Cycle Analysis of Time-variant LDPC Convolutional Codes

Hua Zhou and Norbert Goertz

Institute of Communications and Radio-Frequency Engineering
Vienna University of Technology
Gusshausstrasse 25/389, 1040 Wien, Austria
Email: {Hua.Zhou, Norbert.Goertz}@nt.tuwien.ac.at

***Abstract*—Time-variant Low-Density Parity-Check convolutional codes (LDPCccs) can be derived from unwrapping Quasi-Cyclic (QC) LDPC block codes. Rather than analyzing cycles in a large-scale time-domain parity check matrix, we propose a new way to describe cycle properties in compact form by a "polynomial syndrome former", i.e., a syndrome former in an equivalent polynomial representation. According to the relationship of cycle structures between time- and polynomial-domain syndrome formers, we present a cycle counter algorithm to analyze cycle properties in time-*variant* LDPCccs as well as in time-*in*variant LDPCccs.**

I. Introduction

As a counterpart to classical LDPC block codes, LDPC convolutional codes (LDPCccs) were first proposed in [1]. Studies in [2] have shown that LDPC convolutional codes are suitable for practical implementation with continuous transmission as well as block transmission in frames of arbitrary size. It has been proved that, under pipeline decoding, LDPC convolutional codes have an error performance comparable to that of their block-code counterparts without an increase in computational complexity [3].

LDPC convolutional codes can be separated into two categories, time-invariant and time-variant LDPCccs, with respect to the structure of their syndrome formers. Time-invariant LDPCccs can be derived from Quasi-Cyclic (QC) LDPC block codes [4] and protographs [6], while time-variant ones normally are obtained by unwrapping the parity check matrices of LDPC block codes [1], [8].

In terms of the decoding performance of LDPC convolutional codes, free distance and cycle properties (girth and number of short cycles) are two major issues, which are related to the existence of an error floor and the convergence speed of the Sum Product Algorithm (SPA) [9]. In [5], cycle properties of time-*in*variant LDPC convolutional codes derived from QC LDPC block codes have been analyzed. In this paper, we investigate the cycle formation in time and polynomial domains for time-*variant* LDPC convolutional codes, and we propose a novel algorithm to track cycles in the polynomial syndrome former.

The rest of the paper is organized as follows: In Section II, we review the definition and the syndrome former of an LDPC convolutional code. Section III describes the structure of QC LDPC block codes together with the form of a polynomial matrix for the associated time-*in*variant LDPC convolutional code. In Section IV, we discuss the cycle properties in polynomial syndrome formers of time-*variant* LDPC convolutional codes. Finally, Section V presents an efficient algorithm to track the numbers of cycles with different lengths for time-*variant* LDPC convolutional codes.

II. LDPC Convolutional Codes

A rate $R = b/c$ regular (m_s, J, K) LDPC convolutional code is the set of sequences $\mathbf{v}$ satisfying the equation $\mathbf{v}\mathbf{H}^T$=$\mathbf{0}$, where

$$\mathbf{H}^T = \begin{bmatrix} \ddots & & \ddots & & \\ H_0^T(0) & \dots & H_{m_s}^T(m_s) & & \\ & \ddots & \vdots & \ddots & \\ & & H_0^T(t) & \dots & H_{m_s}^T(t+m_s) \\ & & \ddots & & \ddots \end{bmatrix} \quad (1)$$

This semi-infinite transposed parity check matrix $\mathbf{H}^T$, called syndrome former, is made up of a set of binary matrices $H_i^T(t+i)$ of size $c \times (c-b)$. The syndrome former memory m_s together with the associated constraint length, defined as $v_s = (m_s+1)\cdot c$, is proportional to the decoding complexity. However, to achieve capacity-approaching performance, a large value of m_s is required [3]. $\mathbf{H}^T$ contains exactly J and K ones in each row and column, respectively. For time-invariant LDPC convolutional codes, the binary "sub-matrices" in $\mathbf{H}^T$ are constant, i.e., $H_i^T(i) = H_i^T(i+t)$, $t = 1, 2, 3\dots$, while for periodic time-*variant* LDPC convolutional codes with period T we have $H_i^T(i) = H_i^T(i+T)$.

