two has $d_{col}(c) - 2$ remaining entries which have not been located yet. The $\times$'s in each column of H_{tr} represent available positions for the remaining entries. Note that the available positions of the remaining entries are restricted to the sub-diagonal part in order to maintain the simple encoding feature. The indices of rows and columns, row and col, denote the positions of the $\times$'s in H_{tr}, respectively.

The graph associated with an echelon form contains a 'chain'-like subgraph where a bit node is connected to a check node, and the check node is subsequently connected to another bit node. Thus, any additional edge connecting a bit node and a check node in the chain incurs a cycle. The bit nodes associated with the far right columns of the echelon form are of the lowest degree, thereby having the least possible positions for edges. Since new edges connect only nodes in the echelon form, the far right columns are more likely associated with small cycles, which will result in critical impacts on the decoding performance. Therefore, the examination of cycles incurred by a new entry in H_{tr} commences at the rightmost column which has at least one remaining entry to be located. Since the positions of new entries are closely related to the lengths of cycles containing those entries, the examination is performed for all possible positions in each column with a proper measure, and the evaluated measures are compared to determine the best position. As the location goes on, the addition of a new entry incurs additional cycles formed by the new entry and some previously located entries up to the current column. To consider all cycles incurred by a new entry, the proposed measure function evaluates i) cycles formed by the new entry and diagonal entries, ii) cycles formed by the new entry and some previously located entries, and iii) cycles formed by new entries in the current column, some previously located entries, and diagonal entries. By comparing the measures for feasible positions of the current entry, the best position is determined. This procedure is repeated until the positions of all entries are determined.

By a brief inspection, no new entry can avoid incurring cycles containing it, and the lengths of such cycles are evaluated from its position. Here, we add a structural constraint such that, for any two (not necessarily adjacent) columns with at least two additional entries to be located, the row indices of any two entries of one column should not be the same as those of any two entries of the other column. Note that this constraint not only prevents very short cycles formed only by those additional entries, but also helps to distribute check nodes as even as possible. This constraint will be taken into account for the definition of measure functions and the construction of a trellis.

Now, define the three different classes of cycles as follows.

Definition 1: A Class-1 cycle is a cycle formed only by a single new entry and entries on the two diagonals.

Definition 2: A Class-2 cycle is a cycle formed among a single new entry, previously located entries, and entries on the two diagonals.

Definition 3: A Class-3 cycle is a cycle formed by two new entries in the current column, some previously located entries in the previous columns, and entries on the two diagonals.

Then, it is easily seen that, for an echelon form designed up to a certain column, any cycle caused by entries in that column falls into one of the above three classes.

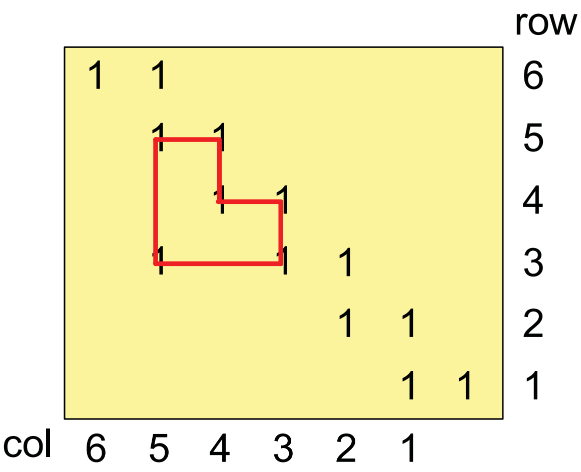


Fig. 2. An example of Class-1 cycle.

A. Class-1 Cycles

Fig. 2 depicts an example of Class-1 cycles. Here, new entry is placed at the third row and the fifth column. We see that the length of the cycle depends only on the difference between the row indices of a new entry and the first sub-diagonal entry in the row of the new entry. Thus, the length of the cycle can be easily calculated as follows.

Theorem 1: Let $\mathbf{e}_q^l$ denote the edge corresponding to the qth new entry of the lth column. Then, the length of the Class-1 cycle caused by placing $\mathbf{e}_q^l$, $L_1(\mathbf{e}_q^l)$, is given by

$$L_1(\mathbf{e}_q^l) = 2(\Delta_r(\mathbf{e}_q^l) + 1), \tag{1}$$

where $\Delta_r(\mathbf{e}_q^l)$ is the absolute value of the difference between the row indices of $\mathbf{e}_q^l$ and the first sub-diagonal entry in the same row.

Proof: Define a 'zigzagger' as a pair of two connected line segments: one horizontally connecting an entry on the first sub-diagonal to the main diagonal entry of the same row and the other vertically connecting the main diagonal entry to the first sub-diagonal entry of the same column. Then, we see that a zigzagger corresponds to the move by two edges along the diagonals of an echelon form. The number of zigzaggers in the loop of a Class-1 cycle incurred by $\mathbf{e}_q^l$ is given by $\Delta_r(\mathbf{e}_q^l)$. Finally, the horizontal and the vertical lines connecting the new entry to the first sub-diagonal complete the loop of the Class-1 cycle, which concludes the proof. ■

B. Class-2 Cycles

The length of a Class-2 cycle mainly depends on the number of the previous entries involved in the cycle and the differences of the row and column indices among the entries in the cycle. As the number of previously located entries in a cycle increases, the length of the cycle increases. Thus, we do not necessarily consider cycles formed by many previously located entries because a long cycle formed by many edges hardly affects the decoding performance. For instance, the length of a cycle including three previously located entries is at least 10 (typically much larger). Thus, in this paper, we consider the cases with up to two previously located entries for simplicity. Let $(\mathbf{e})_c$ and $(\mathbf{e})_r$ denote the column and the row indices of a nonzero matrix entry $\mathbf{e}$, respectively. Here, the column and row indices are defined in the same way as in Fig. 1. Also, define $\Delta_r(\mathbf{e}_1, \mathbf{e}_2)$ and $\Delta_c(\mathbf{e}_1, \mathbf{e}_2)$ as the absolute values of the differences between the row indices and between the column indices of $\mathbf{e}_1$ and $\mathbf{e}_2$, respectively. Then, the following lemmas provide the length of possible connections in a Class-2 cycle.

Lemma 1: Consider a connection starting upward from the new entry $\mathbf{e}_q^l$ and finishing downward to a previously located entry $\mathbf{e}$ through diagonal entries. Then, the length of the connection is given by $2\Delta_c(\mathbf{e}_q^l, \mathbf{e}) + 1$.

Proof: Since $\mathbf{e}_q^l$ and $\mathbf{e}$ are respectively connected to the first sub-diagonal entry and the main diagonal entry on the their columns, there are $\Delta_c(\mathbf{e}_q^l, \mathbf{e}) - 1$ and one-half zigzaggers in the connection, which concludes the proof. ■

Lemma 2: Consider a connection starting rightward from the new entry $\mathbf{e}_q^l$ and finishing downward to a previously located entry $\mathbf{e}$ through diagonal entries. Then, the length of the connection is given by $\phi\left((\mathbf{e}_q^l)_r - (\mathbf{e})_c\right)$, where $\phi(x) = 2\left|x - \frac{1}{2}\right| + 1$.

Proof: When $x = \left(\mathbf{e}_q^l\right)_r - (\mathbf{e})_c > 0$, $\mathbf{e}_q^l$ is connected to the main diagonal entry of the same row at the $\left(\left(\mathbf{e}_q^l\right)_r - 1\right)$th column. Then, there are $(x-1)$ zigzaggers in the connection, which gives the connection length as $\phi(x) = 2x$. On the other hand, when $x = \left(\mathbf{e}_q^l\right)_r - (\mathbf{e})_c \leq 0$, $\mathbf{e}_q^l$ is connected to the first sub-diagonal entry of the same row at the $\left(\mathbf{e}_q^l\right)_r$th column. Then, there are $|x|$ zigzaggers in the connection, which gives the connection length as $\phi(x) = 2|x| + 2$, which concludes the proof. ■

Lemma 3: Consider a connection starting rightward from the new entry $\mathbf{e}_q^l$ and finishing leftward to a previously located entry $\mathbf{e}$ through diagonal entries. Then, the length of the connection is given by $2\Delta_r(\mathbf{e}_q^l, \mathbf{e}) + 1$.

Proof: When $\Delta_r(\mathbf{e}_q^l, \mathbf{e}) \neq 0$, $\mathbf{e}_q^l$ and $\mathbf{e}$ are respectively connected to the main diagonal entry and the first sub-diagonal entry on their rows. Then, there are $\Delta_r(\mathbf{e}_q^l, \mathbf{e}) - 1$ and one-half zigzaggers in the connection. When $\Delta_r(\mathbf{e}_q^l, \mathbf{e}) = 0$, $\mathbf{e}_q^l$ and $\mathbf{e}$ are directly connected with connection length 1, which concludes the proof. ■

Then, the following theorem provides the length of a Class-2 cycle involved with a single previously located entry.

Theorem 2: Let $L_2^1(\mathbf{e}_q^l, \mathbf{e})$ be the length of the cycle formed by the new entry $\mathbf{e}_q^l$, a single previously located entry $\mathbf{e}$, and diagonal entries. Then, $L_2^1(\mathbf{e}_q^l, \mathbf{e})$ is given by

$$L_2^1(\mathbf{e}_q^l, \mathbf{e}) = 2(\Delta_r(\mathbf{e}_q^l, \mathbf{e}) + \Delta_c(\mathbf{e}_q^l, \mathbf{e}) + 1), \ \ \mathbf{e} \in E_p, \tag{2}$$

where E_p denotes the set of all previously located entries excluding the two diagonals.

