Circular-shift Linear Network Coding

Hanqi Tang[†], Qifu Tyler Sun[†], Zongpeng Li[‡*], Xiaolong Yang[†], and Keping Long[†]

[†]University of Science and Technology Beijing, P. R. China

[‡]University of Calgary, Canada [*] Wuhan University, P. R. China

arXiv:1707.02163v1 [cs.IT] 7 Jul 2017

Abstract

We study a class of linear network coding (LNC) schemes, called *circular-shift* LNC, whose encoding operations at intermediate nodes consist of only circular-shifts and bit-wise addition (XOR). Departing from existing literature, we systematically formulate circular-shift LNC as a special type of vector LNC, where the local encoding kernels of an L-dimensional circular-shift linear code of degree δ are summation of at most δ cyclic-permutation matrices of size L. Under this framework, an intrinsic connection between scalar LNC and circular-shift LNC is established. In particular, on a general network, for some block lengths L, every scalar linear solution over GF(2^{L-1}) can induce an $(L-1, L)$-fractional circular-shift linear solution of degree $(L-1)/2$. Specific to multicast networks, an $(L-1, L)$-fractional circular-shift linear solution of arbitrary degree δ can be efficiently constructed. With different δ, the constructed solution has an interesting encoding-decoding complexity tradeoff, and when $\delta = (L-1)/2$, it requires fewer binary operations for both encoding and decoding processes compared with scalar LNC. While the constructed solution has one-bit redundancy per edge transmission, we show that this is inevitable, and that circular-shift LNC is insufficient to achieve the exact capacity of some multicast networks. Last, both theoretical and numerical analysis imply that with increasing L, a randomly constructed circular-shift linear code has comparable linear solvability behavior to a randomly constructed permutation-based linear code, but has much shorter overheads for random coding.

I. Introduction

Assume that every edge in a network transmits a binary sequence of length L. Different linear network coding (LNC) schemes manipulate the binary sequences by different approaches. With

[*] Q. T. Sun (Email: qfsun@ustb.edu.cn) is the corresponding author. This work was partially presented at the 2017 IEEE International Symposium on Information Theory.

conventional scalar LNC (see, e.g., [1][2]) and vector LNC (see, e.g., [3][4]), the binary sequence carried at every edge is modeled, respectively, as an element of GF(2^L) and an L-dimensional vector over GF(2). The coding operations performed at every intermediate node by scalar LNC and by vector LNC are linear functions over GF(2^L) and over the ring of $L \times L$ binary matrices, respectively.

There have been continuous attempts to design LNC schemes with low implementation complexities. A straightforward way is to reduce the block length L. It is well known that when 2^L is no smaller than the number of receivers, a scalar linear solution over GF(2^L) can be efficiently constructed on a multicast network by algorithms in [5] and [6]. Recent literature has witnessed a few interesting multicast networks that have an L-dimensional vector linear solutin over GF(2) but do not have a scalar linear solution over GF($2^{L'}$) for any $L' \leq L$ [7], [8]. This verifies that compared with scalar LNC, vector LNC may yield solutions with lower implementation complexities.

Another approach to reduce the encoding complexity of LNC is to carefully design the coding operations performed at intermediate nodes. A special type of vector LNC based on permutation operations is studied in [9], from a random coding approach. In permutation-based vector LNC, at an intermediate node, every incoming binary sequence is first permuted, and then an outgoing binary sequence is formed by bit-wise addition of the permutated incoming binary sequences. Equivalently, local encoding kernels at intermediate nodes are chosen from $L \times L$ binary permutation matrices, rather than arbitrary $L \times L$ binary matrices. Though permutation can be more efficiently implemented than general matrix multiplication on a binary sequence, its computational complexity may not be low enough for real-world implementation, when the sequence length L is long, as required in random coding.

Towards further reducing the encoding and decoding complexity of LNC, we study another class of LNC schemes whose encoding operations are restricted to merely *circular-shifts* and bit-wise addition. Obviously, on a binary sequence, circular-shift operations have lower computational complexity than permutations, and are amenable to implementation through atomic hardware operations. Though prior to this work, similar ideas of adopting circular-shift and bit-wise addition operations for encoding have been considered in [10], [11] and [12], the work in [10] only focuses on Combination Networks whereas the approach of [11] and [12] is from the perspective of cyclic convolutional coding. Due to the lack of a systematic model, how to efficiently construct a circular-shift linear solution on a general multicast network is still unknown.

In this paper, we algebraically formulate circular-shift LNC as a special type of vector LNC. Specifically, for an L-dimensional circular-shift linear code of degree δ formulated in this work, the local encoding kernels at intermediate nodes correspond to summation of at most δ cyclic-permutation matrices of size L. This framework facilitates us to establish an intrinsic connection between scalar LNC and circular-shift LNC, so that on a general network and for a prime L with primitive root 2, every scalar linear solution over GF(2^{L-1}) can induce an $(L-1, L)$-fractional circular-shift linear solution of degree no larger than $\frac{L-1}{2}$. Consequently, on a multicast network, both the local encoding kernels and decoding matrices of an $(L-1, L)$-fractional circular-shift linear solution of arbitrary degree δ can be efficiently constructed. When $\delta = \frac{L-1}{2}$, the constructed solution requires fewer binary operations for both encoding and decoding processes compared with scalar linear solutions over GF(2^{L-1}). Furthermore, when δ decreases from $\frac{L-1}{2}$ to 1, there is an interesting tradeoff between decreasing encoding complexity and increasing decoding complexity, making the code design more flexible. Though the constructed $(L-1, L)$-fractional solution has one-bit redundancy per edge transmission, this is necessary because we also show that circular-shift LNC is insufficient to achieve the exact capacity of certain multicast networks.

In addition, we study circular-shift LNC of degree 1 from a random coding approach. We derive a lower bound on the successful probability of a randomly generated fractional circular-shift linear code to be a linear solution, which is essentially the same as the one in [9] for permutation-based LNC. For moderate block lengths, numerical results demonstrate comparable successful probability of randomly generating a fractional circular-shift linear solution to the one of randomly generating a permutation-based linear solution of the same rate. These findings are interesting because for block length L, circular-shift LNC provides $L+1$ local encoding kernel candidates, much less than $L!$ in permutation-based LNC. Last, we show that circular-shift LNC has the additional advantage on shorter overheads for random coding.

The rest of the paper is organized as follows. Section II briefly reviews the basic concepts of scalar, vector and fractional LNC. Section III formulates circular-shift LNC from the perspective of vector LNC and establishes an intrinsic connection between scalar LNC and circular-shift LNC on general networks. Section IV discusses efficient construction of circular-shift linear solutions on multicast networks. Section V analyzes circular-shift LNC by the random coding approach. Section VII concludes the paper.