III. Time-Invariant LDPC Convolutional Codes

A particular category of time-invariant LDPC convolutional codes, described by polynomial matrices, is derived from Quasi-Cyclic (QC) LDPC block codes. As described in [4], in the Galois Field GF(m), with m prime, we assume that a and b are two nonzero elements with multiplicative orders $o(a)$=k and $o(b)$=j, respectively. With the (s,t)th element $P_{s,t}$=$b^s a^t \bmod m$, s=0,1,...,j-1 and t=0,1,...,k-1, we form the $j \times k$

matrix **P** of elements from GF(m):

$$\mathbf{P} = \begin{bmatrix} 1 & a & a^2 & ... & a^{k-1} \\ b & ab & a^2b & ... & a^{k-1}b \\ ... & ... & ... & ... & ... \\ b^{j-1} & ab^{j-1} & a^2b^{j-1} & ... & a^{k-1}b^{j-1} \end{bmatrix}. \quad (2)$$

Given the matrix **P** in (2), the parity check matrix of a QC LDPC block code is generated as follows:

$$\mathbf{H}_{QC} = \begin{bmatrix} I_1 & I_a & I_{a^2} & ... & I_{a^{k-1}} \\ I_b & I_{ab} & I_{a^2b} & ... & I_{a^{k-1}b} \\ ... & ... & ... & ... & ... \\ I_{b^{j-1}} & I_{ab^{j-1}} & I_{a^2b^{j-1}} & ... & I_{a^{k-1}b^{j-1}} \end{bmatrix}. \quad (3)$$

The matrices $I_{P_{s,t}}$ are $m \times m$ identity matrices cyclically shifted to the left by $P_{s,t} - 1$ positions.

According to the description in [4], the associated polynomial matrix derived from the corresponding QC LDPC block code in (3) for the time-invariant LDPCcc is given as

$$\mathbf{H}(D) = \begin{bmatrix} D^0 & D^{a-1} & D^{a^2-1} & ... & D^{a^{k-1}-1} \\ D^{b-1} & D^{ab-1} & D^{a^2b-1} & ... & D^{a^{k-1}b-1} \\ ... & ... & ... & ... & ... \\ D^{b^{j-1}-1} & D^{ab^{j-1}-1} & D^{a^2b^{j-1}-1} & ... & D^{a^{k-1}b^{j-1}-1} \end{bmatrix}. \quad (4)$$

Note that a polynomial matrix generated in this way only consists of monomials[1], which is preferred for code design, since large-weight entries result in short girth [7].

In the Tanner Graph of LDPC codes, short cycles degrade the performance of the iterative decoding algorithm. In [5], a cycle counter algorithm has been introduced to analyze the cycle properties of time-invariant LDPC convolutional codes. Instead of examining cycles in the semi-infinite time-domain syndrome former matrix, which seems unpractical when the matrix has large syndrome former memory, the cycle counter algorithm specifies the form of cycles in a compact polynomial syndrome former.

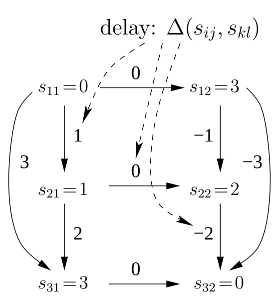
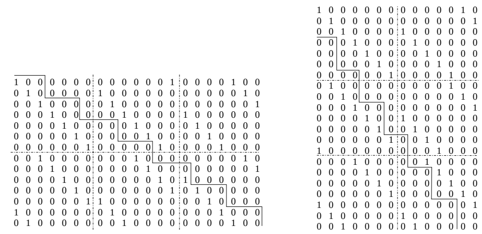


Fig. 1: Delays in polynomial syndrome former in (6)

Here, we use an example to briefly explain the concept. Based on the equations in (2) and (3), with elements $a = 2$ and $b = 6$ picked from GF(7), we obtain the corresponding parity check matrix for the following QC LDPC block code:

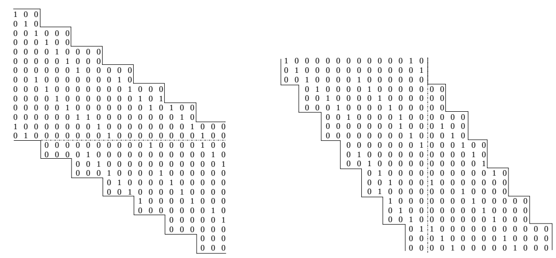
$$\mathbf{H}_{QC} = \begin{bmatrix} I_1 & I_2 & I_4 \\ I_6 & I_5 & I_3 \end{bmatrix}_{2\cdot 7 \times 3\cdot 7}. \quad (5)$$

[1]The term "monomial" is used ambiguously in the literature. In our work, a monomial is essentially denoted by D^l with the integer $l \geq 0$.



(a) $\mathbf{H}_{QC}$ in (5) (b) $\mathbf{H}^T_{QC}$ in (5)

Fig. 2: Parity check matrix of QC LDPC block codes in (5)



(a) **H** matrix (b) Syndrome former

Fig. 3: Unwrapped QC LDPC block codes in (5)

The corresponding polynomial matrix of the time-invariant LDPC convolutional code is given by

$$\mathbf{H}(D) = \begin{bmatrix} D^0 & D^1 & D^3 \\ D^5 & D^4 & D^2 \end{bmatrix} \equiv \begin{bmatrix} 1 & D & D^3 \\ D^3 & D^2 & 1 \end{bmatrix}_{2\times 3} \quad (6)$$

Common factors (such as "D^2" in the upper row) can be equivalently removed from each row of $\mathbf{H}(D)$, details can be found in [5].

The connection property between monomials in (6) is illustrated by Fig. 1: the element s_{ij} is the power of D from the polynomial syndrome former of size 3×2 (transposed $\mathbf{H}(D)$ in (6)) in the ith row and the jth column.

As shown in Fig. 1 for time-invariant LDPCccs, a horizontal edge in the time domain syndrome former $\mathbf{H}^T$ corresponds to a connection between two monomials in the same row of $\mathbf{H}^T(D)$, and there is no time-difference[2] ("delay") between them, denoted by a '0' over an arrow in Fig. 1. In contrast, a vertical edge in $\mathbf{H}^T$ corresponds to a vertical connection between two monomials in $\mathbf{H}^T(D)$ with a delay equal to the difference (including sign) of the "powers" of the two monomials, indicated by a number at an arrow in Fig. 1 [5].

IV. Time-Variant LDPC Convolutional Codes

In [1], the derivation of periodic time-variant LDPC convolutional codes from LDPC block codes was proposed. The general principle is to unwrap the parity check matrix of a LDPC block code and duplicate the unwrapped parity check matrix to infinity along the diagonal. This unwrapping method is also applied to QC LDPC block codes in [8].

Normally, given a QC LDPC block code with rate $R = b/c$ and identity matrix size m, the unwrapping step size is chosen

[2]Time indices can be considered as the column indices of the time-domain transposed parity check matrix $\mathbf{H}^T$ in (1).

to be $(c-b)k\times ck$, $0<k\leqslant m$, $k\in\mathbb{Z}^{+}$; when k=1, the period of time-variant LDPCcc is T=m. The unwrapping process applied to the QC LDPC block code in (5) is illustrated by Figs. 2 and 3, with k being set to one. Fig. 2 gives the original and transposed parity check matrices of the QC LDPC block code in (5), consequently, the unwrapping size is modified into $c\times(c-b)$ in Fig. 2(b). After unwrapping, we get the corresponding one-period parity check matrix and syndrome former for the LDPC convolutional code in Fig. 3(a) and Fig. 3(b), respectively. For consistency, we will analyze the cycle properties in the syndrome former matrix. If we repeat the "period-one" syndrome former to infinity along the diagonal, we obtain the time-domain semi-infinite syndrome former matrix of this LDPC convolutional code in Fig. 4. In addition, we show a 12-cycle in the syndrome former matrix, which will be mapped to polynomial syndrome former later.