Proof: It is straightforward that such a cycle consists of two connections given in Lemmas 1 and 3 as can be seen in Fig. 3 (the solid line formed by the new entry placed at the second row and the fifth column and one previous entry placed at the first row and the third column), which concludes the proof. ■

Now, consider the cases with two previously located entries $\mathbf{e}_1$ and $\mathbf{e}_2$. Without loss of generality, we assume $(\mathbf{e}_1)_c \leq (\mathbf{e}_2)_c$. Let $\max_r(\mathbf{e}_1, \mathbf{e}_2)$ and $\min_r(\mathbf{e}_1, \mathbf{e}_2)$ denote the maximum and the minimum of the row indices of two entries $\mathbf{e}_1$ and $\mathbf{e}_2$, respectively. When $(\mathbf{e}_1)_c < (\mathbf{e}_2)_c$, there are two cases

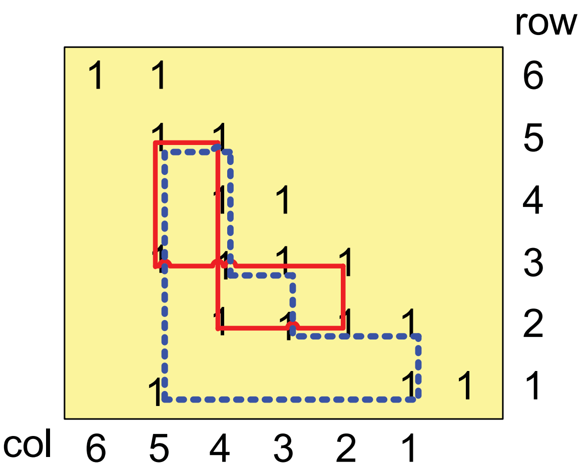


Fig. 3. Examples of Class-2 cycles.

of a Class-2 cycle with the two entries: i) when $(\mathbf{e}_2)_r > \max_r(\mathbf{e}_q^l, \mathbf{e}_1)$ and ii) when $\left(\mathbf{e}_q^l\right)_r > \max_r(\mathbf{e}_1, \mathbf{e}_2)$. In Fig. 3, an instance of the first case is illustrated with a dotted line (a cycle formed by the new entry placed at the first row and the fifth column, and two previous entries, one placed at the third row and the fourth column and the other placed at the second row and the third column). When $(\mathbf{e}_1)_c = (\mathbf{e}_2)_c$, a Class-2 cycle occurs only when $(\mathbf{e}_2)_r > \max_r(\mathbf{e}_q^l, \mathbf{e}_1)$. Then, the following theorem provides the length of a Class-2 cycle with two previous entries.

Theorem 3: Let $L_2^2(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2)$ be the length of a Class-2 cycle formed by the new entry $\mathbf{e}_q^l$ and two previously located entries $\mathbf{e}_1, \mathbf{e}_2 \in E_p$. Here, without loss of generality, we set $(\mathbf{e}_1)_c \leq (\mathbf{e}_2)_c$. When $(\mathbf{e}_1)_c = (\mathbf{e}_2)_c$, we choose $\mathbf{e}_2$ as the one with larger row index. Then, $L_2^2(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2)$ is given by (3), where 'case A2' is when $(\mathbf{e}_2)_r > \max_r(\mathbf{e}_q^l, \mathbf{e}_1)$ and $(\mathbf{e}_q^l)_r < (\mathbf{e}_1)_c$, and 'case B2' is when $\left(\mathbf{e}_q^l\right)_r > \max_r(\mathbf{e}_1, \mathbf{e}_2)$, $(\mathbf{e}_2)_r < (\mathbf{e}_1)_c$ and $(\mathbf{e}_q^l)_r < (\mathbf{e}_2)_c$.

Proof: First, consider the case where $(\mathbf{e}_1)_c < (\mathbf{e}_2)_c$. For case A2, there exists a loop comprised of the three connections: i) starting from $\mathbf{e}_q^l$ upward to the first sub-diagonal, passing through the diagonals, and finishing downward to $\mathbf{e}_2$, ii) starting from $\mathbf{e}_2$ rightward to one of the two diagonal, passing through the diagonals, and finishing downward to $\mathbf{e}_1$, and iii) starting from $\mathbf{e}_q^l$ rightward, passing through the diagonals, and finishing leftward to $\mathbf{e}_1$. Then, from Lemmas 1-3, the length of the cycle is given by $2(\Delta_r(\mathbf{e}_q^l, \mathbf{e}_1) + \Delta_c(\mathbf{e}_q^l, \mathbf{e}_2) + 1) + \phi\left((\mathbf{e}_2)_r - (\mathbf{e}_1)_c\right)$. Similarly for case B2, there exists a loop comprised of the three connections: i) starting from $\mathbf{e}_q^l$ upward to the first sub-diagonal, passing through the diagonals, and finishing downward to $\mathbf{e}_2$, ii) starting from $\mathbf{e}_q^l$ rightward to one of the two diagonal, passing through the diagonals, and finishing downward to $\mathbf{e}_1$, and iii) starting from $\mathbf{e}_2$ rightward, passing through the diagonals, and finishing leftward to $\mathbf{e}_1$. Then, from Lemmas 1-3, the length of the cycle is given by $2(\Delta_r(\mathbf{e}_2, \mathbf{e}_1) + \Delta_c(\mathbf{e}_q^l, \mathbf{e}_2) + 1) + \phi\left(\left(\mathbf{e}_q^l\right)_r - (\mathbf{e}_1)_c\right)$. When $(\mathbf{e}_1)_c = (\mathbf{e}_2)_c$ and $\left(\mathbf{e}_q^l\right)_r \leq \min_r(\mathbf{e}_1, \mathbf{e}_2)$, there exists a loop comprised of the three connections: i) starting from $\mathbf{e}_q^l$ upward to the first sub-diagonal, passing through the diagonals, and finishing leftward to $\mathbf{e}_2$, ii) starting from $\mathbf{e}_2$ directly downward to $\mathbf{e}_1$, and iii) starting from $\mathbf{e}_q^l$ rightward and finishing leftward to $\mathbf{e}_1$. In this case, the length of the first connection is given by $2(\left(\mathbf{e}_q^l\right)_c - (\mathbf{e}_2)_r + 1)$, the length of the second connection is 1, and the length of the third connection is given by $2\Delta_r(\mathbf{e}_q^l, \mathbf{e}_1) + 1$. With a slight manipulation, it is easily seen that the length of the cycle is also given by $2(\Delta_r(\mathbf{e}_q^l, \mathbf{e}_1) + \Delta_c(\mathbf{e}_q^l, \mathbf{e}_2) + 1) + \phi\left((\mathbf{e}_2)_r - (\mathbf{e}_1)_c\right)$. In other cases, a Class-2 cycle including $\mathbf{e}_1$ and $\mathbf{e}_2$ does not exist since at least one zigzagger is included twice in the loop $\mathbf{e}_q^l \rightarrow \mathbf{e}_2 \rightarrow \mathbf{e}_1 \rightarrow \mathbf{e}_q^l$, which concludes the proof. ■

C. The Class-3 Cycles

A Class-3 cycle can be viewed as an extension of the first two classes. For the same reason as for a Class-2 cycle, we consider the cases with up to two previously located entries. Then, there are three different cases among the Class-3 cycles: i) cycles formed by two new entries and diagonal entries, ii) cycles formed by two new entries, one previously located entry, and diagonal entries, and iii) cycles formed by two new entries, two previously located entries, and diagonal entries. Henceforth, we assume without loss of generality that, for the two new entries $\mathbf{e}_q^l$ and $\mathbf{e}_{q'}^l$, $(\mathbf{e}_q^l)_r < (\mathbf{e}_{q'}^l)_r$.

First, consider the case of a Class-3 cycle formed only by two new entries in the current column and the diagonal entries. Fig. 4(a) shows an instance of this case (the solid-line loop formed by the two new entries placed at the second row and the fifth column and at the fourth row and the fifth column, respectively). Since this case can be viewed as an extension of the Class-1 cycle, the length of the cycle is calculated similarly to the Class-1 cycle case as follows.