II. Preliminaries

A general multi-source (acyclic) network, which is modeled as a finite directed acyclic multigraph, with a set S of source nodes and a set T of receivers. For a node v in the network, denote by $In(v)$ and $Out(v)$, respectively, the set of its incoming and outgoing edges. Similarly, denote by $In(N)$ and $Out(N)$ the set of incoming and outgoing edges from a set N of nodes respectively. Every edge has unit capacity to transmit a data unit per channel use. Write $|Out(S)| = \omega$. Without loss of generality (WLOG), assume that every source $s \in S$ generates $|Out(s)|$ source data units, and there are total ω source data units generated by S to be propagated along the network. Assume an arbitrary order on $S = \{s_1, \ldots, s_{|S|}\}$ and a topological order on the set E of edges led by ones in $Out(s_j)$, $1 \leq j \leq |S|$, sequentially. For every receiver $t \in T$, based on the data units received from edges in $In(t)$, its goal is to recover the $\omega_t = |Out(S_t)|$ data units generated from a particular set $S_t \subseteq S$ of sources. WLOG, assume that for every source, its in-degree is zero and there is not any edge leading from it to a receiver. When there is a unique source node s and all receivers need recover the $|Out(s)|$ source data units generated at s, the network is called a *multicast network*.

Notations. Let $\otimes$ denote the Kronecker product. For a positive integer j, let $\mathbf{U}_e^j$ denote such an $\omega j \times j$ matrix that the column-wise juxtaposition $[\mathbf{U}_e^j]_{e \in Out(S)}$ forms an $\omega j \times \omega j$ identity matrix $\mathbf{I}_{\omega j}$, *i.e.*, $\mathbf{U}_e^j = \mathbf{U}_e^1 \otimes \mathbf{I}_j$. Unless otherwise specified, all juxtaposition of matrices or vectors throughout this paper refers to column-wise juxtaposition.

In this paper, we model circular-shift LNC (over GF(2)) as a special type of vector LNC, in which the data unit transmitted along every edge e is an L-dimensional *row vector* $\mathbf{m}_e$ of binary data symbols. An *L-dimensional vector linear code over* GF(2) (See, e.g., [7]), is an assignment of a local encoding kernel $\mathbf{K}_{d,e}$, which is an $L \times L$ matrix over GF(2), to every pair (d, e) of edges such that $\mathbf{K}_{d,e}$ is the zero matrix $\mathbf{0}$ when (d, e) is not an adjacent pair. Then, for every edge e emanating from a non-source node v, the data unit vector of binary data symbols transmitted on e is $\mathbf{m}_e = \sum_{d \in In(v)} \mathbf{m}_d \mathbf{K}_{d,e}$. WLOG, for every $s \in S$, assume the data units $\mathbf{m}_e$, $e \in Out(s)$, just constitute the $|Out(s)|$ source data units generated by s. Every vector linear code uniquely determines a global encoding kernel $\mathbf{F}_e$, which is an $\omega L \times L$ matrix over GF(2), for every edge e such that

- $[\mathbf{F}_e]_{e \in Out(S)} = [\mathbf{U}_e^L]_{e \in Out(S)} = \mathbf{I}_{\omega L}$;

- For every outgoing edge e from a non-source node v, $\mathbf{F}_e = \sum_{d \in In(v)} \mathbf{F}_d \mathbf{K}_{d,e}$.

Correspondingly, the data unit vector transmitted along every edge can also be represented as

$$\mathbf{m}_e = [\mathbf{m}_\mathbf{d}]_{d \in Out(S)} \mathbf{F}_e.$$

A vector linear code is a *vector linear solution* if for every receiver $t \in T$, there is an $|In(t)|L \times \omega_t L$ decoding matrix $\mathbf{D}_t$ over GF(2) such that

$$[\mathbf{F}_e]_{e \in In(t)} \mathbf{D}_t = [\mathbf{U}_e^L]_{e \in Out(S_t)}$$

Based on $\mathbf{D}_t$, the data units generated at sources in S_t can be recovered by receiver t via

$$\begin{aligned}[\mathbf{m}_e]_{e \in In(t)} \mathbf{D}_t &= \left([\mathbf{m}_\mathbf{d}]_{d \in Out(S)} [\mathbf{F}_e]_{e \in In(t)}\right) \mathbf{D}_t \\ &= [\mathbf{m}_\mathbf{d}]_{d \in Out(S)} \left([\mathbf{F}_e]_{e \in In(t)} \mathbf{D}_t\right) \\ &= [\mathbf{m}_\mathbf{d}]_{d \in Out(S)} [\mathbf{U}_e^L]_{e \in Out(S_t)} \\ &= [\mathbf{m}_\mathbf{d}]_{d \in Out(S_t)}.\end{aligned}$$

Fractional LNC is a generalization of vector LNC (See, e.g., [13]). Same as in an L-dimensional vector linear code over GF(2), in an (L', L)-fractional linear code over GF(2), the data unit $\mathbf{m}_e$ transmitted on every edge e is an L-dimensional row vector over GF(2), and the local encoding kernels $\mathbf{K}_{d,e}$ are $L \times L$ matrices over GF(2). The difference is that for an (L', L)-fractional linear code, the $|Out(s)|$ data units generated at every source $s \in S$ are L'-dimensional row vectors over GF(2). To be a little abuse of notations, denote the $|Out(s)|$ L'-dimensional row vectors generated at s by $\mathbf{m}'_e$, $e \in Out(s)$. Each of the L binary data symbols in data unit $\mathbf{m}_e$ transmitted on $e \in Out(s)$, is a GF(2)-linear combination of the ones in $\mathbf{m}'_e$, $e \in Out(s)$, *i.e.*,

$$[\mathbf{m}_e]_{e \in Out(s)} = [\mathbf{m}'_e]_{e \in Out(s)} \mathbf{G}_s$$

for some $|Out(s)|L' \times |Out(s)|L$ matrix $\mathbf{G}_s$ over GF(2). In total, the data units $\mathbf{m}_e$ transmitted on $e \in Out(S)$ can be expressed as

$$[\mathbf{m}_e]_{e \in Out(S)} = [\mathbf{m}'_e]_{e \in Out(S)} \mathbf{G}_S,$$

where $\mathbf{G}_S$ denotes the $\omega L' \times \omega L$ matrix

$$\mathbf{G}_S = \begin{bmatrix} \mathbf{G}_{s_1} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{G}_{s_2} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{G}_{s_{|S|}} \end{bmatrix}$$

which consists of $|S| \times |S|$ blocks with the $(j,j)^{th}$ "diagonal" block, $1 \leq j \leq |S|$, being the $\omega_{s_j}L' \times \omega_{s_j}L$ matrix $\mathbf{G}_{s_j}$.