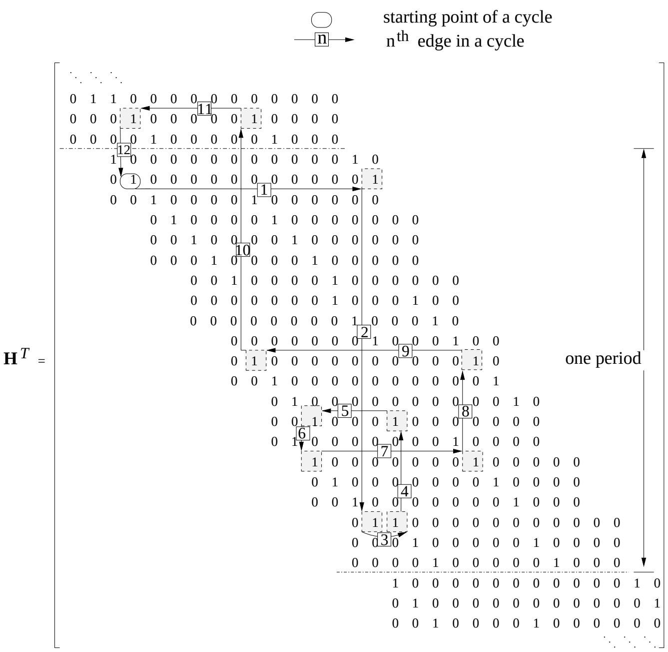


Fig. 4: Syndrome former of the derived LDPCcc

Due to the infinite size of the time-domain syndrome former matrix, it is inconvenient to analyze edge connections. However, we can convert it to a finite compact polynomial syndrome former. The process is summarized as follows.

For a rate $R=b/c$ periodic time-variant LDPC convolutional code with period T, given the time-domain syndrome former $\mathbf{H}^T$ in (1) and syndrome former memory m_s, the polynomial syndrome former is given as:

$$\mathbf{H}^T(D)=\begin{bmatrix} H_1^T(D)\\ H_2^T(D)\\ \vdots\\ H_t^T(D)\\ \vdots\\ H_T^T(D)\end{bmatrix}, \tag{7}$$

where each sub-polynomial matrix is

$$H_t^T(D)=\sum_{n=0}^{m_s} H_n^T(t+n)\cdot D^n \tag{8}$$

with $H_n^T(t+n)$ given in (1). Note that, in (7) the superscript T means transpose while the subscript T refers to the period. Finally, $\mathbf{H}^T(D)$ consists of T distinct sub-polynomial matrices, and each is of size $c\times(c-b)$.

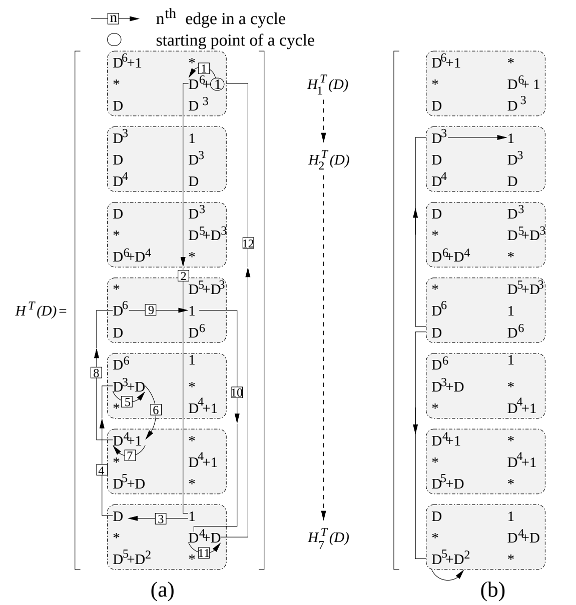


Fig. 5: Polynomial syndrome former of the derived LDPCcc

Similar to time-invariant LDPC convolutional codes, two monomials with powers $s_{ij}^p(t)$ and $s_{ij'}^q(t)$ from the same row (same sub-polynomial matrix in (8)) in $\mathbf{H}^T(D)$ of a time-variant LDPC convolutional code connect to each other without delay, i.e.,

$$\Delta(s_{ij}^p(t), s_{ij'}^q(t))\doteq 0 \tag{9}$$

where $s_{ij}^p(t)$ refers to the power of the monomial in the polynomial entry, which is in the ith row and jth column in the tth sub-polynomial matrix $H_t^T(D)$, while p indicates the pth monomial in this polynomial entry[3].