Theorem 4: Let $L_3^0(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)$ be the length of the Class-3 cycle formed by the new entries $\mathbf{e}_q^l$ and $\mathbf{e}_{q'}^l$ added in the same column. Then, $L_3^0(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)$ is given by

$$L_3^0(\mathbf{e}_q^l, \mathbf{e}_{q'}^l) = 2(\Delta_r(\mathbf{e}_q^l, \mathbf{e}_{q'}^l) + 1), \ \{\mathbf{e}_q^l, \mathbf{e}_{q'}^l\} \subset E_c, \quad (4)$$

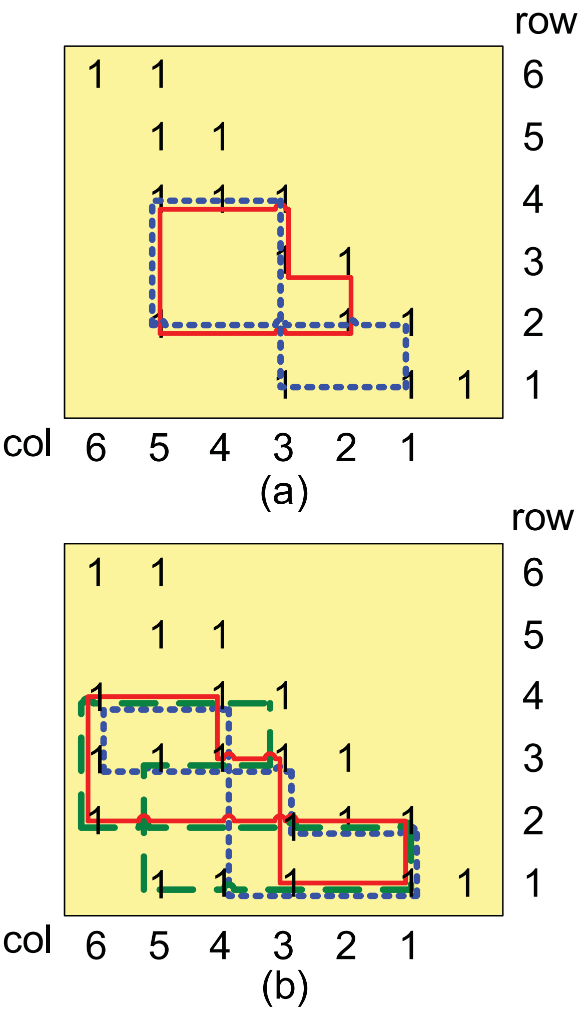
where E_c denotes the set of all entries to be located in the current column.

Proof: The theorem can be proved in the same way as for the Class-1 cycle except one disparity that the number of zigzaggers in the loop of this Class-3 cycle is equal to $\Delta_r(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)$. Therefore, we complete the proof by replacing $\Delta_r(\mathbf{e}_q^l)$ in (1) with $\Delta_r(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)$. ■

Now, consider the case of Class-3 cycles formed by two new entries, one previously located entry, and diagonal entries, whose example is depicted with the dotted-line loop in Fig. 4(a) (formed by the two new entries placed at the second row and the fifth column and at the fourth row and the fifth column, respectively, and one previous entry placed at the first row and the third column). Note that this kind of Class-3 cycle occurs only if the previously located entry, $\mathbf{e}$, satisfies $(\mathbf{e})_r < (\mathbf{e}_{q'}^l)_r$ and $(\mathbf{e})_c > (\mathbf{e}_q^l)_r$.

Theorem 5: Let $L_3^1(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e})$ be the length of the Class-3 cycle formed by the two new entries $\mathbf{e}_q^l, \mathbf{e}_{q'}^l$ and a single previously located entry $\mathbf{e}$. Then, $L_3^1(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e})$ is given by (5).

Proof: Note that (3) still holds when $(\mathbf{e}_2)_c = (\mathbf{e}_q^l)_c (= (\mathbf{e}_{q'}^l)_c)$. Set $\mathbf{e}_2 = \mathbf{e}_{q'}^l$ and $\mathbf{e}_1 = \mathbf{e}$. Then, we see that $(\mathbf{e}_2)_r > \max_r(\mathbf{e}_q^l, \mathbf{e}_1)$, which corresponds to the first case of



(a)

(b)

Fig. 4. Examples of Class-3 cycles.

(3). Thus, the proof is completed by replacing $\mathbf{e}_1$ and $\mathbf{e}_2$ of the first expression in (3) with $\mathbf{e}$ and $\mathbf{e}^l_{q'}$, respectively, because $\Delta_c(\mathbf{e}^l_q, \mathbf{e}_2) = 0$. ■

Next, consider the cases with two previously located entries $\mathbf{e}_1$ and $\mathbf{e}_2$. Let us define $\mu = \min_r(\mathbf{e}_1, \mathbf{e}_2)$ and $\xi = \max_r(\mathbf{e}_1, \mathbf{e}_2)$. Also, let $\mathbf{e}_{\max}$ and $\mathbf{e}_{\min}$ respectively denote the one with the larger row index and the one with smaller row index between $\mathbf{e}_1$ and $\mathbf{e}_2$. Then, the following theorem provides the length of this kind of cycle.

Theorem 6: Let $L_3^2(\mathbf{e}^l_q, \mathbf{e}^l_{q'}, \mathbf{e}_1, \mathbf{e}_2)$ be the length of a Class-3 cycle formed by the two new entries $\mathbf{e}^l_q, \mathbf{e}^l_{q'}$ and two previously located entries $\mathbf{e}_1, \mathbf{e}_2 \in E_p$. Here, without loss of generality, we set $(\mathbf{e}_1)_c \le (\mathbf{e}_2)_c$. When $(\mathbf{e}_1)_c = (\mathbf{e}_2)_c$, we choose $\mathbf{e}_2$ as the one with larger row index. Then, $L_3^2(\mathbf{e}^l_q, \mathbf{e}^l_{q'}, \mathbf{e}_1, \mathbf{e}_2)$ is given by (6), where 'case A3' is when $(\mathbf{e}^l_q)_r < \xi < (\mathbf{e}_1)_c$ and $\mu < (\mathbf{e}^l_{q'})_r < (\mathbf{e}_1)_c$, 'case B3' is when $(\mathbf{e}^l_{q'})_r > \max((\mathbf{e}_2)_r, (\mathbf{e}_1)_c)$, $(\mathbf{e}^l_q)_r < \min((\mathbf{e}_2)_r, (\mathbf{e}_1)_c)$, and $(\mathbf{e}_2)_r > (\mathbf{e}_1)_c$, and 'case C3' is when $(\mathbf{e}_2)_c > (\mathbf{e}^l_q)_r > \xi$ and $(\mathbf{e}^l_{q'})_r > (\mathbf{e}_1)_c > \xi$. Fig. 4(b) illustrates an instance of case A3 (the solid-line loop formed by the two new entries placed at the second row and the sixth column and at the fourth row and the sixth column, respectively, and the two previous entries, one placed at the third row and the fourth column and the other placed at the first row and third column), an instance of case B3 (the dotted-line loop formed by the two new entries placed at the third row and the sixth column and at the fourth row and the sixth column, respectively, and two previous entries, one placed at the first row and the fourth column and the other placed at the second row and the third column) and an instance of case C3 (the dashed-line loop formed by the two new entries placed at the second row and the sixth column and at the fourth row and the sixth column, respectively, and two previous entries placed at the first row and the fifth column and at the third row and the fifth column, respectively), respectively.

Proof: The loop consists of the following four connections: i) $\mathbf{e}^l_q \to \mathbf{e}^l_{q'}$, ii) $\mathbf{e}^l_{q'} \to$ one previous entry, iii) between the two previous entries, iv) the other previous entry $\to \mathbf{e}^l_q$. Since i) is a vertical connection, ii) (or iii)) should be a connection given by either Lemma 2 or Lemma 3. Then, the possible combinations for the connections ii), iii), and iv) are as follows: case A3) Lemma 3 → Lemma 1 → Lemma 3, case B3) Lemma 2 → Lemma 2 → Lemma 3, and case C3) Lemma 2 → Lemma 3 → Lemma 2.

First, consider case A3. In this case, ii), iii), and iv) are $\mathbf{e}^l_{q'} \to \mathbf{e}_{\max}$, $\mathbf{e}_{\max} \to \mathbf{e}_{\min}$, and $\mathbf{e}_{\min} \to \mathbf{e}^l_q$, respectively. Since the connection i) is of length 1, the length of the cycle is easily given by $2(\Delta_c(\mathbf{e}_1, \mathbf{e}_2) + \Delta_r(\mathbf{e}^l_q, \mathbf{e}_{\min}) + \Delta_r(\mathbf{e}^l_{q'}, \mathbf{e}_{\max}) + 2)$ from Lemmas 1 and 3. The connections ii), iii), and iv), respectively, contain zigzaggers in the columns between $(\mathbf{e}^l_{q'})_r$ and ξ, those between $(\mathbf{e}_1)_c$ and $(\mathbf{e}_2)_c$, and those between $(\mathbf{e}^l_q)_r$ and μ. In order for such a cycle to exist, there should be no zigzagger contained in more than one path. Since $(\mathbf{e}_2)_c \ge (\mathbf{e}_1)_c$, $(\mathbf{e}_2)_c > \xi$, and $(\mathbf{e}_1)_c > \mu$, such a cycle exists if and only if $(\mathbf{e}^l_q)_r < \xi < (\mathbf{e}_1)_c$ and $\mu < (\mathbf{e}^l_{q'})_r < (\mathbf{e}_1)_c$. Next, consider case B3. In this case, ii), iii), and iv) are $\mathbf{e}^l_{q'} \to \mathbf{e}_2$, $\mathbf{e}_2 \to \mathbf{e}_1$, and $\mathbf{e}_1 \to \mathbf{e}^l_q$, respectively. Thus, the length of the cycle is given by $2(\Delta_r(\mathbf{e}^l_q, \mathbf{e}_1) + 1) + \phi(\Delta_r(\mathbf{e}_2) - \Delta_c(\mathbf{e}_1)) + \phi(\Delta_r(\mathbf{e}^l_{q'}) - \Delta_c(\mathbf{e}_2))$ from Lemmas 2 and 3. The

