Generalized Quasi-Cyclic LDPC Codes: Design and Efficient Encoding

Roxana Smarandache*†, Anthony Gómez Fonseca†, and David G. M. Mitchell‡
Departments of Mathematics*and Electrical Engineering†, University of Notre Dame, Notre Dame, IN 46556, USA
Email: {rsmarand, agomezfo}@nd.edu
‡Klipsch School of Electrical and Computer Engineering, New Mexico State University, Las Cruces, NM, USA
Email: dgmm@nmsu.edu

***Abstract*—Generalized low-density parity-check (GLDPC) codes, where single parity-check constraints on the code bits are replaced with generalized constraints (an arbitrary linear code), are a promising class of codes for low-latency communication. The block error rate performance of the GLDPC codes, combined with a complementary outer code, has been shown to outperform a variety of state-of-the-art code and decoder designs with suitable lengths and rates for the 5G ultra-reliable low-latency communication (URLLC) regime. A major drawback of these codes is that it is not known how to construct appropriate polynomial matrices to encode them efficiently. In this paper, we analyze practical constructions of quasi-cyclic GLDPC (QC-GLDPC) codes and show how to construct polynomial generator matrices in various forms using minors of the polynomial matrix. We consider mixed QC-GLDPC constructions, where favorable tradeoffs can be found in code rate vs. error correcting performance by only generalizing a proportion of the constraint nodes, and show that our approach extends naturally to these constructions. Finally, we show that by applying double graph-liftings, the code parameters can be improved without affecting the ability to obtain a polynomial generator matrix.**

I. Introduction

Generalized low-density parity-check (GLDPC) codes, where single parity-check constraints are replaced by a general linear code, have been shown to have several advantages over conventional LDPC codes, including large minimum distance [1]–[3], good iterative decoding performance [4], [5], fast decoding convergence speed [6], and low error floors [7], [8]. The performance and convergence speed of the BP decoder can be further improved by applying a reinforcement learning approach to optimize the scheduling of messages in the decoder [9]. This improved performance from the generalized constraints comes at the cost of reduced coding rate, which is generally not desirable for many communication systems. However, these codes are well suited for next generation machine-to-machine (M2M) type communications and ultra reliable low latency communications (URLLC), which are expected to have low coding rate, e.g., $R = 1/12$, and short block lengths ranging from hundreds of bits to one or two thousand [10].

Conventional quasi-cyclic LDPC (QC-LDPC) codes have a highly structured parity-check matrix, which can be composed as an array of circulant matrices. Consequently, they are attractive for implementation purposes since their structure leads to efficiencies in decoder design [11]. Quasi-cyclic GLDPC (QC-GLDPC) codes were proposed in [7], where irregular protograph-based designs were shown to possess good performance in both the waterfall and error floor regions. In [5], a practical construction of QC-GLDPC codes were proposed for URLLC, where the optimal proportion of generalized Hamming constraints (to minimize the gap to capacity) was determined by an asymptotic analysis. The optimal proportions were found to be 0.75% of the constraint nodes in the $(2, 6)$-regular and $(2, 7)$-regular cases, while this proportion increases to the 0.8% in the $(2, 15)$-regular case. A major drawback of these codes is that it is not known how to leverage the circulant based generalized matrix to construct appropriate polynomial matrices to encode them efficiently.

As a result of the rich structure of QC-LDPC codes, their matrix representations have been studied in a number of works. These include methods to construct a generator in standard form (e.g., [12]), which allows high throughput systematic encoding but the resulting generator matrix is typically dense.[1] The sparse parity-check matrix is often represented as an array of circulant permutations, which faciliate efficient implementation but will typically have a number of linearly dependent rows. Although Gaussian elimination can be employed to compute the rank with a complexity of $\mathcal{O}(n^3)$, it is desirable to have an analytic way to compute the rank, particularly for classes of algebraic QC-GLDPC codes. Methods to compute the rank of QC-LDPC codes have been investigated, including approaches involving Fourier transforms [13] and the matrix polynomial representation [14]. However, these approaches are limited to certain code parameters. In [15], we presented a method to compute the rank of any parity-check matrix representing a QC-LDPC code, and hence the dimension of the code, by using the minors of the corresponding polynomial parity-check matrix. This formula, can be applied to compute the rank of the QC-GLDPC codes as well.

In this paper, we extend our results from [15] and show how the approach extends naturally to QC-GLDPC codes. Using the QC-GLDPC constructions from [5] and [7] as examples, we demonstrate how to compute polynomial generator matrices, allowing efficient encoding. It is shown that the method also applies to mixed constructions, where the parity-check

[1]Systematic encoding, where the information is directly embedded in the codeword, is often preferred in practice.

matrix has a mixture of generalized and single parity-check constraints. We consider the design of these codes and show that they can have large minimum distance. In particular, the generator matrices will often have rows that are equal (or close to) the minimum distance of the code, giving tight upper bounds that are hard to obtain otherwise. We discuss how the approach can also be used to determine the rank of the parity-check matrix of QC-GLDPC codes. Throughout the paper, we show that by applying double graph-liftings, the code parameters can be improved without affecting the ability to obtain a polynomial generator matrix. Importantly, this allows the code designer to finely tune the number of generalized constraints to optimize the performance/rate trade-off.