In time-invariant LDPC convolutional codes, monomials in the same column are always connected. However, for a time-variant LDPCcc with period T, two monomials with powers $s_{ij}^p(t)$ and $s_{i'j}^q(t')$ in the same column in $\mathbf{H}^T(D)$ are connected if and only if

$$(t-t'+s_{ij}^p(t)-s_{i'j}^q(t')) \mod T=0 \tag{10}$$

where $t\neq t'$ in (10) and the delay is given by

$$\Delta(s_{ij}^p(t), s_{i'j}^q(t'))\doteq s_{i'j}^q(t')-s_{ij}^p(t)\ . \tag{11}$$

[3]for instance, D^3 and 1 are the first and second monomials in a polynomial entry D^3+1, respectively.

A path $\mathcal{P}_v$ in $\mathbf{H}^T(D)$ of a time-variant LDPC convolutional code – which is a sequence of pairs of elements from the polynomial syndrome former matrix – forms a cycle of length $2 \times L$, when

$$\sum_{\forall \{s,s'\} \in \mathcal{P}_v} \Delta(s, s') = 0 \tag{12}$$

with the path given by

$$\begin{aligned} \mathcal{P}_v = \{ & \underbrace{\{\overbrace{s^{p_1}_{i_1 j_1}(t_1)}^{\text{start}}, s^{q_1}_{i_1 j_2}(t_1)\}}_{\text{horizontal}}, \underbrace{\{s^{q_1}_{i_1 j_2}(t_1), s^{p_2}_{i_2 j_2}(t_2)\}}_{\text{vertical}}, ... \\ & \underbrace{\{s^{p_{L-1}}_{i_{L-1} j_{L-1}}(t_{L-1}), s^{q_{L-1}}_{i_{L-1} j_L}(t_{L-1})\}}_{\text{horizontal}} \\ & \underbrace{\{s^{q_{L-1}}_{i_{L-1} j_L = j_1}(t_{L-1}), \overbrace{s^{p_L = p_1}_{i_1 j_1}(t_L = t_1)}^{\text{end}}\}}_{\text{vertical}} \} \end{aligned} \tag{13}$$

with $i_x \in \{1, ..., c\}$, $j_y \in \{1, ..., c-b\}$, $t_x \in \{1, ..., T\}$, $p_x \in \mathbb{Z}^+$, $q_y \in \mathbb{Z}^+$ and $x \in \{1, ..., L\}$ and $y \in \{1, ..., L\}$. As for time-*in*variant LDPC convolutional LDPC codes, the first and the last elements in the path have to be the same in order for $\mathcal{P}_v$ to form a cycle.

To illustrate the above concept, we apply the formulas in (7) and (8) to the time-domain syndrome former in Fig. 4; we obtain the polynomial syndrome former and the corresponding 12-cycle mapping from Fig. 4 to Fig. 5(a). This polynomial syndrome former contains seven sub-polynomial matrices, i.e., period $T = 7$, and each is of size (3×2). According to (12) and (13), the accumulated delay of the path in Fig. 5(a) is given as

$$\sum_{\forall \{s,s'\} \in \mathcal{P}_v} \Delta(s, s') =$$

$$\left\{ \begin{array}{lclcl} \overbrace{\Delta(s^2_{22}(1), s^1_{22}(1))}^{\text{horizontal moves}} & + & \overbrace{\Delta(s^1_{22}(1), s^1_{12}(7))}^{\text{vertical moves}} & + \\ \Delta(s^1_{12}(7), s^1_{11}(7)) & + & \Delta(s^1_{11}(7), s^1_{21}(5)) & + \\ \Delta(s^1_{21}(5), s^2_{21}(5)) & + & \Delta(s^2_{21}(5), s^2_{11}(6)) & + \\ \Delta(s^2_{11}(6), s^1_{11}(6)) & + & \Delta(s^1_{11}(6), s^1_{21}(4)) & + \\ \Delta(s^1_{21}(4), s^1_{22}(4)) & + & \Delta(s^1_{22}(4), s^1_{22}(7)) & + \\ \Delta(s^1_{22}(7), s^2_{22}(7)) & + & \Delta(s^2_{22}(7), s^2_{22}(1)) & \end{array} \right.$$