$$L_2^2(\mathbf{e}^l_q, \mathbf{e}_1, \mathbf{e}_2) = \begin{cases} 2(\Delta_r(\mathbf{e}^l_q, \mathbf{e}_1) + \Delta_c(\mathbf{e}^l_q, \mathbf{e}_2) + 1) + \phi\left((\mathbf{e}_2)_r - (\mathbf{e}_1)_c\right) & \text{if case A2,} \\ 2(\Delta_r(\mathbf{e}_1, \mathbf{e}_2) + \Delta_c(\mathbf{e}^l_q, \mathbf{e}_2) + 1) + \phi\left((\mathbf{e}^l_q)_r - (\mathbf{e}_1)_c\right) & \text{if case B2,} \\ \infty & \text{otherwise.} \end{cases} \tag{3}$$

$$L_3^1(\mathbf{e}^l_q, \mathbf{e}^l_{q'}, \mathbf{e}) = \begin{cases} 2(\Delta_r(\mathbf{e}^l_q, \mathbf{e}) + 1) + \phi\left((\mathbf{e}^l_{q'})_r - (\mathbf{e})_c\right) & \text{if } (\mathbf{e})_r < (\mathbf{e}^l_{q'})_r \text{ and } (\mathbf{e})_c > (\mathbf{e}^l_q)_r, \\ \infty & \text{otherwise.} \end{cases} \tag{5}$$

connections ii), iii), and iv) respectively contain zigzaggers in the columns between $(\mathbf{e}^l_{q'})_r$ and $(\mathbf{e}_2)_c$, those between $(\mathbf{e}_2)_r$ and $(\mathbf{e}_1)_c$, and those between $(\mathbf{e}_1)_r$ and $(\mathbf{e}^l_q)_r$. Since $(\mathbf{e}_2)_c > (\mathbf{e}_2)_r$ and $(\mathbf{e}_1)_c > (\mathbf{e}_1)_r$, such a cycle exists if and only if $(\mathbf{e}^l_{q'})_r > \max((\mathbf{e}_2)_r, (\mathbf{e}_1)_c)$, $(\mathbf{e}^l_q)_r < \min((\mathbf{e}_2)_r, (\mathbf{e}_1)_c)$, and $(\mathbf{e}_2)_r > (\mathbf{e}_1)_c$. Finally, consider case C3. In this case, ii), iii), and iv) are $\mathbf{e}^l_{q'} \to \mathbf{e}_2$, $\mathbf{e}_2 \to \mathbf{e}_1$, and $\mathbf{e}_1 \to \mathbf{e}^l_q$, respectively. Thus, the length of the cycle is given by $2(\Delta_r(\mathbf{e}_1, \mathbf{e}_2) + 1) + \phi(\Delta_r(\mathbf{e}^l_q) - \Delta_c(\mathbf{e}_1)) + \phi(\Delta_r(\mathbf{e}^l_{q'}) - \Delta_c(\mathbf{e}_2))$ from Lemmas 2 and 3. Then, the connections ii), iii), and iv) respectively contain zigzaggers in the columns between $(\mathbf{e}^l_{q'})_r$ and $(\mathbf{e}_2)_c$, those between μ and ξ, and those between $(\mathbf{e}_1)_c$ and $(\mathbf{e}^l_q)_r$. Thus, it is easily seen that such a cycle exists if and only if $(\mathbf{e}_2)_c > (\mathbf{e}^l_q)_r > \xi$ and $(\mathbf{e}^l_{q'})_r > (\mathbf{e}_1)_c > \xi$, which concludes the proof. ■

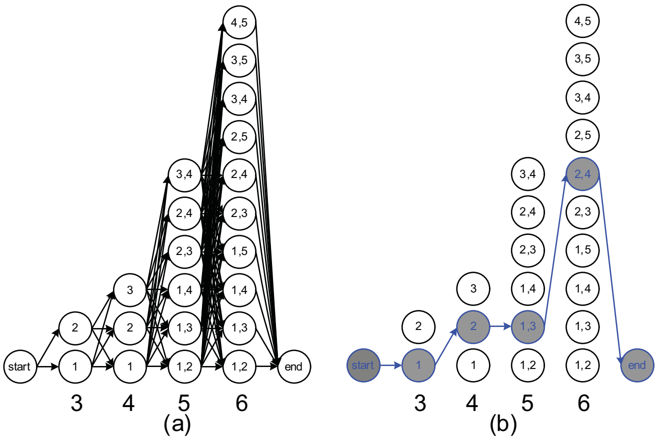


Fig. 5. A trellis search example.

D. The Measure Function

Now, let us define the measure function for a possible edge configuration E_c.

Definition 4: Let $\mathcal{M}(E_c)$ be the measure function for the cycle distribution over a given edge configuration E_c. Then, $\mathcal{M}(E_c)$ is given by

$$\mathcal{M}(E_c) = \sum_{\mathbf{e}^l_q \in E_c} \left(\lambda_1(\mathbf{e}^l_q) + \lambda_2(\mathbf{e}^l_q)\right) + \sum_{\{\mathbf{e}^l_q, \mathbf{e}^l_{q'}\} \subset E_c} \lambda_3(\mathbf{e}^l_q, \mathbf{e}^l_{q'}), \tag{7}$$

where $\lambda_1(\mathbf{e}^l_q)$ is the measure for Class-1 cycles defined as

$$\lambda_1(\mathbf{e}^l_q) = D^{-\frac{1}{2}L_1(\mathbf{e}^l_q)} \tag{8}$$

and D is a positive real design parameter discussed later in this subsection. Also, $\lambda_2(\mathbf{e}^l_q)$ is the measure for Class-2 cycles defined as

$$\lambda_2(\mathbf{e}^l_q) = \sum_{\mathbf{e} \in E_p} D^{-\frac{1}{2}L^1_2(\mathbf{e}^l_q, \mathbf{e})} + \sum_{(\mathbf{e}_1, \mathbf{e}_2) \in \hat{E}^2_p} D^{-\frac{1}{2}L^2_2(\mathbf{e}^l_q, \mathbf{e}_1, \mathbf{e}_2)}, \tag{9}$$

where $\hat{E}^2_p$ is the set of all two-tuples $(\mathbf{e}_1, \mathbf{e}_2)$ satisfying i) $\mathbf{e}_1, \mathbf{e}_2 \in E_p$ and ii) $(\mathbf{e}_2)_c \geq (\mathbf{e}_1)_c$ (if $(\mathbf{e}_2)_c = (\mathbf{e}_1)_c$, $(\mathbf{e}_2)_r > (\mathbf{e}_1)_r$). Finally, $\lambda_3(\mathbf{e}^l_q, \mathbf{e}^l_{q'})$ is the measure for Class-3 cycles defined as (10).

Note that such an exponential-type measure function was already used in [10] and can be justified by the fact that a small number of short cycles are more likely to degrade the decoding performance than a large number of large cycles. Note that a single cycle of length $2L$ is assumed to have as much impact as D cycles of length $2(L+1)$ in the proposed measure function. Here, D is a design parameter to be set to an appropriate positive real number to quantify the relations between adjacent lengths of cycles. However, those relations have not yet been quantitatively studied. If D is chosen too large, the impact caused by larger cycles is not distinguished. If it is chosen too small, short cycles may not be prevented in the resulting graph. For design examples given in Section IV, D is set to 10.

E. The Trellis Search

Let us consider the construction of a trellis for the optimization. Fig. 5(a) depicts the trellis for the echelon form (the parity-check part) shown in Fig. 1. Here, the *start* and *end* nodes indicate the origin and the termination of the trellis, respectively. In each column of the trellis, each possible configuration of the additional entries is denoted as a single node. The digits labeled below the columns indicate the column indices of the corresponding parity-check nodes as defined in Fig. 1, and the digits labeled in nodes represent the row indices of new entries in the column. For the lth column with degree of d_l and M_l available positions in the echelon form, $\binom{M_l}{d_l-2}$ configurations are possible. In many practical applications [1][7][11], the maximum degree of bit nodes in the parity-check part is chosen to be no more than four. Thus, especially for low-rate codes commonly designed to have relatively short lengths, each column with $\binom{M_l}{d_l-2}$ nodes in the trellis can be handled with manageable complexity. Therefore, the proposed method fits well into the case of structured codes, where the values of M_l are usually small. The extension to structured codes will be addressed in the next section.