II. Definitions, notations and background

We use the following notation. For any positive integer L, $[L]$ denotes the set $\{1, 2, \ldots, L\}$. For any matrix M, we let $M_{\mathcal{I},\mathcal{J}}$ be the sub-matrix of M that contains only the rows of M whose index appears in the set $\mathcal{I}$ and only the columns of M whose index appears in the set $\mathcal{J}$; if $\mathcal{I}$ equals the set of all row indices of M, we will simply write $M_{\mathcal{J}}$. We use the shorthand $M_{\mathcal{J}\setminus i}$ for $M_{\mathcal{J}\setminus\{i\}}$. If $\mathcal{I}$ and $\mathcal{J}$ have the same cardinality, we use $\Delta_{\mathcal{I},\mathcal{J}} = \det(H_{\mathcal{I},\mathcal{J}})$, and $\Delta_{\mathcal{J}} = \det(H_{[n_c],\mathcal{J}})$.

As usual, an LDPC code $\mathcal{C}$ is described as the nullspace of a parity-check matrix H to which we associate a Tanner graph [16] in the usual way. The girth $\mathrm{girth}(H)$ of a graph is the length of the shortest cycle in the graph.

A. Protographs and polynomial representations

A protograph [17], [18] is a small bipartite graph represented by an $n_c \times n_v$ parity-check or *base* biadjacency matrix B with non-negative integer entries b_{ij}. The parity-check matrix H of a protograph-based LDPC block code can be created by replacing each non-zero entry b_{ij} by a sum of b_{ij} non-overlapping $N \times N$ permutation matrices and a zero entry by the $N \times N$ all-zero matrix. Graphically, this operation is equivalent to taking an N-fold graph cover, or "lifting", of the protograph. We denote the $N \times N$ circulant permutation matrix where the entries of the $N \times N$ identity matrix I are shifted to the left by r positions modulo N as I_r.

A QC-LDPC code of length $n = n_v N$ is a protograph-based LDPC code, for which the $N \times N$ lifting permutation matrices are all circulant matrices I_r. Thus a QC code has an $n_c N \times n_v N$ parity-check matrix H of the form

$$H = \begin{bmatrix} H_{1,1} & H_{1,2} & \cdots & H_{1,n_v} \\ \vdots & \vdots & \ddots & \vdots \\ H_{n_c,1} & H_{n_c,2} & \cdots & H_{n_c,n_v} \end{bmatrix}, \qquad (1)$$

where the $N \times N$ sub-matrices $H_{i,j}$ are circulant; applying equal circular shifts to each length-N sub-blocks of a codeword results in a codeword. With the help of the well-known isomorphism between the ring of circulant matrices over the binary field $\mathbb{F}_2$ and the ring $\mathbb{F}_2[x]/(x^N-1)$ of $\mathbb{F}_2$-polynomials modulo $x^N - 1$ (see, *e.g.*, [19]), a QC LDPC code can also be described by an $n_c \times n_v$ polynomial parity-check matrix over $\mathbb{F}_2[x]/(x^N - 1)$. In particular, with the $n_c N \times n_v N$ parity-check matrix H described above we associate the polynomial parity-check matrix $H(x) \in \mathbb{F}_2^{\langle N \rangle}[x]^{n_c \times n_v}$ as

$$H(x) = \begin{bmatrix} h_{1,1}(x) & h_{1,2}(x) & \cdots & h_{1,n_v}(x) \\ \vdots & \vdots & \ddots & \vdots \\ h_{n_c,1}(x) & h_{n_c,2}(x) & \cdots & h_{n_c,n_v}(x) \end{bmatrix}. \qquad (2)$$

Here $h_{i,j}(x) = x^r$ corresponds to the circulant permutation matrix I_r. Moreover, with any vector

$$\mathbf{c} = (c_{1,0}, \ldots, c_{1,N-1}, \ldots, c_{n_v,0}, \ldots, c_{n_v,N-1})$$

in $\mathbb{F}_2^{n_v N}$, we associate the polynomial vector $\mathbf{c}(x) = \big(c_1(x), \ldots, c_{n_v}(x)\big)$ where $c_i(x) \triangleq \sum_{s=0}^{N-1} c_{i,s} x^s$. Then, the condition $H \cdot \mathbf{c}^{\mathsf{T}} = \mathbf{0}^{\mathsf{T}}$ (in $\mathbb{F}_2$) is equivalent to $H(x) \cdot \mathbf{c}(x)^{\mathsf{T}} = \mathbf{0}^{\mathsf{T}}$ in $\mathbb{F}_2[x]/(x^N - 1)$.

B. Minors and equivalent matrices

In [15] we gave a method (based on polynomial minors of $H(x)$) to construct polynomial generator matrices for QC-LDPC codes. We recall here the part of the theorem that addresses the case in which H has full rank, since it is used in this paper.