With (9) we have "zero" delay for all horizontal moves and we use (11) for the vertical moves. We obtain

$$\sum_{\forall \{s,s'\} \in \mathcal{P}_v} \Delta(s, s') = \left\{ \begin{array}{ccccc} 0 & + & (0-6) & + \\ 0 & + & (3-1) & + \\ 0 & + & (0-1) & + \\ 0 & + & (6-4) & + \\ 0 & + & (4-0) & + \\ 0 & + & (0-1) & \end{array} \right\} = 0$$

which confirms the condition in (12): this 12-cycle does indeed exist.

V. Cycle Counter Algorithm

In the previous section, we have discussed how a cycle is formed for a time-variant LDPC convolutional code both in the time- and polynomial-domains. In this section, we introduce a cycle counter algorithm to track cycles in a LDPCcc given its polynomial syndrome former matrix. Similar to time-invariant LDPCccs, given a "starting" monomial as shown in Fig. 4 and Fig. 5(a), this algorithm extends all the possible two consecutive edges (vertical and horizontal) with delays in (9) and (11); we consider these two path extensions as "one" iteration. Furthermore, connectivity of monomials in the same column of the polynomial syndrome former refers to (10).

For the description of the cycle counter algorithm, we define the notation as follows:

- $s^p_{ij}(t)$: power of the pth monomial in the polynomial entry from the ith row and the jth column in $H^T_t(D)$. Throughout this paper, we refer to $s^p_{ij}(t)$ as "a monomial" or "the power of a monomial", alternatively.
- $\mathcal{N}_v(s^p_{ij}(t))$: all the vertically connectable neighbors of monomial $s^p_{ij}(t)$ in $\mathbf{H}^T(D)$.
- $\mathcal{N}_h(s^p_{ij}(t))$: all the horizontally connectable neighbors of monomial $s^p_{ij}(t)$ in $\mathbf{H}^T(D)$.
- $\mathbf{N}_t\{i, j\}\{1, p\}$: current accumulative powers' sum (APS) [5] for all paths temporarily intermittent in monomial $s^p_{ij}(t)$.
- $\mathcal{W}(\mathbf{N}_t\{i, j\}\{1, q\})$: number of paths temporarily intermittent at monomial $s^q_{ij}(t)$.
- $\mathbf{S}$: the set of starting monomials; for each iteration, after extending all the possible two consecutive paths (vertical and horizontal) from this starting monomial, the set of starting monomials is redefined as those monomials that all the paths temporarily end with in this iteration.

The main challenge of applying this cycle counter algorithm lies in generating a register, which informs a monomial of its connectable neighboring monomials according to (10). Each iteration consists of two-step updates. Firstly, update the set of starting monomials by vertically extending the paths to all their connectible neighboring monomials. Afterwards, update the monomials, that currently all the paths intermittent at, by extending paths horizontally. At the end of each iteration, the path extension history is cleared for this iteration and then the set of starting monomials is redefined as those monomials that all the horizontally extended paths currently end with. The process runs iteratively until the maximum testing cycle length is achieved; full details can be found in Algorithm 1.

As an example shown in Fig. 5(b) we select $s^1_{31}(4)$=D as the starting monomial in the first iteration, i.e., $\mathbf{S}$=$s^1_{31}(4)$. First of all, the cycle counter algorithm updates monomial $s^1_{31}(4)$ by extending the only two vertical available connections to $s^1_{11}(2)$ and $s^1_{31}(7)$, respectively. Then it updates monomials $s^1_{11}(2)$ and $s^1_{31}(7)$ by extending paths to $s^1_{12}(2)$ and $s^2_{31}(7)$. Before starting the next iteration, the set of starting monomials is renewed as $\mathbf{S}$=$\{s^1_{12}(2), s^2_{31}(7)\}$ and the previous search-history is cleared.