The edges in the trellis, emanating from a node of one column and arriving at some nodes of the next column, denote possible configurations for entries of the next column after the entry addition of the current column. Although a node of the current column can be connected to all nodes of the next column so that all possible configurations of entry addition are covered, some connections will be obviated by the 'pruning' procedure which removes the edges corresponding to the cases

$$L^2_3(\mathbf{e}^l_q, \mathbf{e}^l_{q'}, \mathbf{e}_1, \mathbf{e}_2) = \begin{cases} 2(\Delta_c(\mathbf{e}_1, \mathbf{e}_2) + \Delta_r(\mathbf{e}^l_q, \mathbf{e}_{\min}) + \Delta_r(\mathbf{e}^l_{q'}, \mathbf{e}_{\max}) + 2) & \text{if case A3,} \\ 2(\Delta_r(\mathbf{e}^l_q, \mathbf{e}_1) + 1) + \phi(\Delta_r(\mathbf{e}_2) - \Delta_c(\mathbf{e}_1)) + \phi(\Delta_r(\mathbf{e}^l_{q'}) - \Delta_c(\mathbf{e}_2)) & \text{if case B3,} \\ 2(\Delta_r(\mathbf{e}_1, \mathbf{e}_2) + 1) + \phi(\Delta_r(\mathbf{e}^l_q) - \Delta_c(\mathbf{e}_1)) + \phi(\Delta_r(\mathbf{e}^l_{q'}) - \Delta_c(\mathbf{e}_2)) & \text{if case C3,} \\ \infty & \text{otherwise.} \end{cases} \tag{6}$$

not allowed by the aforementioned structural constraint. Except this pruning procedure, the calculation for entry addition proceeds in the similar way of dynamic programming [12]. Also, each node in the previous column has the corresponding survived path associated with all selected nodes from the origin all the way to that column. If any node labeled with at least two digits in the survived path has a common row index with the entry at the node in the current column, the corresponding edge in the trellis is discarded before the selection of the best measure. This operation inhibits any two columns with degree more than two from having entries with common row indices, which avoids short cycles formed only by added entries. After the pruning procedure, the edge with the best measure up to the current column is selected among arriving edges at each node while the remaining edges are discarded. Finally, the path associated with the best measure is chosen among all possible paths to complete the addition of new entries. Fig. 5(b) shows the result of the proposed algorithm with the trellis associated with the echelon form in Fig. 1. The resultant entry configuration is represented by the indices of the rows and columns as

$$\{(\mathbf{r},\mathbf{c})\} = \{(1,3),(2,4),(1,5),(3,5),(2,6),(4,6)\}. \quad (11)$$

The remaining part of the parity-check matrix, H_1, can then be constructed from the initial graph associated with H_{tr} in a controlled random manner, as in [8], which requires additional cycle conditioning scheme. However, since most part of the parity-check matrix has been already designed, the design of H_1 becomes simple in most cases. Although only cycle distribution is considered in this paper for the performance measure, other measures, such as in [13], can be easily adopted in the proposed algorithm if a careful management of error floor is required. Furthermore, from the property of parity-check matrix with respect to the minimum distance [14], we see that any set of columns in the echelon form designed by the proposed algorithm does not form the set of columns which determine the minimum distance.

III. Extension to the Structured Code Case

In practical applications, a structured LDPC code is preferred since it has much lower memory requirement and allows more parallelism for the implementation of encoding and decoding [1][7][11]. In this section, the proposed algorithm is extended for the design of a block-type structured LDPC code, such as in [1][7][11][15][16][17]. The parity-check matrix H of an $(n,k) = (n_bB, k_bB)$ structured code can be represented by its constituent submatrices as $H = [\gamma_{i,j}]$ [1][15]. Here, $\gamma_{i,j}$, $0 \le i \le n_b - k_b - 1$ and $0 \le j \le n_b - 1$, denotes the $(i,j)th$ constituent submatrix that is either a $B \times B$ zero matrix or a μ-time ($0 \le \mu \le B-1$) right-cyclic-shifted matrix σ_μ of the $B \times B$ identity matrix. Also, let $C_H = [\nu_{i,j}]$ be the $(n_b - k_b) \times n_b$ characteristic matrix of H, where $\nu_{i,j} = 0$ if $\gamma_{i,j}$ is a zero matrix and $\nu_{i,j} = \mu+1$ if $\gamma_{i,j} = \sigma_\mu$.[2] Fig. 6 illustrates an example of the characteristic matrix for a structured code. Here, C_H is partitioned into $C_H = [C_{H,1}\ C_{H,tr}]$, where $C_{H,1}$ and $C_{H,tr}$ are associated with the message part and the parity-check part, respectively. Hence, $C_{H,tr}$ should be an echelon form, and $\mathbf{a}_i$ and $\mathbf{b}_i$ denote the nonzero values in the two diagonals in $C_{H,tr}$, which can be arbitrarily selected among $\{1, \cdots, B\}$. From the result in [15], if a cycle of length L is incurred by nonzero entries in $C_{H,tr}$, there exist corresponding $\gcd(S,B)$ cycles in H_{tr} of length given by

$$L^{actual} = \frac{LB}{\gcd(S,B)}, \quad (12)$$

where S is the alternating sum of the shift indices in the L nonzero entries forming the cycle in $C_{H,tr}$ (addition for even ordered entries and subtraction for odd ordered entries).

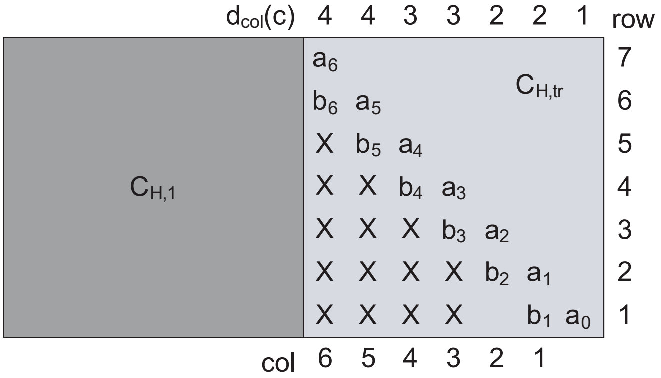


Fig. 6. The representation of the echelon form for a structured code.

A. The Calculation of Alternating Sums

If the length of a cycle in $C_{H,tr}$ is determined using (1)-(10), the lengths of the corresponding cycles in H_{tr} are directly evaluated using (12). Also, due to the structural property of the echelon form, the alternating sum S for each class is easily evaluated. Let $\nu(\mathbf{e})$ be the value of the entry in C_H corresponding to the entry $\mathbf{e}$ and $\Delta_S(\mathbf{e}_1, \mathbf{e}_2) = \nu(\mathbf{e}_1) - \nu(\mathbf{e}_2)$. For a compact representation, define $\min_{cr}(\mathbf{e}_1,\mathbf{e}_2) = \min((\mathbf{e}_1)_c, (\mathbf{e}_2)_r)$, $\max_{cr}(\mathbf{e}_1,\mathbf{e}_2) = \max((\mathbf{e}_1)_c, (\mathbf{e}_2)_r)$, $\mathrm{sgn}_r(\mathbf{e}_1,\mathbf{e}_2) = \mathrm{sgn}((\mathbf{e}_1)_\mathrm{r} - (\mathbf{e}_2)_\mathrm{r})$ and $\mathrm{sgn}_{cr}(\mathbf{e}_1,\mathbf{e}_2) = \mathrm{sgn}((\mathbf{e}_1)_\mathrm{c} - (\mathbf{e}_2)_\mathrm{r})$. Here, the function $\mathrm{sgn}(\cdot)$ is a signum function whose value is -1 when the argument is negative, 0 when zero, and 1 otherwise. As can be seen in (12), the length of an actual cycle does not depend on the sign of the alternating sum, so that the starting point can be arbitrarily chosen. For simplicity, we set the starting point at $\mathbf{e}_q^l$. Since a single zigzagger is associated with one $\mathbf{a}_i$ followed by one $\mathbf{b}_i$, the alternating sum of the zigzaggers contained in

$$\lambda_3(\mathbf{e}_q^l, \mathbf{e}_{q'}^l) = D^{-\frac{1}{2}L_3^0(\mathbf{e}_q^l,\mathbf{e}_{q'}^l)} + \sum_{\mathbf{e}\in E_p} D^{-\frac{1}{2}L_3^1(\mathbf{e}_q^l,\mathbf{e}_{q'}^l,\mathbf{e})} + \sum_{(\mathbf{e}_1,\mathbf{e}_2)\in \hat{E}_p^2} D^{-\frac{1}{2}L_3^2(\mathbf{e}_q^l,\mathbf{e}_{q'}^l,\mathbf{e}_1,\mathbf{e}_2)}. \quad (10)$$

[2] The model matrix defined in [1] is another way to represent a structured code and can be obtained from the characteristic matrix by subtracting 1 from each element of the characteristic matrix.

the loop is given as $\sum_i(\mathbf{a}_i - \mathbf{b}_i)$. Then, we can easily obtain the alternating sum of each class cycle as follows.

Theorem 7: Let $S_1(\mathbf{e}_q^l)$ be the alternating sum for a Class-1 cycle associated with $L_1(\mathbf{e}_q^l)$. Then, $S_1(\mathbf{e}_q^l)$ is given by

$$S_1(\mathbf{e}_q^l) = \nu(\mathbf{e}_q^l) - \mathbf{b}_l + \sum_{i=(\mathbf{e}_q^l)_r}^{l-1} (\mathbf{a}_i - \mathbf{b}_i). \tag{13}$$

Proof: The alternating sum can be evaluated by keeping track of entries in the loop associated with $L_1(\mathbf{e}_q^l)$. One can start from $\mathbf{e}_q^l$ by adding $\nu(\mathbf{e}_q^l)$ and goes upward to the first sub-diagonal entry by subtracting its index ($\mathbf{b}_l$). Then, the calculation of the remaining terms along the diagonals completes (13), which concludes the proof. ■

Lemma 4: The alternating sums for the connections given in Lemmas 1-3, $\sigma_i(\mathbf{e}_q^l, \mathbf{e})$, $i = 1, 2, 3$, are respectively given by (14)-(16).