Theorem 1 ([15]). *Let H be the $n_c N \times n_v N$ parity-check matrix of a QC code $\mathcal{C}$ and let $H(x)$ be its corresponding polynomial parity-check matrix over $\mathbb{F}_2[x]/(x^N - 1)$.*

If there exists a subset $\mathcal{S}$ of size n_c of $[n_v]$ where, for simplicity and w.l.o.g., we assume that $\mathcal{S} = [n_c]$, such that $\Delta_{\mathcal{S}} = \det\big(H_{\mathcal{S}}(x)\big)$ is invertible in $\mathbb{F}_2[x]/(x^N - 1)$, then the $m \times n_v$ matrix below is a polynomial generator matrix for $\mathcal{C}$,

$$\left[\begin{array}{ccc|c} (\Delta_{S_1\setminus 1})^{\mathsf{T}} & \cdots & (\Delta_{S_1\setminus n_c})^{\mathsf{T}} & \\ \vdots & & \vdots & \mathrm{diag}_m(\Delta_{\mathcal{S}}) \\ (\Delta_{S_m\setminus 1})^{\mathsf{T}} & \cdots & (\Delta_{S_m\setminus n_c})^{\mathsf{T}} & \end{array}\right],$$

where $m \triangleq n_v - n_c$, $S_i = S \cup \{n_c + i\}$, for all $i \in [m]$, $\Delta_{S_i\setminus j} \triangleq \det\big(H_{S_i\setminus j}(x)\big)$, for all $j \in [n_c]$, $\mathrm{diag}_m(\Delta_{\mathcal{S}})$ is a diagonal $m \times m$ matrix with each diagonal entry equal to $\Delta_{\mathcal{S}}$.

C. GLDPC Codes

Let a *component code* corresponding to one of the constraint c_i be represented by a $m^{c_i} \times n^{c_i}$ parity-check matrix. The constraint matrix of a QC-GLDPC code is also represented as (1) and (2), however the full parity-check matrix is obtained by replacing each one entry with the corresponding column of the component parity-check matrix. Consequently, the design rate of the GLDPC code is

$$R = 1 - \frac{\sum_{i=1}^{n_c} m^{c_i}}{n_v}. \qquad (3)$$

Conventionally, BP decoding is performed on the constraint graph, corresponding to (1), where component decoding can be performed as desired, e.g., BCJR, BP, etc., with different performance/complexity trade-offs. This paper focuses on the design and efficient encoding of GLDPC codes.

III. Generator Matrices for QC-GLDPC Codes

In this section, we construct generalized LDPC codes using pre-lifting [20] and results developed in [5], that are likely to have good performance, and show how we can find generator matrices for them.

We start from a $2N \times n_v N$ QC LDPC codes based on the all-one protograph, with $n_v = 6, 7, 15$, therefore, we can take, w.l.o.g., the following constraint matrix

$$H \triangleq \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & x^{i_2} & x^{i_3} & \cdots & x^{i_{n_v}} \end{bmatrix}.$$

A. Case $n_v = 6$

We consider first the case where the first constraint (N check nodes) are all simple nodes and the second constraint node is a shortened Hamming code with parity-check matrix

$$h_{GC} \triangleq \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}, \tag{4}$$

Therefore, the parity-check matrix of the GLDPC code can be written as

$$H_{GC} \triangleq \left[\begin{array}{cccccc} 1 & 1 & 1 & 1 & 1 & 1 \\ \hline 1 & x^{i_2} & 0 & x^{i_4} & 0 & 0 \\ 1 & 0 & x^{i_3} & 0 & x^{i_5} & 0 \\ 0 & x^{i_2} & x^{i_3} & 0 & 0 & x^{i_6} \end{array}\right]. \tag{5}$$

Let $f_1 \triangleq 1+x^{-i_4}+x^{-i_5}$, $f_2 \triangleq 1+x^{i_2-i_4}+x^{i_2-i_6}$, and $f_3 \triangleq 1+x^{i_3-i_5}+x^{i_3-i_6}$. Following Theorem 1, if the gcd between any of f_1, f_2, f_3, and x^N+1 is 1, e.g., $\gcd(f_3, x^N+1) = 1$, then the matrix

$$\begin{bmatrix} f_3^{\mathsf{T}} & 0 & f_1^{\mathsf{T}} & x^{i_4} f_3^{\mathsf{T}} & x^{i_5} f_3^{\mathsf{T}} + x^{i_3-i_5} f_1^{\mathsf{T}} & x^{i_3-i_6} f_1^{\mathsf{T}} \\ 0 & f_3^{\mathsf{T}} & f_2^{\mathsf{T}} & x^{i_4-i_2} f_3^{\mathsf{T}} & x^{i_3-i_5} f_2^{\mathsf{T}} & x^{i_6-i_2} f_3^{\mathsf{T}} + x^{i_3-i_6} f_2^{\mathsf{T}} \end{bmatrix}$$

is a polynomial generator matrix for the code. In the case that this is not true, we add a few naturally occurring codewords to obtain a generator matrix using the method in [15].