In the first iteration of Step 2 in Algorithm 1, we test every monomial in $\mathbf{H}^T(D)$ as a starting monomial, which results in repetitious tracking of a cycle. In addition, short cycles with length i in the LDPCccs usually have i distinct vertices in the time-domain syndrome former. Therefore, C_i/i in Step 3 gives the exact number of cycles for short cycle length. However, considering larger i, it is possible that an i-cycle exists that consists of less than i distinct vertices when a cycle contains smaller inner cycles. This explains the ceil manipulation $\lceil\rceil$ in Step 3 of the algorithm.

Algorithm 1 Cycle counter for time-variant LDPCccs

- Step 1: Initialization
  1) Define the maximum cycle length as $2 \times L$ for examining and the initial path length as $l = 0$.
  2) Generate a $KT \times J$ empty cell $\mathbf{N}\{\}$ used to update the accumulative powers' sum (APS) and create a cycle counter C_i recording number of cycles with different lengths i, $i = 4, 6, 8, ..., 2 \times L$.
  3) Generate a monomials' "connectivity cell", which is used as a reference for $\mathcal{N}_v(s_{ij}^p(t))$ to indicate the indices of neighbors of $s_{ij}^p(t)$ in the polynomial syndrome former.
- Step 2: Main function.

for each monomial $s_{i^*j^*}^{q^*}(t^*)$ in $\mathbf{H}^T(D)$ **do**
  $\mathbf{S}$=$s_{i^*j^*}^{q^*}(t^*)$; $\mathbf{N}_{t^*}\{i^*, j^*\}\{1, q^*\}$=0;
  while $l < L$ **do**
    for each $s_{ij}^q(t) \in \mathbf{S}$ **do**
      %vertical path extensions
      for each $s_{i'j}^p(t') \in \mathcal{N}_v(s_{ij}^q(t))$ **do**
        if $\mathbf{N}_t\{i, j\}\{1, q\}$=$\varnothing$ **then**
          $\mathbf{N}_t\{i, j\}\{1, q\}$=0;
        end if
        k=$\mathcal{W}(\mathbf{N}_t\{i, j\}\{1, q\})$;
        $\mathbf{T_1}$=$s_{i'j}^p(t')$-$s_{ij}^q(t)$+$\mathbf{N}_t\{i, j\}\{1, q\}$(1:$k$);
        $\mathbf{N}_{t'}\{i', j\}\{1, p\}$=[$\mathbf{N}_{t'}\{i', j\}\{1, p\}$|$\mathbf{T_1}$];
        %horizontal path extensions
        for each $s_{i'j'}^r(t') \in \mathcal{N}_h(s_{i'j}^p(t'))$ **do**
          if $i' \neq j'|r \neq p$ **then**
            $\mathbf{T_2}$=$\mathbf{N}_{t'}\{i', j\}\{1, p\}$(end-$k$+1:end);
            $\mathbf{N}_{t'}\{i', j'\}\{1, r\}$=[$\mathbf{N}_{t'}\{i', j'\}\{1, r\}$|$\mathbf{T_2}$];
            $\mathbf{S}$=$\mathbf{S} \cup s_{i'j'}^r(t')$;
          end if
        end for
        %clear history of horizontal path extensions
        $\mathbf{N}_{t'}\{i', j\}\{1, p\}$(end-$k$+1:end)=[];
      end for
      $\mathbf{S}$=$\mathbf{S} \cap s_{ij}^q(t)$; %renew starting monomials
      %clear history of vertical path extensions
      $\mathbf{N}_t\{i, j\}\{1, q\}$(1:$k$)=[];
    end for
    $\mathbf{S}$=unique($\mathbf{S}$); %remove repetitive monomials
    Calculate zeros in $\mathbf{N}_{t^*}\{i^*, j^*\}\{1, q^*\}$;
    l=l+2; C_l=C_l+n;
  end while
end for

- Step 3: Process cycle counter C_i.
  The number of cycles with length-i in this syndrome former is defined as $\lceil C_i/i \rceil$. C_i is the number of cycles for length-i.