Proof: The alternating sums of the connections given in the lemma can be straightforwardly obtained from Lemmas 1-3, their proofs, and Fig. 6. ■

Theorem 8: Let $S_2^1(\mathbf{e}_q^l, \mathbf{e})$ and $S_2^2(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2)$ be the alternating sum for the Class-2 cycles associated with $L_2^1(\mathbf{e}_q^l, \mathbf{e})$ and $L_2^2(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2)$, respectively. Then, $S_2^1(\mathbf{e}_q^l, \mathbf{e})$ and $S_2^2(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2)$ are respectively given by

$$S_2^1(\mathbf{e}_q^l, \mathbf{e}) = \sigma_1(\mathbf{e}_q^l, \mathbf{e}) - \sigma_3(\mathbf{e}, \mathbf{e}_q^l) - \Delta_{\mathcal{S}}(\mathbf{e}_q^l, \mathbf{e}), \tag{17}$$

and (18).

Proof: Note that from Lemmas 1-3, the connection given by Lemma 1 or Lemma 3 has odd number of edges (even number of nodes) while the connection given by Lemma 2 has even number of edges (odd number of nodes). Thus, we need to change the signs of all alternating sums of connections after a Lemma 1 or a Lemma 3 connection to calculate the alternating sum of a loop. It is easily seen from the proof of Theorem 2 that $L_2^1(\mathbf{e}_q^l, \mathbf{e})$ consists of two connections given by Lemma 1 and Lemma 3. Thus, (17) is obtained from Lemma 4 by considering that $\nu(\mathbf{e}_q^l)$ is added twice and $\nu(\mathbf{e})$ is subtracted twice. Similarly, $L_2^2(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2)$ consists of three connections each given by Lemma 1→Lemma 2→Lemma 3 when case A2 (Lemma 1→Lemma 3→Lemma 2 when case B2). Thus, (18) is similarly obtained from Lemma 4 by considering doubly counted terms. ■

Theorem 9: Let $S_3^0(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)$, $S_3^1(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e})$ and $S_3^2(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}_1, \mathbf{e}_2)$ be the alternating sum for the Class-3 cycles associated with $L_3^0(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)$, $L_3^1(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e})$ and $L_3^2(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}_1, \mathbf{e}_2)$, respectively. Then, those are given by

$$S_3^0(\mathbf{e}_q^l, \mathbf{e}_{q'}^l) = \sigma_3(\mathbf{e}_{q'}^l, \mathbf{e}_q^l), \tag{19}$$

$$S_3^1(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}) = \sigma_2(\mathbf{e}_{q'}^l, \mathbf{e}) + \sigma_3(\mathbf{e}, \mathbf{e}_q^l,) - \nu(\mathbf{e}), \tag{20}$$

and (21).

Proof: Note that we can set $\mathbf{e}_{q'}^l$ as the starting point of the alternating sum. $L_3^0(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)$ consists of the Lemma 3 connection and the direct connection between $\mathbf{e}_q^l$ and $\mathbf{e}_{q'}^l$, which gives (19) from Lemma 4. Also, $L_3^1(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e})$ consists of the Lemma 2 connection between $\mathbf{e}_{q'}^l$ and $\mathbf{e}$, the Lemma 3 connection between $\mathbf{e}$ and $\mathbf{e}_q^l$, and the direct connection between $\mathbf{e}_q^l$ and $\mathbf{e}_{q'}^l$, which gives (20) from Lemma 4. Finally, (21) can be similarly obtained from the proof of Theorem 6 and Lemma 4. ■

To expedite a code design, we can set values of all diagonal entries in C_H corresponding to the echelon form to all ones, i.e. identity submatrices, which makes the alternating sums depend only on the values of additionally located entries. Then, the expressions of the alternating sums are rendered into simple expressions.

Corollary 1: If values of all diagonal entries in C_H corresponding to the echelon form are all ones (i.e. $\mathbf{a}_i = \mathbf{b}_i = 1$), then $\sigma_1(\mathbf{e}_q^l, \mathbf{e}) = \sigma_3(\mathbf{e}_q^l, \mathbf{e}) = \Delta_{\mathcal{S}}(\mathbf{e}_q^l, \mathbf{e})$ and $\sigma_2(\mathbf{e}_q^l, \mathbf{e}) = \nu(\mathbf{e}_q^l) + \nu(\mathbf{e}) - 1$.

Proof: They are straightforwardly obtained from Lemma 4 by substituting $\mathbf{a}_i$ and $\mathbf{b}_i$ with 1. ■

B. The Measure Function

Definition 5: Let $\mathcal{M}_{\mathcal{S}}(E_c)$ be the measure function for the cycle distribution over a given edge configuration E_c of a structured code. Then, $\mathcal{M}_{\mathcal{S}}(E_c)$ is given by

$$\mathcal{M}_{\mathcal{S}}(E_c) = \sum_{\mathbf{e}_q^l \in E_c} \left(\Lambda_1(\mathbf{e}_q^l) + \Lambda_2(\mathbf{e}_q^l)\right) + \sum_{\{\mathbf{e}_q^l, \mathbf{e}_{q'}^l\} \subset E_c} \Lambda_3(\mathbf{e}_q^l, \mathbf{e}_{q'}^l), \tag{22}$$

where $\Lambda_1(\mathbf{e}_q^l)$ is a measure for the Class-1 cycle defined by

$$\Lambda_1(\mathbf{e}_q^l) = N_1(\mathbf{e}_q^l) D^{-\frac{1}{2}L_1^{actual}(\mathbf{e}_q^l)}. \tag{23}$$

Here, $N_1(\mathbf{e}_q^l) = \gcd(S_1(\mathbf{e}_q^l), B)$ corresponds to the number of cycles actually incurred in the parity-check matrix and $L_1^{actual}(\mathbf{e}_q^l) = L_1(\mathbf{e}_q^l)B/N_1(\mathbf{e}_q^l)$. Also, $\Lambda_2(\mathbf{e}_q^l)$ is the measure for Class-2 cycles defined as (24), where

$$\begin{aligned} N_2^1(\mathbf{e}_q^l, \mathbf{e}) &= \gcd(S_2^1(\mathbf{e}_q^l, \mathbf{e}), B), \\ L_2^{1,actual}(\mathbf{e}_q^l, \mathbf{e}) &= \frac{L_2^1(\mathbf{e}_q^l, \mathbf{e})B}{N_2^1(\mathbf{e}_q^l, \mathbf{e})}, \\ N_2^2(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2) &= \gcd(S_2^2(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2), B), \end{aligned}$$

and

$$L_2^{2,actual}(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2) = \frac{L_2^2(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2)B}{N_2^2(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2)}.$$

Finally, $\Lambda_3(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)$ is the measure for Class-3 cycles defined as (25), where

$$\begin{aligned} N_3^0(\mathbf{e}_q^l, \mathbf{e}_{q'}^l) &= \gcd(S_3^0(\mathbf{e}_q^l, \mathbf{e}_{q'}^l), B), \\ L_3^{0,actual}(\mathbf{e}_q^l, \mathbf{e}_{q'}^l) &= \frac{L_3^0(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)B}{N_3^0(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)}, \\ N_3^1(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}) &= \gcd(S_3^1(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}), B), \\ L_3^{1,actual}(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}) &= \frac{L_3^1(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e})B}{N_3^1(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e})}, \\ N_3^2(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}_1, \mathbf{e}_2) &= \gcd(S_3^2(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}_1, \mathbf{e}_2), B), \end{aligned}$$

and

$$L_3^{2,actual}(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}_1, \mathbf{e}_2) = \frac{L_3^2(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}_1, \mathbf{e}_2)B}{N_3^2(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}_1, \mathbf{e}_2)}.$$

The trellis search with the new measure function $\mathcal{M}_{\mathcal{S}}(\cdot)$ is the same as for the non-structured code case except the structure of the trellis. In the structured code case, we need to determine both the position and the shift value of each entry in the characteristic matrix. These can be done either separately or jointly. In the separated method, the position of an entry is first decided using the measure in (7) and then the selection of the shift value achieving the best measure of (22) is followed, which does not increase the size of the trellis. However, in the joint method, the positions and the shift values are jointly optimized to provide better performance. In this case, each node of the original trellis should be split into a larger number of nodes corresponding to all possible configurations for the shift values of newly added entries. Although the number of nodes in a column of the new trellis increases remarkably as the size of sub-matrix B grows, the length of the trellis (the number of columns in the trellis) is reduced by a factor of B. As will be seen in Table I, the overall computational complexity is reduced as B increases for a given codeword size even in the joint method.