An (L', L)-fractional linear code over GF(2) qualifies as an $\omega L' \times \omega L$ *fractional linear solution* if for each receiver t, there is an $|In(t)|L \times \omega_t L'$ matrix $\mathbf{D}_t$ over GF(2) such that

$$\mathbf{G}_S[\mathbf{F}_e]_{e \in In(t)}\mathbf{D}_t = [\mathbf{U}_e^{L'}]_{e \in Out(S_t)}.$$

Based on the decoding matrix $\mathbf{D}_t$, the data units $\mathbf{m}'_e$, $e \in Out(S_t)$ generated by sources in S_t can be recovered at t via

$$\begin{aligned}
[\mathbf{m}_e]_{e \in In(t)}\mathbf{D}_t &= \left([\mathbf{m}_e]_{e \in Out(S)}[\mathbf{F}_e]_{e \in In(t)}\right)\mathbf{D}_t \\
&= \left([\mathbf{m}'_e]_{e \in Out(S)}\mathbf{G}_S[\mathbf{F}_e]_{e \in In(t)}\right)\mathbf{D}_t \\
&= [\mathbf{m}'_e]_{e \in Out(S)}[\mathbf{U}_e^{L'}]_{e \in Out(S_t)} \\
&= [\mathbf{m}'_e]_{e \in Out(S_t)}.
\end{aligned}$$

Conventional scalar linear codes over GF(2) and L-dimensional vector linear codes over GF(2) can be respectively regarded as $(1,1)$-fractional and (L,L)-fractional linear codes over GF(2).

III. Algebraic Formulation of Circular-Shift LNC

Similar ideas of LNC based on circular-shifts and bit-wise addition are considered in [11] and [12]. Their approach stems from the cyclic codes in coding theory, and relates the binary sequences transmitted on edges and the local encoding kernels to polynomials. We instead model circular-shift LNC as a subclass of vector LNC. The advantage of such formulation is that more transparent matrix manipulations can be conducted on the local encoding kernels. An inherent connection between circular-shift LNC and scalar LNC can be readily unveiled on a general network.

For a positive integer L, denote by $\mathbf{C}_L$ the following $L \times L$ cyclic permutation matrix

$$\mathbf{C}_L = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \ddots & 0 \\ 0 & \ddots & \ddots & \ddots & 0 \\ 0 & \ddots & \ddots & 0 & 1 \\ 1 & 0 & \dots & 0 & 0 \end{bmatrix},$$

and by $\mathbf{I}_L$ the identity matrix of size L. Both $\mathbf{C}_L$ and $\mathbf{I}_L$ are defined over GF(2). In addition, for $1 \leq \delta \leq L$, let $\mathcal{C}_\delta$ denote the following set of matrices:

$$\mathcal{C}_\delta = \left\{ \sum_{j=0}^{L-1} a_j \mathbf{C}_L^j : a_j \in \{0,1\}, \sum_{j=0}^{L-1} a_j \leq \delta \right\}, \tag{1}$$

that is, $\mathcal{C}_\delta$ contains the matrices that are summation of at most δ cyclic permutation matrices of of size L. We model circular shift LNC as vector LNC with local encoding kernels $\mathbf{K}_{d,e}$ chosen from $\mathcal{C}_\delta$. The operation $\mathbf{m}_d\mathbf{K}_{d,e}$ conducts at most δ circular-shifts on $\mathbf{m}_d$ and then computes bit-wise addition among at most δ circular-shifted L-dimensional row vectors.

Definition 1. On a general network, an (L', L) *circular-shift linear code of degree* δ refers to an (L', L)-fractional linear code over GF(2) with all local encoding kernels chosen from $\mathcal{C}_\delta$. It is called an (L', L) *circular-shift linear solution of degree* δ if it is an (L', L)-fractional linear solution.

It is interesting to note that the set $\mathcal{C}_L$ forms a *commutative* subring of the (non-commutative) ring $M_L(\mathrm{GF}(2))$ of $L \times L$ binary matrices. Thus, circular-shift LNC conforms to the assumption in the algebraic structure of vector LNC that local encoding kernels are selected from commutative matrices [4]. In addition, under the general model in [14], an L-dimensional circular-shift linear code of degree L can be regarded as a linear code over the $\mathcal{C}_L$-module $\mathrm{GF}(2)^L$.

It is also worthwhile to note that rotation-and-add coding studied in [11] can be regarded as a special type of circular-shift LNC of degree 1, where matrix $\mathbf{0}$ is not a candidate for local encoding kernels.

The following diagonalization manipulation of the cyclic permutation matrix $\mathbf{C}_L$ over a larger field is one of the benefits to study circular-shift LNC from the perspective of vector LNC instead of cyclic convolutional coding.

Lemma 2. Let L be an odd integer and α be a primitive L^{th} root of unity. Denote by $\mathbf{V}_L$ the $L \times L$ Vandermonde matrix generated by $1, \alpha, \ldots, \alpha^{L-1}$ over GF(2)(α), the minimal field containing GF(2) and α:

$$\mathbf{V}_L = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ 1 & \alpha & \ldots & \alpha^{L-1} \\ \vdots & \vdots & \ldots & \vdots \\ 1 & \alpha^{L-1} & \ldots & \alpha^{(L-1)(L-1)} \end{bmatrix}, \tag{2}$$

and by $\mathbf{\Lambda}_\alpha$ the $L \times L$ diagonal matrix with diagonal entries equal to $1, \alpha, \ldots, \alpha^{L-1}$, *i.e.*,

$$\mathbf{\Lambda}_\alpha = \begin{bmatrix} 1 & 0 & \ldots & 0 \\ 0 & \alpha & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \ldots & 0 & \alpha^{L-1} \end{bmatrix}.$$

The inverse of $\mathbf{V}_L$ is

$$\mathbf{V}_L^{-1} = \begin{bmatrix} 1 & 1 & \ldots & 1 \\ 1 & \alpha^{-1} & \ldots & \alpha^{-(L-1)} \\ \vdots & \vdots & \ldots & \vdots \\ 1 & \alpha^{-(L-1)} & \ldots & \alpha^{-(L-1)(L-1)} \end{bmatrix}, \tag{3}$$

$$\text{and} \quad \mathbf{C}_L^i = \mathbf{V}_L \cdot \mathbf{\Lambda}_\alpha^i \cdot \mathbf{V}_L^{-1} \qquad \forall i \geq 0 \tag{4}$$