Example 2. For $N = 79$, the exponents were taken to be $[i_2, i_3, i_4, i_5, i_6] = [54, 66, 71, 55, 69]$ in [5], for which H has girth 12. We compute f_i and obtain

$$f_1 = 1 + x^8 + x^{24}, f_2 = 1 + x^{62} + x^{64}, f_3 = 1 + x^{11} + x^{76}.$$

Since $\gcd(f_3, x^{79}+1) = 1$ in $\mathbb{F}_2[x]$, f_3 is invertible and the matrix above is a polynomial generator matrix of the code. Note that the weight 18 of the rows of the matrix G is an upper bound to the minimum distance of the GLDPC code, while the actual minimum distance of this code is 16. As such, we obtain a $[475, 158, 16]$ code. □

For this protograph, it was shown in [5] that the iterative decoding threshold was closest to capacity when 75% of the simple check nodes of the lifted $2N \times 6N$ scalar matrix with (4). But this will, in most cases, break the QC structure and make the encoding less efficient. In order to maintain the QC structure, and possibly create a stronger code (with better minimum distance), we will apply some of our previous techniques from [21] concerning double liftings (a "pre-lift" and a "final lift"). First, we must increase the lifting to $N = 90$, which is the first even exponent for which the girth of the 2×6 QC-code is still 12. Then it can be seen that the 2×6 polynomial constraint matrix is equivalent to a 4×12 matrix by rearranging the rows so that the N-lift is split into a 2-(pre)lift and an $N/2$ final lift. Finally, we will take 3 out of the 4 polynomial rows and change their constraint nodes into generalized ones (achieving the desired optimal proportion).

Example 3. We revisit Example 2 and rearrange the constraint matrix

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & x^{54} & x^{66} & x^{71} & x^{55} & x^{69} \end{bmatrix}$$

to observe a 2-prelift, obtaining the following equivalent matrix (see [20] for details):

$$\left[\begin{array}{cc|cc|cc|cc|cc|cc} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ \hline 1 & 0 & x^{27} & 0 & x^{33} & 0 & 0 & x^{36} & 0 & x^{28} & 0 & x^{35} \\ 0 & 1 & 0 & x^{27} & 0 & x^{33} & x^{35} & 0 & x^{27} & 0 & x^{34} & 0 \end{array}\right].$$

By generalizing the first three constraints of this matrix using (4) and the factor $N = 90/2 = 45$, we obtain the following parity-check matrix H_{GC} for a GLDPC code:

$$\left[\begin{array}{cc|cc|cc|cc|cc|cc} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & x^{27} & 0 & 0 & 0 & 0 & x^{36} & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & x^{33} & 0 & 0 & 0 & 0 & x^{28} & 0 & 0 \\ 0 & 0 & x^{27} & 0 & x^{33} & 0 & 0 & 0 & 0 & 0 & 0 & x^{35} \\ \hline 0 & 1 & 0 & x^{27} & 0 & x^{33} & x^{35} & 0 & x^{27} & 0 & x^{34} & 0 \end{array}\right]$$

This code has minimum distance 39 and rate $R = 91/540 = 0.168$. The matrix in (6) of rank 88 is obtained through Theorem 1. Due to the rank loss, additional codewords need to be considered that increase the rank to the desired 91: $(f, f, f, f, 0, 0, 0, 0, f, f, f, f)$, $(0, 0, f, f, f, f, f, f, f, f, 0, 0)$ and $(g, 0, g, 0, g, 0, 0, 0, 0, 0, 0, 0)$, where f and g are the polynomials satisfying $(1+x)\cdot f = 1+x^{45}$ and $(1+x^3)\cdot g = 1+x^{45}$, in $\mathbf{F}_2[x]$. Since these are dense, substituting these with vectors obtained by creating linear combinations of these vectors with the rows of the matrix obtained with Theorem 1, will yield sparser vectors while maintaining rank 91.

We can also use different matrices for the generalized constraint nodes. For example, we generalized some of the nodes using

$$h_{GC,0} \triangleq \begin{bmatrix} M_1 & M_2 \end{bmatrix} \triangleq \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

and some with $h_{GC,1} \triangleq \begin{bmatrix} M_2 & M_1 \end{bmatrix}$, where M_i are 3×3 matrices. The following GLDPC code with overall parity-check matrix below in (8) has minimum distance 48 and rate $R = 90/540 = 1/6 = 0.166$ for $N = 78$, while, for $N = 38$, it has the same rate and minimum distance 40 (while the original H has girth 8). The matrix H has full rank, so there is a nonzero polynomial minor, and we can quickly find the polynomial generator matrix using the results from [15].