IV. Complexity Analysis and Design Example

We now analyze the computational complexity of the proposed algorithm. We first examine the complexity for the non-structured code case. Let d_l and M_l denote the degree and the number of possible positions for new entry placement of the lth column, respectively. Consider first the number of calculations at a node of the trellis. We see that the calculation for the Class-3 cycle takes most of the operations in a node processing. The calculation for the Class-3 cycle involves $\binom{d_l-2}{2}$ times the evaluation of $\Lambda_3(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)$, and each $\Lambda_3(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)$ involves $\binom{\sum_{l'=1}^{l}(d_{l'}-2)}{2}$ times of the evaluation of $L_3^2(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}_1, \mathbf{e}_2)$. Therefore, the number of operations for a node is approximately $\binom{d_l-2}{2}\binom{\sum_{l'=1}^{l}(d_{l'}-2)}{2}$. Since the number of node processing in the lth column in the trellis is $\binom{M_l}{d_l-2}$, the overall number of operations, N, is given by (26). Here, l_1 is the index of the column where the trellis for the optimization begins, and $m = n - k$ is the number of rows in H_{tr}, which is also the index of the last column where the trellis is terminated. Without loss of generality, we can assume $d_i \geq d_j$ for $i > j$. Then, we can obtain (27), where d_{max} is the maximum degree of the columns involved in the trellis search and $r = k/n$ is the code rate. Here, the first approximation comes from the fact that the last term dominates the summation. Also, the second approximation

$$\sigma_1(\mathbf{e}_q^l, \mathbf{e}) = \Delta_{\mathcal{S}}(\mathbf{e}_q^l, \mathbf{e}) + (\mathbf{a}_{(\mathbf{e})_c} - \mathbf{b}_{(\mathbf{e}_q^l)_c}) + \sum_{i=(\mathbf{e})_c+1}^{(\mathbf{e}_q^l)_c-1} (\mathbf{a}_i - \mathbf{b}_i). \tag{14}$$

$$\sigma_2(\mathbf{e}_q^l, \mathbf{e}) = \nu(\mathbf{e}_q^l) + \operatorname{sgn}_{cr}(\mathbf{e}, \mathbf{e}_q^l) \sum_{i=\min_{cr}(\mathbf{e},\mathbf{e}_q^l)}^{\max_{cr}(\mathbf{e},\mathbf{e}_q^l)-1} (\mathbf{a}_i - \mathbf{b}_i) + \nu(\mathbf{e}) - \mathbf{b}_{(\mathbf{e})_c}. \tag{15}$$

$$\sigma_3(\mathbf{e}_q^l, \mathbf{e}) = \Delta_{\mathcal{S}}(\mathbf{e}_q^l, \mathbf{e}) + \operatorname{sgn}_{r}(\mathbf{e}, \mathbf{e}_q^l) \sum_{i=\min_{r}(\mathbf{e},\mathbf{e}_q^l)}^{\max_{r}(\mathbf{e},\mathbf{e}_q^l)-1} (\mathbf{a}_i - \mathbf{b}_i). \tag{16}$$

$$S_2^2(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2) = \begin{cases} \sigma_1(\mathbf{e}_q^l, \mathbf{e}_2) - \sigma_2(\mathbf{e}_2, \mathbf{e}_1) - \sigma_3(\mathbf{e}_1, \mathbf{e}_q^l) + \nu(\mathbf{e}_2) - \Delta_{\mathcal{S}}(\mathbf{e}_q^l, \mathbf{e}_1) & \text{if case A2,} \\ \sigma_1(\mathbf{e}_q^l, \mathbf{e}_2) - \sigma_3(\mathbf{e}_2, \mathbf{e}_1) + \sigma_2(\mathbf{e}_1, \mathbf{e}_q^l) + \nu(\mathbf{e}_2) + \Delta_{\mathcal{S}}(\mathbf{e}_q^l, \mathbf{e}_1) & \text{if case B2,} \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

$$S_3^2(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}_1, \mathbf{e}_2) = \begin{cases} \sigma_3(\mathbf{e}_{q'}^l, \mathbf{e}_2) - \sigma_1(\mathbf{e}_2, \mathbf{e}_1) + \sigma_3(\mathbf{e}_1, \mathbf{e}_q^l) + \Delta_{\mathcal{S}}(\mathbf{e}_2, \mathbf{e}_1) & \text{if case A3,} \\ \sigma_2(\mathbf{e}_{q'}^l, \mathbf{e}_2) + \sigma_2(\mathbf{e}_2, \mathbf{e}_1) + \sigma_3(\mathbf{e}_1, \mathbf{e}_q^l) - \nu(\mathbf{e}_2) - \nu(\mathbf{e}_1) & \text{if case B3,} \\ \sigma_2(\mathbf{e}_{q'}^l, \mathbf{e}_2) + \sigma_3(\mathbf{e}_2, \mathbf{e}_1) - \sigma_2(\mathbf{e}_1, \mathbf{e}_q^l) - \Delta_{\mathcal{S}}(\mathbf{e}_2, \mathbf{e}_1) & \text{if case C3,} \\ \infty & \text{otherwise.} \end{cases} \tag{21}$$

$$\Lambda_2(\mathbf{e}_q^l) = \sum_{\mathbf{e} \in E_p} N_2^1(\mathbf{e}_q^l, \mathbf{e}) D^{-\frac{1}{2} L_2^{1,actual}(\mathbf{e}_q^l, \mathbf{e})} + \sum_{(\mathbf{e}_1, \mathbf{e}_2) \in \hat{E}_p^2} N_2^2(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2) D^{-\frac{1}{2} L_2^{2,actual}(\mathbf{e}_q^l, \mathbf{e}_1, \mathbf{e}_2)}. \tag{24}$$

$$\begin{aligned} \Lambda_3(\mathbf{e}_q^l, \mathbf{e}_{q'}^l) = {} & N_3^0(\mathbf{e}_q^l, \mathbf{e}_{q'}^l) D^{-\frac{1}{2} L_3^{0,actual}(\mathbf{e}_q^l, \mathbf{e}_{q'}^l)} + \sum_{\mathbf{e} \in E_p} N_3^1(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}) D^{-\frac{1}{2} L_3^{1,actual}(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e})} \\ & + \sum_{(\mathbf{e}_1, \mathbf{e}_2) \in \hat{E}_p^2} N_3^2(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}_1, \mathbf{e}_2) D^{-\frac{1}{2} L_3^{2,actual}(\mathbf{e}_q^l, \mathbf{e}_{q'}^l, \mathbf{e}_1, \mathbf{e}_2)}. \end{aligned} \tag{25}$$

comes from the fact that $\binom{n}{k} \approx \frac{n^k}{k!}$ for large n. Thus, for a given code rate and the degree distribution, the complexity is roughly in the order of $n^{2d_{max}-2}$. For the joint method of the structured code case, we modify the definition of d_l and M_l as the degree and the number of possible positions for new entry placement of the lth column in the characteristic matrix $C_{H,tr}$, respectively. The number of nodes in a column with M possible positions for $d-2$ additional entries of the trellis of a structured code case is given by $B^{d-2}\binom{M}{d-2}$. Then, the number of node processing in the trellis search, N_b, can be approximated as (28), where $m_b = n_b - k_b = m/B$ is the number of rows in $C_{H,tr}$. Then, similarly to (27), we obtain (29). From (29), it is easily seen that the complexity of the structured code case is basically of the same order as the non-structured code case with respect to the codeword length. However, it decays as the block matrix size B increases. In Table I, the computation time of the proposed algorithm for several example codes is shown when $B = 1, 8, 16$, and 32. We used MATLAB software on a 3.0GHz Pentium 4 PC. Although the computation time of the proposed algorithm is much larger than that of the PEG algorithm when $B = 1$, the computation time of the proposed algorithm for the structured code case decreases quickly as B increases (approximately decreases with B^2 as expected in (29)). For comparison, the computation times of the structured-code design algorithm in [15] are also given for $B = 8, 16$, and 32. Although the proposed algorithm requires more computation time than the algorithm in [15], the required computation time of the proposed algorithm is not a burden (within several minutes) for the codes with codeword lengths of interest in practical applications (up to a few thousands, especially for low-rate codes) if the block matrix size is chosen sufficiently large.

TABLE I
COMPUTATION TIME OF THE PROPOSED ALGORITHM FOR SEVERAL EXAMPLE CODES.

	Elapsed time		
Code size (n,k)	(512, 128)	(1024, 256)	(2048,512)
Conventional PEG (B=1)	120 s	280 s	1550 s
Structured [15] (B=8)	5 s	20 s	83 s
Structured [15] (B=16)	2 s	8 s	36 s
Structured [15] (B=32)	1 s	4 s	16 s
Proposed (B=1)	2990 s	10990 s	86080 s
Proposed (B=8)	45 s	270 s	1500 s
Proposed (B=16)	11 s	60 s	350 s
Proposed (B=32)	3 s	11 s	87 s

For evaluating the performance and the design complexity of the proposed algorithm, non-structured and structured LDPC codes with code rate 1/3 and 1/4 are tested. The column degree profile is chosen as $\{2, 4, 20\}$ and the degree distribution of each code is obtained using the density evolution technique [5] as

$$\begin{aligned} \lambda_{1/3}(x) &= 0.2375x + 0.509375x^3 + 0.253125x^{19}, \\ \rho_{1/3}(x) &= 0.17x^5 + 0.83x^6, \\ \lambda_{1/4}(x) &= 0.3003x + 0.4674x^3 + 0.2323x^{19}, \\ \rho_{1/4}(x) &= 0.3435x^3 + 0.6565x^4. \end{aligned} \tag{30}$$