Proof. First note that the i^{th} row in $\mathbf{V}_L$ times the j^{th} column in $\mathbf{V}_L^{-1}$ $(0 \leq i, j \leq L-1)$ equals to $\sum_{i'=0}^{L-1} \alpha^{i'(i-j)}$. Since α is a primitive L^{th} root of unity, $\alpha^{i'}$ is a root of $x^L - 1$ and not equal to 1 for all $1 \leq i' \leq L-1$. In addition, since $x^L - 1 = (x-1)(x^{L-1} + \ldots + 1)$, $\sum_{i'=0}^{L-1} \alpha^{i'(i-j)} = 0$ when $i \neq j$. Furthermore, when $i = j$, $\sum_{i'=0}^{L-1} \alpha^{i'(i-j)} = 1$ for summation of 1 by (odd) L times is still equal to 1 over GF(2). In sum, $\mathbf{V}_L \mathbf{V}_L^{-1} = \mathbf{I}_L$.

Next, note that

$$\mathbf{V}_L \cdot \mathbf{\Lambda}_\alpha = \begin{bmatrix} 1 & \alpha & \ldots & \alpha^{L-1} \\ \vdots & \vdots & \ldots & \vdots \\ 1 & \alpha^{L-1} & \ldots & \alpha^{(L-1)(L-1)} \\ 1 & 1 & \ldots & 1 \end{bmatrix} = \mathbf{C}_L \cdot \mathbf{V}_L.$$

$$\text{As a result,} \quad \mathbf{V}_L \cdot \mathbf{\Lambda}_\alpha \cdot \mathbf{V}_L^{-1} = \mathbf{C}_L \cdot \mathbf{V}_L \cdot \mathbf{V}_L^{-1} = \mathbf{C}_L,$$

and thus (4) holds. □

In the remaining part of this paper, we focus on the case that L that is a prime with primitive root 2 (*i.e.*, the multiplicative order of 2 modulo L is $L-1$). Such an assumption enables us to establish an inherent connection between every scalar linear solution over GF(2^{L-1}) and an $(L-1, L)$ circular-shift linear solution over an general network.

Notations. Let α be a primitive L^{th} root of unity. If an $m \times n$ matrix $\mathbf{M}$ is defined over the polynomial ring GF(2)$[x]$, then this matrix will be denoted by $\mathbf{M}(x)$. Concomitantly, $\mathbf{M}(\alpha^i)$, $i \geq 0$, represents an $m \times n$ matrix obtained from $\mathbf{M}(x)$ via setting x to α^i. Similarly, $\mathbf{M}(\mathbf{C}_L^i)$

represents the $mL \times nL$ matrix over GF(2) obtained from $\mathbf{M}(x)$ via replacing zero entry by an $L \times L$ zero matrix and setting x to be matrix $\mathbf{C}_L^i$.

Lemma 3. The followings hold for the considered L and α:

a) $f(x) = x^{L-1} + \ldots + x + 1$ is an irreducible polynomial over GF(2) and it has $L-1$ roots: $\alpha, \ldots, \alpha^{L-1}$, which belong to GF($2^{L-1}$).

b) Corresponding to every element $k \in \mathrm{GF}(2^{L-1})$, there is a unique polynomial over GF(2)

$$(*) \qquad g(x) := a_{L-1}x^{L-1} + \ldots + a_1x^1 + a_0,$$

such that $k = g(\alpha)$ and at most $\frac{L-1}{2}$ coefficients a_j, $0 \le j \le L-1$, are nonzero.

c) For two arbitrary polynomials $g_1(x)$ and $g_2(x)$ over GF(2), if $g_1(\alpha^{k_1}) = g_2(\alpha^{k_2})$, then $g_1(\alpha^{jk_1}) = g_2(\alpha^{jk_2})$ for all $1 \le j \le L-1$.

Proof. See Appendix-A. □

Theorem 4. For a network scalar linearly solvable over $\mathrm{GF}(2^{L-1})$, consider an arbitrary scalar linear solution over $\mathrm{GF}(2^{L-1})$ with local encoding kernels $g_{d,e}(\alpha)$ and the $|In(t)| \times \omega_t$ decoding matrix $\mathbf{D}_t(\alpha)$ for a receiver t, where both $g_{d,e}(\alpha)$ and entries in $\mathbf{D}_t(\alpha)$ are expressed as the unique polynomial evaluation subject to $(*)$. Define an $(L-1, L)$-fractional linear code over GF(2) for the network as follows:

- for each $s \in S$, the data unit transmitted on $e \in Out(s)$ is $[0\ \mathbf{m}_e']$, where $\mathbf{m}_e'$ is one of the $|Out(s)|$ $(L-1)$-dimensional binary row vectors generated at s.
- $\mathbf{K}_{d,e} = g_{d,e}(\mathbf{C}_L)$.

This code is an $(L-1, L)$ circular-shift linear solution of degree $\frac{L-1}{2}$. Moreover, the decoding matrix of the constructed circular-shift linear solution at receiver t is given by

$$\mathbf{D}_t(\mathbf{C}_L) \cdot (\mathbf{I}_{\omega_t} \otimes \tilde{\mathbf{I}}_L),$$

where $\tilde{\mathbf{I}}_L$ denotes the $L \times (L-1)$ matrix obtained by inserting a row vector of 1 on top of $\mathbf{I}_{L-1}$.

Proof. For every edge $e \in E$, denote by $\mathbf{F}_e$ and $\mathbf{f}_e$ the global encoding kernels of the considered $(L-1, L)$-fractional linear code over GF(2) and scalar linear solution over $\mathrm{GF}(2^{L-1})$, respectively. For brevity, write $E_S = Out(S)$ and $E_{S_t} = Out(S_t)$.