$$\left[\begin{array}{c|c}
x^{41}+x^{40}+x^{34}+x^{14}+x^7+x^2 & 0 \\
(x+1)(x^{42}+x^{11}+x^7+x^4)+x^{41}+x^{34}+x^{15}+1 & (x+1)(x^3+x^{27}+x^{34}+x^{38}+x^{41})+x^{25}+x^{32} \\
0 & x^{41}+x^{40}+x^{34}+x^{14}+x^7+x^2 \\
x^{42}+x^{41}+x^{34}+x^{23}+x^{16}+x^{15}+x^{12}+x^8+x^7+1 & (1+x+x^2)(x^3+x^{41})+(1+x)(x^{27}+x^{34})+x^{39}+x^{31} \\
x^{26}+x^{18}+x^{14}+x^8+x^6+1 & x^{34}+x^{33}+x^{28}+x^{27}+x+1 \\
(1+x+x^2)(x^{41}+x^6)+x^{34}+x^{24}+x^{23}+x^{15}+x^4+1 & (1+x+x^2)(x^{33}+x^3)+(1+x)(x^{41}+x^{27})+x^{25}+x^6 \\
x^{41}+x^{40}+x^{34}+x^{14}+x^7+x^2 & x^{41}+x^{40}+x^{34}+x^{14}+x^7+x^2 \\
x^{43}+x^{23}+x^{16}+x^{11}+x^5+x^4 & x^{43}+x^{38}+x^{32}+x^{31}+x^{25}+x^5 \\
x^{41}+x^{40}+x^{34}+x^{26}+x^{18}+x^8+x^7+x^6+x^2+1 & x^{34}+x^{33}+x^{28}+x^{27}+x+1 \\
x^{24}+x^{23}+x^{12}+x^{11}+x^6+x^5 & x^{39}+x^{38}+x^{33}+x^{32}+x^6+x^5 \\
x^{26}+x^{18}+x^{14}+x^8+x^6+1 & x^{41}+x^{40}+x^{33}+x^{28}+x^{27}+x^{14}+x^7+x^2+x+1 \\
x^{43}+x^{24}+x^{16}+x^{12}+x^6+x^4 & x^{43}+x^{39}+x^{33}+x^{31}+x^{25}+x^6
\end{array}\right] \tag{6}$$

$$\left[\begin{array}{c|c}
x^{20}+x^{15}+x^{12}+x^9+x^7+x & x^{28}+x^{23}+x^{22}+x^{21}+x^{20}+x^{16}+x^{14}+x^{11}+x^4+x^3 \\
x^{28}+x^{27}+x^{13}+x^{11}+x^6+x & x^{30}+x^{26}+x^{23}+x^{19}+x^{18}+x^{15}+x^{14}+x^{13}+x^{12}+x^{10}+x^9+x^6+x^4+x^3 \\
x^{31}+x^{26}+x^{20} & x^{33}+x^{28}+x^{26}+x^{25}+x^{22}+x^{21}+x^{20}+x^{15}+x^{14} \\
x^{28}+x^{27}+x^{22}+x^{17}+x^{13}+x^6+x & x^{26}+x^{25}+x^{24}+x^{23}+x^{19}+x^{18}+x^{17}+x^{15}+x^{14}+x^{10}+x^9+x^4+x^3 \\
x^{17}+x^{12}+x^6 & x^{33}+x^{28}+x^{26}+x^{25}+x^{22}+x^{21}+x^{20}+x^{15}+x^{14} \\
x^{28}+x^{27}+x^{22}+x^{17}+x^{13}+x^6+x & x^{31}+x^{25}+x^{24}+x^{23}+x^{20}+x^{19}+x^{18}+x^{15}+x^{14}+x^{12}+x^{10}+x^9+x^6+x^4+x^3 \\
x^{31}+x^{26}+x^{15}+x^{12}+x^9+x^7+x & x^{33}+x^{26}+x^{25}+x^{23}+x^{16}+x^{15}+x^{11}+x^4+x^3 \\
x^{22}+x^{17}+x^{11} & x^{30}+x^{25}+x^{24}+x^{17}+x^{13}+x^{12}+x^6 \\
x^{20}+x^{17}+x^{15}+x^9+x^7+x^6+x & x^{33}+x^{26}+x^{25}+x^{23}+x^{16}+x^{15}+x^{11}+x^4+x^3 \\
x^{22}+x^{17}+x^{11} & x^{31}+x^{30}+x^{26}+x^{25}+x^{24}+x^{20}+x^{13} \\
x^{31}+x^{26}+x^{20}+x^{17}+x^{12}+x^6 & 0 \\
0 & x^{31}+x^{26}+x^{20}+x^{17}+x^{12}+x^6
\end{array}\right] \tag{7}$$

$$\left[\begin{array}{cc|cc|cc|cc|cc|cc}
1&0&1&0&0&0&1&0&0&0&0&0 \\
1&0&0&0&1&0&0&0&1&0&0&0 \\
0&0&1&0&1&0&0&0&0&0&1&0 \\
\hline
0&1&0&1&0&0&0&1&0&0&0&0 \\
0&1&0&0&0&1&0&0&0&1&0&0 \\
0&0&0&1&0&1&0&0&0&0&0&1 \\
\hline
1&0&0&0&0&0&0&x^{36}&0&x^{28}&0&0 \\
0&0&x^{27}&0&0&0&0&x^{36}&0&0&0&x^{35} \\
0&0&0&0&x^{33}&0&0&0&0&x^{28}&0&x^{35} \\
\hline
0&1&0&x^{27}&0&x^{33}&x^{35}&0&x^{27}&0&x^{34}&0
\end{array}\right] \tag{8}$$

□

The polynomial generator matrix is given in (7). Note that it is transposed (the powers in (7) should also be changed to minus the values in the exponents, but we do not do this for readability). We also note that the matrix is relatively dense, which relates to the complexity of encoding. Recall however, that the minimum distance (i.e., row weight) must be 40.