We apply the cycle counter algorithm to a couple of time-variant LDPCccs derived from unwrapping (J=3,K=5) QC LDPC block codes in [4] with unwrapping step size $(c-b)\times c$. The cycle properties of these time-variant LDPCccs are shown in Table I with the number of cycles in time-variant LDPCccs normalized by period T, which makes sense to compare cycles properties with time-invariant LDPCccs given in [5].

TABLE I: Cycle properties of LDPCccs

m/m_s	8-cycle		10-cycle		12-cycle	
	time-invariant	time-variant	time-invariant	time-variant	time-invariant	time-variant
31/21	11	4.55	62	42.39	351	228.77
61/57	0	0	21	15.07	148	118.06
181/134	0	0	0	0	67	46.35
211/187	0	0	0	0	68	54.57
241/204	0	0	0	0	52	44.83

Note: common factors have been removed from polynomial syndrome former for time-invariant LDPCccs. C_i has been normalized for time-variant LDPCccs, i.e., C_i/T, where T=m, due to unwrapping size k=1.

Columns in dark gray and light gray correspond to cycle properties of time-variant and time-invariant LDPCccs, respectively. Generally speaking (and as expected), time-*variant* LDPCccs are superior to time-*in*variant LDPCccs with respect to the decrease in the number of cycles.

VI. Conclusion

In this paper, the relationship of cycle formations between time- and polynomial-domain syndrome formers of time-*variant* LDPC convolutional codes, derived from unwrapping QC LDPC block codes, has been analyzed. Based on the connectivities between monomials in the polynomial syndrome former, we present a cycle counter algorithm to examine the cycle properties of time-variant LDPCccs. The maximum achievable cycle-length in the numerical search (limited by algorithmic complexity) depends on the values of J and K as well as period T. In addition, since the algorithm is based on the indices of monomials, apart from time-*variant* LDPCccs, it can also be applied to time-*invariant* LDPC convolutional codes. Furthermore, even though this cycle counter algorithm is proposed to LDPCccs, it is universally applicable to any convolutional code defined by a polynomial syndrome former.

References

[1] A. J. Felström and K. Sh. Zigangirov, "Time-varying periodic convolutional codes with low-density parity-check matrices," *IEEE Trans. Information Theory*, VOL. 45, NO. 6, September 1999.
[2] S. Bates, D. G. Elliot, and R. Swamy, "Termination sequence generation circuits for low-density parity-check convolutional codes," in *IEEE Trans. Circuits and Systems*, VOL. 53, NO. 9, September 2006.
[3] D. J. Costello, Jr., A. E. Pusane, S. Bates, and K. Sh. Zigangirov, "A comparison between LDPC block and convolutional codes," in *Proc. Information Theory and Applications Workshop*, San Diego, USA, 2006.
[4] R. M. Tanner, D. Sridhara, A. Sridharan, T. E. Fuja, and D. J. Costello Jr. "LDPC block and convolutional codes based on circulant matrices," *IEEE Trans. Information Theory*, VOL. 50, NO. 12, December 2004.
[5] Hua Zhou, Norbert Goertz, "Cycle analysis of time-invariant LDPC convolutional codes," *17th International Conference on Telecommunications*, Doha, Qatar, April, 2010.
[6] D. G. M. Mitchell, A. E. Pusane, K. Sh. Zigangirov, and D. J. Costello, "Asymptoticallly good LDPC convolutional codes based on protographs," *IEEE Intern. Symp. on Inform. Theory*, Toronto, Canada, July 6, 2008.
[7] A. Sridharan, "Design and analysis of LDPC convolutional codes," Phd Dissertation, University of Notre Dame, Indiana, USA, Feb 2005.
[8] A. E. Pusane, R. Smarandache, P. O. Vontobel, D.J. Costello "On deriving good LDPC convolutional codes from QC LDPC block codes," *IEEE Intern. Symp. on Inform. Theory*, Nice, France, June, 2007.
[9] S. Lin and D. J. Costello, "Error Control Coding," Pearson Prentice Hall, 2004