Consider an arbitrary receiver t. Denote by $\mathbf{B}_t(x)$ the $(|E| - \omega) \times |In(t)|$ index matrix of which the unique nonzero entry x in every column corresponds to an edge in $In(t)$. Thus,

$[\mathbf{f}_e]_{e\in In(t)} = [\mathbf{f}_e]_{e\notin E_S}\mathbf{B}_t(1)$ and $[\mathbf{F}_e]_{e\in In(t)} = [\mathbf{F}_e]_{e\notin E_S}\mathbf{B}_t(\mathbf{I}_L)$. Following the classic algebraic framework of scalar LNC for acyclic multicast networks [2],

$$\begin{aligned}&[\mathbf{f}_e]_{e\in In(t)}\\ =&[k_{d,e}]_{d\in E_S, e\notin E_S}\cdot(\mathbf{I}_{|E|-\omega}+[k_{d,e}]_{d,e\notin E_S}+[k_{d,e}]^2_{d,e\notin E_S}+\ldots+[k_{d,e}]^{|E|}_{d,e\notin E_S})\cdot\mathbf{B}_t(1)\\ =&[k_{d,e}]_{d\in E_S, e\notin E_S}\cdot(\mathbf{I}_{|E|-\omega}-[k_{d,e}]_{d,e\notin E_S})^{-1}\cdot\mathbf{B}_t(1),\end{aligned}$$

where the last equality holds as $[k_{d,e}]_{d,e\notin E_s}$ is a strictly upper triangular matrix and thus nilpotent. Write $\mathbf{M}(\alpha) = [\mathbf{f}_e]_{e\in In(t)}$, in which every entry is expressed as the polynomial evaluation subject to $(*)$. Thus,

$$\mathbf{M}(\alpha)\mathbf{D}_t(\alpha) = [\mathbf{U}_e^1]_{e\in E_{S_t}}. \tag{5}$$

Now consider the $(L-1, L)$-fractional code with $\mathbf{K}_{d,e} = g_{d,e}(\mathbf{C}_L)$. According to the framework of vector LNC [4],

$$\begin{aligned}[\mathbf{F}_e]_{e\in In(t)} &= [\mathbf{K}_{d,e}]_{d\in E_s, e\notin E_s}\cdot\left(\mathbf{I}_{(|E|-\omega)L}+[\mathbf{K}_{d,e}]_{d,e\notin E_s}+\ldots+[\mathbf{K}_{d,e}]^{|E|}_{d,e\notin E_s}\right)\cdot\mathbf{B}_t(\mathbf{I}_L)\\ &= [\mathbf{K}_{d,e}]_{d\in E_s, e\notin E_s}\cdot\left(\mathbf{I}_{(|E|-\omega)L}-[\mathbf{K}_{d,e}]_{d,e\notin E_s}\right)^{-1}\cdot\mathbf{B}_t(\mathbf{I}_L)\end{aligned}$$

By Lemma 2, $\mathbf{K}_{d,e} = g_{d,e}(\mathbf{C}_L) = \mathbf{V}_L\cdot g_{d,e}(\mathbf{\Lambda}_\alpha)\cdot\mathbf{V}_L^{-1}$. Thus,

$$[\mathbf{K}_{d,e}]_{d\in E_s, e\notin E_s} = (\mathbf{I}_\omega\otimes\mathbf{V}_L)\cdot[g_{d,e}(\mathbf{\Lambda}_\alpha)]_{d\in E_s, e\notin E_s}\cdot(\mathbf{I}_{|E|-\omega}\otimes\mathbf{V}_L^{-1})$$

$$[\mathbf{K}_{d,e}]^j_{d,e\notin E_s} = (\mathbf{I}_{|E|-\omega}\otimes\mathbf{V}_L)\cdot[g_{d,e}(\mathbf{\Lambda}_\alpha)]^j_{d,e\notin E_s}\cdot(\mathbf{I}_{|E|-\omega}\otimes\mathbf{V}_L^{-1}) \qquad \forall 1\leq j\leq|E|$$

In addition, note that

$$\mathbf{B}_t(\mathbf{I}_L) = (\mathbf{I}_{|E|-\omega}\otimes\mathbf{V}_L)\cdot\mathbf{B}_t(\mathbf{I}_L)\cdot(\mathbf{I}_{|In(t)|}\otimes\mathbf{V}_L^{-1}).$$

Consequently, $[\mathbf{F}_e]_{e\in In(t)} = (\mathbf{I}_\omega\otimes\mathbf{V}_L)\cdot\tilde{\mathbf{M}}\cdot(\mathbf{I}_{|In(t)|}\otimes\mathbf{V}_L^{-1})$, where $\tilde{\mathbf{M}}$ represents the $\omega L\times|In(t)|L$ matrix

$$\begin{aligned}&[g_{d,e}(\mathbf{\Lambda}_\alpha)]_{d\in E_s, e\notin E_s}\cdot\left(\mathbf{I}_{(|E|-\omega)L}+[g_{d,e}(\mathbf{\Lambda}_\alpha)]_{d,e\notin E_s}+\ldots+[g_{d,e}(\mathbf{\Lambda}_\alpha)]^{|E|}_{d,e\notin E_s}\right)\cdot\mathbf{B}_t(\mathbf{I}_L)\\ =&[g_{d,e}(\mathbf{\Lambda}_\alpha)]_{d\in E_s, e\notin E_s}\cdot\left(\mathbf{I}_{(|E|-\omega)L}-[g_{d,e}(\mathbf{\Lambda}_\alpha)]_{d,e\notin E_s}\right)^{-1}\cdot\mathbf{B}_t(\mathbf{I}_L)\end{aligned}$$

In the decoding matrix $\mathbf{D}_t(\mathbf{C}_L)\cdot(\mathbf{I}_{\omega_t}\otimes\tilde{\mathbf{I}}_L)$, note that

$$\mathbf{D}_t(\mathbf{C}_L) = (\mathbf{I}_{|In(t)|}\otimes\mathbf{V}_L)\cdot\mathbf{D}_t(\mathbf{\Lambda}_\alpha)\cdot(\mathbf{I}_{\omega_t}\otimes\mathbf{V}_L^{-1}).$$

Thus,

$$[\mathbf{F}_e]_{e\in In(t)}\cdot\mathbf{D}_t(\mathbf{C}_L) = (\mathbf{I}_\omega\otimes\mathbf{V}_L)\cdot\tilde{\mathbf{M}}\cdot\mathbf{D}_t(\mathbf{\Lambda}_\alpha)\cdot(\mathbf{I}_{\omega_t}\otimes\mathbf{V}_L^{-1}). \tag{6}$$