Remark 4. In summary, the construction takes the first constraint row $\begin{bmatrix}1 & 1 & \dots & 1\end{bmatrix}$ and replaces it by $\begin{bmatrix}M_1 & M_2\end{bmatrix}$. We then take half of the rows and columns of the second constraint row $\begin{bmatrix}1 & x^{i_2} & \dots & x^{i_{n_v}}\end{bmatrix}$ and replace it by $\begin{bmatrix}M_2 & M_1\end{bmatrix}$, while leaving the rest as simple check nodes. It is important to do this by rearranging rows and columns to display the matrix as a double lifting. This method can be used for the other protographs and it can also be used with larger than double lifting. We will demonstrate this in the next subsections. □

B. Case $n_v = 7$

We consider that the component codes will be the Hamming code with parity check matrix

$$h_{GC} \triangleq \begin{bmatrix}1&1&1&0&1&0&0\\1&1&0&1&0&1&0\\1&0&1&1&0&0&1\end{bmatrix} \tag{9}$$

(or any other parity-check matrix of a Hamming $(7,4,3)$ code). For example, let $N = 68$ and set the exponents $[i_2, i_3, i_4, i_5, i_6, i_7] = [61, 49, 44, 1, 46, 14]$, such that H has girth 12. By generalizing one constraint, we get the parity-check matrix

$$H_{GC} \triangleq \begin{bmatrix}1&x^{61}&x^{49}&x^{44}&x&x^{46}&x^{14}\\1&1&1&0&1&0&0\\1&1&0&1&0&1&0\\1&0&1&1&0&0&1\end{bmatrix}.$$

The rate is $R = 204/476 = 0.428$ and the minimum distance is 16. Since the matrix H has a nonzero minor, we apply the method described in Theorem 1 to find the generator matrix.

We can improve the code parameters by applying the same idea as before, rearranging rows and columns in

$$\begin{bmatrix}1&1&1&1&1&1&1\\1&x^{61}&x^{49}&x^{44}&x&x^{46}&x^{14}\end{bmatrix}$$

to display a sequence of two liftings (since N is even), and obtain

$$\left[\begin{array}{cc|cc|cc|cc|cc|cc|cc}
1&0&1&0&1&0&1&0&1&0&1&0&1&0 \\
0&1&0&1&0&1&0&1&0&1&0&1&0&1 \\
\hline
1&0&0&x^{31}&0&x^{25}&x^{22}&0&0&x&x^{23}&0&x^7&0 \\
0&1&x^{30}&0&x^{24}&0&0&x^{22}&1&0&0&x^{23}&0&x^7
\end{array}\right].$$

We generalize some of its nodes with $h_{GC,0}$ and some with $h_{GC,1}$, where

$$h_{GC,0} \triangleq \begin{bmatrix}M_1 & M_2\end{bmatrix} \triangleq \left[\begin{array}{ccc|ccc}1&1&1&0&1&0&0\\1&1&0&1&0&1&0\\1&0&1&1&0&0&1\end{array}\right],$$

$$h_{GC,1} \triangleq \begin{bmatrix}M_2 & M_1\end{bmatrix} = \left[\begin{array}{ccc|ccc}1&0&0&1&1&1&0\\0&1&0&1&1&0&1\\0&0&1&1&0&1&1\end{array}\right].$$

$$\left[\begin{array}{cccccccccccccccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & x^6 & x^{18} & x^{23} & x^7 & x^{21} & x & x^3 & x^9 & x^{27} & x^{19} & x^{28} & x^{16} & x^{17} & x^{20} & \end{array}\right] \sim \tag{10}$$

$$\left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc|cc|cc|cc|cc|cc|cc} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ \hline 1 & 0 & x^3 & 0 & x^9 & 0 & 0 & x^{12} & 0 & x^4 & 0 & x^{11} & 0 & x & 0 & x^2 & 0 & x^5 & 0 & x^{14} & 0 & x^{10} & x^{14} & 0 & x^8 & 0 & 0 & x^9 & x^{10} & 0 \\ 0 & 1 & 0 & x^3 & 0 & x^9 & x^{11} & 0 & x^3 & 0 & x^{10} & 0 & 1 & 0 & x & 0 & x^4 & 0 & x^{13} & 0 & x^9 & 0 & 0 & x^{14} & 0 & x^8 & x^8 & 0 & 0 & x^{10} \end{array}\right] \tag{11}$$