Also, we set D in (7) to 10. Fig. 7 shows the performance of the constructed codes using the proposed algorithm in an AWGN channel with 100 iterations in the message passing decoder. Here, 'DP.', 'DP. BL', and 'Conv.' denote the codes constructed from the proposed algorithm for the non-structured code, the proposed algorithm for the structured code with $B = 16$, and the conventional PEG algorithm (non-structured), respectively. The code designed by the conventional method

$$\begin{aligned} N = &\binom{M_{l_1}}{d_{l_1}-2} + \binom{d_{l_1+1}-2}{2}\binom{\sum_{i=0}^{1}(d_{l_1+i}-2)}{2}\binom{M_{l_1}}{d_{l_1}-2}\binom{M_{l_1+1}}{d_{l_1+1}-2} + \dots \\ &+ \binom{d_m-2}{2}\binom{\sum_{l\le m}(d_l-2)}{2}\binom{M_{m-1}}{d_{m-1}-2}\binom{M_m}{d_m-2} \\ = &\binom{M_{l_1}}{d_{l_1}-2} + \sum_{l=l_1+1}^{m}\binom{d_l-2}{2}\binom{\sum_{l'\le l-1}(d_{l'}-2)}{2}\binom{M_{l-1}}{d_{l-1}-2}\binom{M_l}{d_l-2}. \end{aligned} \tag{26}$$

$$\begin{aligned} N &\le \sum_{l=l_1}^{m}\frac{(d_l-2)^4 l^2}{4}\binom{M_l}{d_l-2}^2 \approx \frac{(d_{max}-2)^4 m^2}{4}\binom{m-2}{d_{max}-2}^2 \\ &\approx \frac{(d_{max}-2)^4 m^{2d_{max}-2}}{4((d_{max}-2)!)^2} = \frac{(d_{max}-2)^4((1-r)n)^{2d_{max}-2}}{4((d_{max}-2)!)^2}. \end{aligned} \tag{27}$$

$$N_b \approx \left[B^{d_{l_1}-2}\binom{M_{l_1}}{d_{l_1}-2} + \sum_{l=l_1+1}^{m_b} B^{d_{l-1}+d_l-4}\binom{d_l-2}{2}\binom{\sum_{l'\le l-1}(d_{l'}-2)}{2}\binom{M_{l-1}}{d_{l-1}-2}\binom{M_l}{d_l-2}\right]. \tag{28}$$

$$N_b \approx B^{2d_{max}-4}\frac{(d_{max}-2)^4}{4}\frac{((1-r)n_b)^{2d_{max}-2}}{((d-2)!)^2} = \frac{(d_{max}-2)^4((1-r)n)^{2d_{max}-2}}{4B^2((d_{max}-2)!)^2}. \tag{29}$$

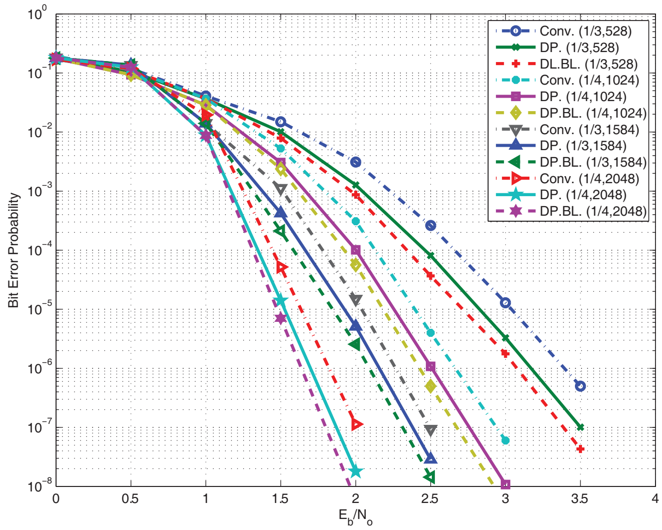


Fig. 7. The performance of the designed codes in an AWGN channel.

was constructed by applying the PEG algorithm for a base graph containing an echelon form.

From the results, it can be seen that the performance of the proposed LDPC codes outperforms the conventionally constructed LDPC codes. It is due to the fact that the lengths of overall cycles formed by low degree columns are considered in the proposed algorithm while the conventional algorithm does not take it fully into account. Also, the additional performance improvement of the structured code is attributed to the fact that a joint search among B consecutive edges in a block matrix improves the optimization result. However, the performance can be degraded if B is too large such that the degree distribution is far from the optimal one.

V. Conclusion

In this paper, an efficient code design using trellis search was proposed for low-rate LDPC codes with an easy-encoding structure. Compared to the conventional controlled random construction, the proposed algorithm eliminates small cycles associated with low-degree columns more effectively without significant increase in complexity since the proposed algorithm exploits a simple evaluation method of the lengths of cycles for a parity-check matrix using an echelon form. Simulation results showed that the codes constructed from the proposed algorithm outperform the codes constructed from the conventional PEG algorithm and that the design complexity of the proposed algorithm is manageable if a structured code with appropriate size of block matrix is used.

References

[1] B. Classon, *et al.*, “LDPC coding for OFDMA PHY,” IEEE C802.16e-05/066r3, Jan. 2005.

[2] T. J. Richardson and R. L. Urbanke, “Efficient encoding of low-density parity-check codes,” *IEEE Trans. Inform. Theory*, vol. 47, pp. 638-656, Feb. 2001.

[3] D. J. C. MacKay, S. T. Wilson, and M. C. Davey, ”Comparison of constructions of irregular Gallager codes,” *IEEE Trans. Commun.*, vol. 47, pp. 1449-1454, Oct. 1999.

[4] M. Yang, W. E. Ryan, and Y. Li, “Design of efficiently encodable moderate-length high-rate irregular LDPC codes,” *IEEE Trans. Commun.*, vol. 52, pp. 564-571, Apr. 2004.

[5] T. Richardson, M. Shokrollahi, and R. Urbanke, “Design of capacity-approaching irregular low-density parity-check codes,” *IEEE Trans. Inform. Theory*, vol. 47, pp. 638-656, Feb. 2001.

[6] [Online]. Available: http://lthcwww.epfl.ch/research/ldpcopt/

[7] E. Shasha, S. Litsyn, and E. Sharon, “Multi-rate LDPC code for OFDMA PHY,” *IEEE C802.16e-04/361*, Aug. 2004.

[8] X. Hu, E. Eleftheriou, and D. Arnold, “Regular and irregular progressive edge-growth Tanner graphs,” *IEEE Trans. Inform. Theory*, vol. 51, pp. 386-398, Jan. 2005.

[9] S. H. Lee, D. Rhee, I. M. Byun, and K. S. Kim, “Use of dynamic programming for the design of irregular LDPC codes,” in *Proc. IEEE Inter. Symp. Inform. Theory Appl. (ISITA)*, Oct. 2006, pp. 377-380.

[10] J. Thorpe, “Design of LDPC graphs for hardware implementation,” in *Proc. IEEE Inter. Symp. Inform. Theory. (ISIT)*, June 2002, p. 483.

[11] E. Jacobsen, “LDPC FEC for IEEE 802.11n applications,” IEEE 802.11-03/0865r1, Nov. 2003.

[12] E. Horowitz and S. Sahni, *Fundamentals of Computer Algorithms*. Computer Science Press, 1978.

[13] L. Dinoi, F. Sottile, and S. Benedetto, “Design of variable-rate irregular LDPC codes with low error floor,” in *Proc. IEEE ICC*, May 2005.

[14] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*. Prentice-Hall, 1983.

[15] K. S. Kim, S. H. Lee, Y. H. Kim, and J. Y. Ahn, “Design of binary LDPC code using cyclic shift matrices,” *IEEE Electron. Lett.*, vol. 40, pp. 325-326, Mar. 2004.

[16] R. M. Tanner, D. Sridhara, and T. E. Fuja, “A class of group-structured LDPC codes,” in *Proc. Inter. Symp. Comm. Theory Appl. (ISCTA)*, July 2001.

[17] M. P. C. Fossorier, “Quasi-cyclic low-density parity-check codes from circulant permutation matrices,” *IEEE Trans. Inform. Theory*, vol. 50, no. 8, pp. 1788-1793, Aug. 2004.

Sang Hyun Lee received the B.S. and M.S. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1999 and 2001. Since 2001, he has been with the Electronics and Telecommunications Research Institute (ETRI) in Daejeon, Korea. His current research interests include channel codes, applications of statistical inference, and information theory.

Kwang Soon Kim was born in Seoul, Korea, on September 20, 1972. He received the B.S. (summa cum laude), M.S.E., and Ph.D. degrees in electrical engineering from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in February 1994, February 1996, and February 1999, respectively. From March 1999 to March 2000, he was with the Department of Electrical and Computer Engineering, the University of California at San Diego, La Jolla, CA, U.S.A., as a Postdoctoral Researcher. From April 2000 to February 2004, he was with the Mobile Telecommunication Research Laboratory, Electronics and Telecommunication Research Institute, Daejeon, Korea, as a Senior Member of Research Staff. In March 2004, he joined the Department of Electrical and Electronic Engineering, Yonsei University, Seoul, Korea, as an Assistant Professor, where he became an Associate Professor in 2009. He was a recipient of the Postdoctoral Fellowship from the Korea Science and Engineering Foundation (KOSEF) in 1999. He received the Outstanding Researcher Award from the Electronics and Telecommunication Research Institute (ETRI) in 2002 and the Jack Neubauer Memorial Award (Best system paper award, IEEE Transactions on Vehicular Technology) from the IEEE Vehicular Technology Society in 2007. His research interests include communication theory, channel coding and iterative decoding, adaptive modulation and coding, cooperative communication, and wireless CDMA/OFDM systems.