Observe that both $\tilde{\mathbf{M}}$ and $\mathbf{D}_t(\mathbf{\Lambda}_\alpha)$ can be respectively regarded as an $\omega \times |In(t)|$ and an $|In(t)| \times \omega_t$ block matrix, and every block entry is an $L \times L$ diagonal matrix. Hence, $\tilde{\mathbf{M}} \cdot \mathbf{D}_t(\mathbf{\Lambda}_\alpha)$ is an $\omega \times \omega_t$ block matrix with every block entry being an $L \times L$ diagonal matrix. For an integer k, define an $kL \times kL$ permutation matrix $\mathbf{P}_j$ (over GF(2)) as follows. It is a $k \times k$ block matrix $\begin{bmatrix} \mathbf{J}_{1,1} & \cdots & \mathbf{J}_{1,k} \\ \vdots & \ddots & \vdots \\ \mathbf{J}_{k,1} & \cdots & \mathbf{J}_{k,k} \end{bmatrix}$ in which the only nonzero entry in the $L \times L$ matrix $\mathbf{J}_{i,j}$ is in row j and column i. Rearrange the rows and columns in $\tilde{\mathbf{M}} \cdot \mathbf{D}_t(\mathbf{\Lambda}_\alpha)$ by respectively left-multiplying $\mathbf{P}_\omega$ and right-multiplying $\mathbf{P}_{\omega_t}^T$ to it. In this way, $\mathbf{P}_\omega \left(\tilde{\mathbf{M}} \cdot \mathbf{D}_t(\mathbf{\Lambda}_\alpha) \right) \mathbf{P}_{\omega_t}^T$ becomes an $L \times L$ block diagonal entry. The j^{th} diagonal block entry, $0 \le j \le L-1$, in it is an $\omega \times \omega_t$ matrix

$$[g_{d,e}(\alpha^j)]_{d \in E_s, e \notin E_s} \cdot \left(\mathbf{I}_{(|E|-\omega)L} - [g_{d,e}(\alpha^j)]_{d,e \notin E_s} \right)^{-1} \cdot \mathbf{B}_t(1) \cdot \mathbf{D}_t(\alpha^j) = \mathbf{M}(\alpha^j) \cdot \mathbf{D}_t(\alpha^j),$$

where the equality holds because of the definition of $\mathbf{M}(\alpha)$ and Lemma 3.c). In total,

$$\mathbf{P}_\omega \left(\tilde{\mathbf{M}} \cdot \mathbf{D}_t(\mathbf{\Lambda}_\alpha) \right) \mathbf{P}_{\omega_t}^T = \begin{bmatrix} \mathbf{M}(1)\mathbf{D}_t(1) & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{M}(\alpha)\mathbf{D}_t(\alpha) & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{M}(\alpha^{L-1})\mathbf{D}_t(\alpha^{L-1}) \end{bmatrix}.$$

By (5), $\mathbf{M}(\alpha)\mathbf{D}_t(\alpha) = [\mathbf{U}_e^1]_{e \in E_{S_t}}$. As a consequence of Lemma 3.c),

$$\mathbf{M}(\alpha^j)\mathbf{D}_t(\alpha^j) = [\mathbf{U}_e^1]_{e \in E_{S_t}} \quad \forall 1 \le j \le L-1.$$

In addition, write $\mathbf{M}(1)\mathbf{D}_t(1) = \begin{bmatrix} a_{11} & \cdots & a_{1\omega_t} \\ \vdots & \ddots & \vdots \\ a_{\omega 1} & \cdots & a_{\omega\omega_t} \end{bmatrix}$. Note that the entries a_{ij} belong to GF(2). Then,

$$\tilde{\mathbf{M}} \cdot \mathbf{D}_t(\mathbf{\Lambda}_\alpha) = \begin{bmatrix} \begin{smallmatrix} a_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_{11} \end{smallmatrix} & \cdots & \begin{smallmatrix} a_{1\omega_t} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_{1\omega_t} \end{smallmatrix} \\ \vdots & \ddots & \vdots \\ \begin{smallmatrix} a_{\omega 1} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_{\omega 1} \end{smallmatrix} & \cdots & \begin{smallmatrix} a_{\omega\omega_t} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_{\omega\omega_t} \end{smallmatrix} \end{bmatrix}, \tag{7}$$

where $\mathbf{J}_{i,j}$, $1 \le i \le \omega$, $1 \le j \le \omega_t$, is set to $\mathbf{I}_{L-1}$ if the $(i,j)^{th}$ entry in $[\mathbf{U}_e^1]_{e \in E_{S_t}}$ is equal to 1, and set to the $(L-1) \times (L-1)$ zero matrix otherwise. Let $\hat{\mathbf{I}}_L$ denote the $L \times L$ matrix which is identical to $\mathbf{I}_L$ except for the $(1,1)^{st}$ entry equal to 0, and $\mathbf{1}_L$ denote the $L \times L$ matrix with all entries equal to 1. It can be readily checked that

$$\mathbf{V}_L \cdot \hat{\mathbf{I}}_L \cdot (\mathbf{1}_L + \mathbf{V}_L^{-1}) \cdot \tilde{\mathbf{I}}_L = \tilde{\mathbf{I}}_L. \tag{8}$$

Based on (6), (7) and (8), we have

$$\begin{aligned}
&[\mathbf{F}_e]_{e\in In(t)} \cdot \mathbf{D}_t(\mathbf{C}_L) \cdot (\mathbf{I}_{\omega_t} \otimes \tilde{\mathbf{I}}_L) \\
=&(\mathbf{I}_\omega \otimes \mathbf{V}_L) \cdot \tilde{\mathbf{M}} \cdot \mathbf{D}_t(\mathbf{\Lambda}_\alpha) \cdot (\mathbf{I}_{\omega_t} \otimes \mathbf{V}_L^{-1}) \cdot (\mathbf{I}_{\omega_t} \otimes \tilde{\mathbf{I}}_L) \\
=&(\mathbf{I}_\omega \otimes \mathbf{V}_L) \cdot \tilde{\mathbf{M}} \cdot \mathbf{D}_t(\mathbf{\Lambda}_\alpha) \cdot (\mathbf{I}_{\omega_t} \otimes (\hat{\mathbf{I}}_L \cdot (\mathbf{1}_L + \mathbf{V}_L^{-1}) \cdot \tilde{\mathbf{I}}_L)) \\
=&(\mathbf{I}_\omega \otimes \mathbf{V}_L) \cdot \begin{bmatrix} \begin{smallmatrix} a_{11} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_{11} \end{smallmatrix} & \cdots & \begin{smallmatrix} a_{1\omega_t} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_{1\omega_t} \end{smallmatrix} \\ \vdots & \ddots & \vdots \\ \begin{smallmatrix} a_{\omega 1} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_{\omega 1} \end{smallmatrix} & \cdots & \begin{smallmatrix} a_{\omega\omega_t} & \mathbf{0} \\ \mathbf{0} & \mathbf{J}_{\omega\omega_t} \end{smallmatrix} \end{bmatrix} \cdot (\mathbf{I}_{\omega_t} \otimes (\hat{\mathbf{I}}_L \cdot (\mathbf{1}_L + \mathbf{V}_L^{-1}) \cdot \tilde{\mathbf{I}}_L)) \\
=&(\mathbf{I}_\omega \otimes \mathbf{V}_L) \cdot ([\mathbf{U}_e^1]_{e\in E_{S_t}} \otimes \hat{\mathbf{I}}_L) \cdot (\mathbf{I}_{\omega_t} \otimes (\mathbf{1}_L + \mathbf{V}_L^{-1}) \cdot \tilde{\mathbf{I}}_L) \\
=&[\mathbf{U}_e^1]_{e\in E_{S_t}} \otimes \tilde{\mathbf{I}}_L.
\end{aligned}$$