$$\left[\begin{array}{cc|cc|cc|cc|cc|cc|cc|cc|cc|cc|cc|cc|cc|cc|cc} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & x^9 & 0 & 0 & 0 & 0 & x^4 & 0 & 0 & 0 & x & 0 & x^2 & 0 & 0 & 0 & x^{14} & 0 & 0 & x^{14} & 0 & 0 & 0 & 0 & x^9 & 0 & 0 \\ 0 & 0 & x^3 & 0 & x^9 & 0 & 0 & 0 & 0 & 0 & 0 & x^{11} & 0 & x & 0 & 0 & 0 & x^5 & 0 & x^{14} & 0 & 0 & 0 & 0 & x^8 & 0 & 0 & x^9 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & x^{12} & 0 & x^4 & 0 & x^{11} & 0 & x & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x^{10} & x^{14} & 0 & x^8 & 0 & 0 & x^9 & 0 & 0 \\ 1 & 0 & x^3 & 0 & x^9 & 0 & 0 & x^{12} & 0 & x^4 & 0 & x^{11} & 0 & x & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x^{10} & 0 \\ \hline 0 & 1 & 0 & x^3 & 0 & x^9 & x^{11} & 0 & x^3 & 0 & x^{10} & 0 & 1 & 0 & x & 0 & x^4 & 0 & x^{13} & 0 & x^9 & 0 & 0 & x^{14} & 0 & x^8 & x^8 & 0 & 0 & x^{10} \end{array}\right] \tag{12}$$

For example, the parity-check matrix

$$\left[\begin{array}{cc|cc|cc|cc|cc|cc|cc} 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 & 0 & x^{22} & 0 & 0 & x & x^{23} & 0 & 0 & 0 \\ 0 & 0 & 0 & x^{31} & 0 & 0 & x^{22} & 0 & 0 & x & 0 & 0 & x^7 & 0 \\ 0 & 0 & 0 & 0 & 0 & x^{25} & x^{22} & 0 & 0 & 0 & x^{23} & 0 & x^7 & 0 \\ \hline 0 & 1 & x^{30} & 0 & x^{24} & 0 & 0 & x^{22} & 1 & 0 & 0 & x^{23} & 0 & x^7 \end{array}\right]$$

gives a GLDPC code, with $N = 68/2 = 34$, minimum distance between 26 and 31, and rate $R = 136/476 = 2/7 = 0.28$. The matrix H has full rank, so there is a nonzero polynomial minor, so we can find quickly a polynomial generator matrix using Theorem 1.

We can also change the order of the second matrix and possibly create better codes. For example, with one transposition of two columns, we obtain a code with the same rate but possibly better distance between 27 and 33. Each of these matrices have full rank, so there is a nonzero polynomial minor that allows us to find a polynomial generator matrix. t

C. Case $n_v = 15$

For $n_v = 15$, we suppose that the component codes are again Hamming codes, with $h_{GC,0} \triangleq \begin{bmatrix} M_1 & M_2 \end{bmatrix} =$

$$\left[\begin{array}{ccccccc|ccccccc} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right]$$

and $h_{GC,1} \triangleq \begin{bmatrix} M_2 & M_1 \end{bmatrix}$ following [7].

We take $N = 31$ and the exponents $[i_2, i_3, \ldots, i_{15}] = [6, 18, 23, 7, 21, 1, 3, 9, 27, 19, 28, 16, 17, 20]$, for which H has girth 8 to obtain H_{GC} given by

$$\left[\begin{array}{ccccccc|ccccccc} 1 & x^6 & x^{18} & x^{23} & x^7 & x^{21} & x & x^3 & x^9 & x^{27} & x^{19} & x^{28} & x^{16} & x^{17} & x^{20} \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right].$$

We apply the method described earlier to find a generator matrix, since the matrix H has a nonzero polynomial minor.

The code has rate $R = 310/465$ and minimum distance 6 (only an increase of 2 over the original), with a decrease of rank (404 decreased to 310). The reason for the improvement not being significant is that N is too small. We can instead take

$$\left[\begin{array}{cccccccccccccccc} 1 & 0 & x^{18} & 0 & x^7 & 0 & x & x^3 & 0 & x^{27} & 0 & x^{28} & 0 & x^{17} & 0 & \\ 0 & x^6 & x^{18} & 0 & 0 & x^{21} & x & 0 & x^9 & x^{27} & 0 & 0 & x^{16} & x^{17} & 0 & \\ 0 & 0 & 0 & x^{23} & x^7 & x^{21} & x & 0 & 0 & 0 & x^{19} & x^{28} & x^{16} & x^{17} & 0 & \\ 1 & x^6 & x^{18} & x^{23} & x^7 & x^{21} & x & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x^{20} & \\ \hline 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & \end{array}\right],$$

like in [7], or try any other rearrangement of the matrix H. In this case, the rank is not full and the rate is $R = 218/465$, but the minimum distance is $16 \leq d \leq 29$. Note that the original matrix has rank 8 (not 12).

Alternatively, we can write the parity-check matrix, for an even N, as a 2-lifting and a $N/2$ lifting, and apply the method from the other cases. We rearrange rows and column to display a sequence of two liftings (for some even N) to obtain the code in (10). We now generalize the first $2N$ nodes with $h_{GC,0}$ and the next N nodes with $h_{GC,1}$, leaving the last N nodes as simple parity-checks. The final parity-check matrix is in (12).

IV. Concluding remarks

This paper shows how to obtain a generator matrix for a QC-GLDPC code using minors of the polynomial parity-check matrix. The approach can be applied to fully generalized matrices, or partially generalized (with mixed constraint nodes) in order to find better performance/rate trade-offs. The resulting matrices can be presented in several forms that may facilitate efficient encoder implementation as well as minimum distance analysis. We also demonstrated that the code parameters can be improved by applying a double graph-lifting procedure that does not affect the ability to obtain a polynomial generator matrix.

Acknowledgements

This work was partially supported by the NSF under Grant Nos. CCF-2145917 and CNS-2148358, and the fellowships from GFSD and Kinesis-Fernández Richards.

REFERENCES

[1] M. Lentmaier and K. Sh. Zigangirov, "On generalized low-density parity-check codes based on Hamming component codes," *IEEE Communications Letters*, vol. 8, no. 8, pp. 248–250, Aug. 1999.

[2] J. J. Boutros, O. Pothier, and G. Zémor, "Generalized low density Tanner codes," in *Proc. IEEE International Conference on Communications*, Vancouver, Canada, June 1999.

[3] M. Lentmaier, G. Liva, E. Paolini, and G. Fettweis, "From product codes to structured generalized LDPC codes," in *Proc. 5th International ICST Conference on Communications and Networking*, Beijing, China, Aug. 2010.

[4] M. Lentmaier and G. Fettweis, "On the thresholds of generalized LDPC convolutional codes based on protographs," in *Proc. IEEE International Symposium on Information Theory*, Austin, TX, July 2010.

[5] Y. Liu, P. M. Olmos, and D. G. M. Mitchell, "Generalized LDPC codes for ultra reliable low latency communication in 5G and beyond," *IEEE Access*, vol. 6, no. 1, pp. 72 002–72 014, Dec. 2018.

[6] I. P. Mulholland, E. Paolini, and M. F. Flanagan, "Design of LDPC code ensembles with fast convergence properties," in *Proc. IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, May 2015, pp. 53–57.

[7] G. Liva, W. E. Ryan, and M. Chiani, "Quasi-cyclic generalized LDPC codes with low error floors," *IEEE Transactions on Communications*, vol. 56, no. 1, pp. 49–57, Jan. 2008.

[8] D. G. M. Mitchell, P. M. Olmos, M. Lentmaier, and D. J. Costello, Jr., "Spatially coupled generalized LDPC codes: Asymptotic analysis and finite length scaling," *IEEE Transactions on Information Theory*, vol. 67, no. 6, pp. 3708–3723, Jun. 2021.

[9] S. Habib and D. G. M. Mitchell, "Reinforcement learning for sequential decoding of generalized LDPC codes," in *Proc. International Symposium on Topics in Coding (ISTC)*. Brest, France: IEEE, 2023, pp. 1–5.

[10] M. Sybis, K. Wesolowski, K. Jayasinghe, V. Venkatasubramanian, and V. Vukadinovic, "Channel coding for ultra-reliable low-latency communication in 5G systems," in *Proc. IEEE Vehicular Technology Confrence*, Montreal, QC, Canada, 2016, pp. 1–5.

[11] Z. Wang and Z. Cui, "A memory efficient partially parallel decoder architecture for quasi-cyclic LDPC codes," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 4, pp. 483–488, Apr. 2007.

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

[13] L. Zhang, Q. Huang, S. Lin, K. Abdel-Ghaffar, and I. F. Blake, "Quasi-cyclic LDPC codes: An algebraic construction, rank analysis, and codes on latin squares," *IEEE Transactions on Communications*, vol. 58, no. 11, pp. 3126–3139, Nov. 2010.

[14] P.-C. Yang, C.-H. Wang, and C.-C. Chao, "Rank analysis of parity-check matrices for quasi-cyclic LDPC codes," in *Proc. IEEE International Symposium on Information Theory*. Vail, CO: IEEE, 2018, pp. 491–495.

[15] R. Smarandache, A. Gómez-Fonseca, and D. G. M. Mitchell, "Using minors to construct generator matrices for quasi-cyclic LDPC codes," in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, 2022, pp. 548–553.

[16] R. M. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533–547, Sept. 1981.

[17] J. Thorpe, "Low-density parity-check (LDPC) codes constructed from protographs," Jet Propulsion Laboratory, Pasadena, CA, INP Progress Report 42-154, Aug. 2003.

[18] D. Divsalar, S. Dolinar, C. Jones, and K. Andrews, "Capacity-approaching protograph codes," *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 6, pp. 876–888, Aug. 2009.

[19] K. Lally and P. Fitzpatrick, "Algebraic structure of quasicyclic codes," in *Proc. IEEE International Symposium on Information Theory*, Sorrento, Italy, 2000.

[20] R. Smarandache and D. G. M. Mitchell, "A unifying framework to construct QC-LDPC tanner graphs of desired girth," *IEEE Transactions on Information Theory*, vol. 68, no. 9, pp. 5802–5822, 2022.

[21] D. G. M. Mitchell, R. Smarandache, and D. J. Costello, Jr., "Quasi-cyclic LDPC codes based on pre-lifted protographs," *IEEE Transactions on Information Theory*, vol. 60, no. 10, pp. 5856–5874, Oct. 2014.