Finally, as for each $e \in Out(S)$, the binary sequences transmitted on e is $[0\ \mathbf{m}'_e]$, $G_S = \mathbf{I}_\omega \otimes [\mathbf{0}\ \ \mathbf{I}_{L-1}]$, *i.e.*,

$$[\mathbf{m}_e]_{e\in Out(S)} = [\mathbf{m}'_e]_{e\in Out(S)} \cdot (\mathbf{I}_\omega \otimes [\mathbf{0}\ \ \mathbf{I}_{L-1}]) \,.$$

In summary,

$$\begin{aligned}
&\mathbf{G}_S \cdot [\mathbf{F}_e]_{e\in In(t)} \cdot \mathbf{D}_t(\mathbf{C}_L) \cdot (\mathbf{I}_{\omega_t} \otimes \tilde{\mathbf{I}}_L) \\
=&(\mathbf{I}_\omega \otimes [\mathbf{0}\ \ \mathbf{I}_{L-1}]) \cdot ([\mathbf{U}_e^1]_{e\in E_{S_t}} \otimes \tilde{\mathbf{I}}_L) = \mathbf{U}_t \otimes \mathbf{I}_{L-1} = [\mathbf{U}_e^{L-1}]_{e\in E_{S_t}},
\end{aligned}$$

i.e., receiver t can recover $(L-1)$-dimensional source row vectors $\mathbf{m}'_e$, $e \in Out(S_t)$ generated by sources in S_t based on the decoding matrix $\mathbf{D}_t(\mathbf{C}_L) \cdot (\mathbf{I}_{\omega_t} \otimes \tilde{\mathbf{I}}_L)$. □

One may observe that the mapping from $k_{d,e} \in$ GF(2^{L-1}) to $\mathbf{K}_{d,e} \in \mathcal{C}_L$ used in the above theorem is one-to-one correspondence. However, such a mapping is not an isomorphism for $\mathcal{C}_L$ contains zero-divisors and is not an integral domain (e.g., $\mathbf{I}_L + \mathbf{C}_L$ is not invertible for any $L \geq 1$). This makes the established intrinsic connection between circular-shift LNC and scalar LNC non-trivial.

It turns out that when L is a prime with primitive root 2, as long as a general network has a scalar linear solution over GF(2^{L-1}), it has an alternative $(L-1, L)$ circular-shift linear solution of degree $(L-1)/2$ too. Different from previous studies in [9]-[11], which mainly consider low complexity encoding operations, the constructed $(L-1, L)$ circular-shift linear solution builds up not only local encoding kernels, but also the decoding matrix based on cyclic permutation matrices.

IV. Deterministic Circular-Shift LNC on Multicast Networks

A. Deterministic Construction

In this section, we still assume that L is a prime with primitive root 2. When L is larger than the number $|T|$ of receivers, the work in [11] has already proved that there exists an $(L-1, L)$ circular-shift linear solution of degree 1 for a multicast network. In addition, when reference [12] (Theorem 7) shows the existence of a low-complexity functional-repair regenerating code for a distributed storage system, it essentially proves the existence of an $(L-1, L)$ circular-shift linear solution of degree $\frac{L-1}{2}$ for that system. However, how to efficiently construct such desired circular-shift linear solutions is unknown. This issue is solved by Theorem 4 which reduces the construction of a (fractional) circular-shift linear solution to the construction of a scalar linear solution. Unlike a general network, which may not have a linear solution over any module alphabet [13], there are various known algorithms, such as the ones in [5] and [6], to efficiently construct a scalar linear solution.

Corollary 5. Let $1 \leq \delta \leq \frac{L-1}{2}$. For a multicast network, an $(L-1, L)$ circular-shift linear solution of degree δ can be efficiently constructed if $\binom{L}{0} + \binom{L}{1} + \ldots + \binom{L}{\delta} \geq |T|$.

Proof. By Lemma 3.a), GF(2^{L-1}) contains a primitive L^{th} root of unity, which will be denoted by α. Let $\mathcal{C}$ be a set of elements in GF(2^{L-1}) which can be expressed in the form $a_0 + a_1\alpha + \ldots + a_{L-1}\alpha^{L-1}$ such that at most δ nonzero binary coefficients a_j, $0 \leq j \leq L-1$, are nonzero. Lemma 3.b) implies that $\mathcal{C}$ contains $\binom{L}{0} + \binom{L}{1} + \ldots + \binom{L}{\delta}$ distinct elements. Then, if $|\mathcal{C}|$ is no smaller than the number of receivers, a scalar linear solution over $\mathrm{GF}(2^{L-1})$ can be efficiently constructed by the algorithm in [5] with local encoding kernels selected from $\mathcal{C}$. Thus, by Theorem 4, it directly induces an $(L-1, L)$ circular-shift linear solution as well as the concomitant decoding matrix at every receiver. □

While the constructed circular-shift linear solution has one-bit redundancy per edge transmission in Corollary 5, we next show that this is inevitable and that in the most general setting, circular-shift LNC is still insufficient to achieve the exact capacity of certain multicast networks.

Proposition 6. For $n \geq 4$, both the classical $(n, 2)$-Combination Network (See, e.g., [17][10]) depicted in Fig. 1 and the Swirl Network depicted in Fig. 2 with parameter $\omega = n$ designed in [15] are not (L, L) circular-shift linearly solvable of degree L for any $L \geq 1$.

Proof. Consider an arbitrary matrix $\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j$ in $\mathcal{C}_L$ with an even number of nonzero coefficients a_j. We shall first show that $rank\left(\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j\right) < L$ regardless of the choice of L. When L is odd, $\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j$ can be decomposed into $\mathbf{V}_L \cdot \sum_{j=0}^{L-1} a_j \mathbf{\Lambda}_\alpha^j \cdot \mathbf{V}_L^{-1}$ as a consequence of Lemma 2 to be presented in the sequel. As the $(1,1)^{st}$ entry in $\mathbf{\Lambda}_\alpha^j$ is always equal to 1 regardless of the exponent j, $rank(\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j) = rank(\sum_{j=0}^{L-1} a_j \mathbf{\Lambda}_\alpha^j) < L$. When $L = 2^m$ for some m, $\left(\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j\right)^{2^m} = \sum_{j=0}^{L-1} a_j \mathbf{C}_L^{j2^m} = \sum_{j=0}^{L-1} \mathbf{I}_L$, which turns out to be a zero matrix, so that $\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j$ is not of full rank. Last assume $L = 2^m l$ for some odd $l > 1$. There must exist a positive integer i such that $2^{m+i} \equiv 2^m$ modulo L. Thus,

$$\left(\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j\right)^{2^{m+i}} = \sum_{j=0}^{L-1} a_j \mathbf{C}_L^{j2^{m+i}} = \sum_{j=0}^{L-1} a_j \mathbf{C}_L^{j2^m} = \left(\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j\right)^{2^m}.$$

This implies that $\left(\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j\right)^{2^m} \cdot \mathbf{M}$ is a zero matrix, where $\mathbf{M} = \left(\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j\right)^{2^{m+i}-2^m} + \mathbf{I}_L$. As there is an even number of nonzero coefficients a_j for $\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j$, when $\left(\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j\right)^{2^{m+i}-2^m}$ is represented as $\sum_{j=0}^{L-1} a_j' \mathbf{C}_L^j \in \mathcal{C}_L$ with $a_j' \in \{0,1\}$, there is also an even number of nonzero coefficients a_j'. Thus, the matrix $\mathbf{M}$ is nonzero, and hence $\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j$ is not of full rank. We conclude that $rank(\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j) < L$ for every $L \geq 1$.

As deduced in [7], when $n \geq 4$, a necessary condition for both the $(n,2)$-Combination Network and the Swirl Network with $|Out(s)| = n$ to have an L-dimensional vector linear solution over GF(2) is that there are two $L \times L$ invertible matrices $\mathbf{A}_1, \mathbf{A}_2$ over GF(2) such that $rank(\mathbf{A}_i + \mathbf{A}_j) = L$. Let $\mathbf{A}_1 = \sum_{j=0}^{L-1} a_{1j} \mathbf{C}_L^j$, $\mathbf{A}_2 = \sum_{j=0}^{L-1} a_{2j} \mathbf{C}_L^j$ be two invertible matrices in $\mathcal{C}_L$. The argument in the previous paragraph implies that the number of nonzero coefficients in $\{a_{1j} : 0 \leq j \leq L-1\}$ and in $\{a_{2j} : 0 \leq j \leq L-1\}$ must be odd. Thus, the number of nonzero coefficients in $\{a_{1j} + a_{2j} : 0 \leq j \leq L-1\}$ must be even, so $\mathbf{A}_1 + \mathbf{A}_2 = \sum_{j=0}^{L-1} (a_{1j} + a_{2j}) \mathbf{C}_L^j \in \mathcal{C}_L$ cannot be of full rank. We can then conclude that neither the $(n,2)$-Combination Network nor the Swirl Network is (L,L) circular-shift linearly solvable of degree L. □

According to Artin's conjecture on primitive roots (See, e.g., [16]), there are infinitely many primes with primitive root 2. While the conjecture is open, there are sufficiently many such primes L (See the table in [16]) to choose for efficient construction of an $(L-1, L)$ circular-shift linear solution.

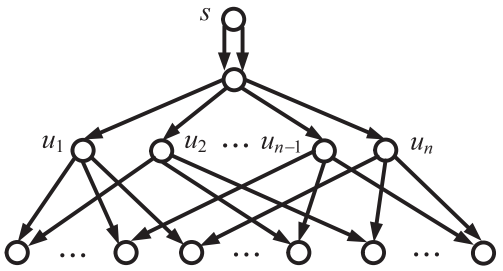

Fig. 1. The classical $(n, 2)$-Combination Network, which is known to have an L-dimensional vector linear solution over GF(2) if and only if $2^L \geq n - 1$.

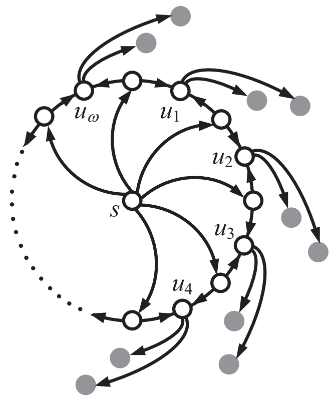

Fig. 2. The Swirl Network constructed in [15] has a non-depicted receiver connected from every set N of ω grey nodes with *maxflow*$(N) = \omega$.

B. Computational Complexity Comparison

We now theoretically compare the encoding and decoding complexity between circular-shift LNC and scalar LNC, from the perspective of required binary operations. Same as the analysis in [12], we shall ignore the complexity of a circular-shift operation on a binary sequence, which can be software implemented by modifying the pointer to the starting address in the sequence.

On a multicast network, let v be an intermediate node with indegree η, and $t \in T$ be a receiver. First consider a scalar linear solution over GF(2^L). Node v takes η multiplications and $\eta - 1$ additions over GF(2^L) to generate the data symbol $m_e \in \mathrm{GF}(2^L)$ for an outgoing edge $e \in Out(v)$. Receiver t takes ω^2 multiplications and $\omega(\omega - 1)$ additions over GF(2^L) in the decoding process to recover ω source data symbols. Here we just consider the standard implementation of multiplication in GF(2^L) by polynomial multiplication modulo an irreducible polynomial $g(x)$. For two polynomials $f_1(x), f_2(x)$ of degree $L - 1$ over GF(2), it takes L^2

binary multiplications and $L(L-1)$ binary additions to compute $f_1(x)f_2(x)$. It takes additional $(L-1)(\kappa-1)$ binary operations to obtain $f_1(x)f_2(x)$ modulo $g(x)$, where $\kappa \geq 3$ represents the number of nonzero coefficients in $g(x)$. In total, intermediate node v takes at least $\eta(2L^2+L)$ binary operations to obtain the L-bit data symbol m_e, and receiver t takes at least $\omega^2 L(2L+1)$ binary operations to recover ω L-bit source data symbols.

Next consider an $(L-1, L)$ circular-shift linear solution of degree δ constructed by Theorem 4. Node v takes $L\,(\delta\eta - 1)$ binary operations to obtain the L-dimensional binary row vector $\mathbf{m}_e = \sum_{d\in In(v)} \mathbf{m}_d \mathbf{K}_{d,e}$ for $e \in Out(v)$. For receiver t, recall that the decoding matrix in Theorem 4 is given by $\mathbf{D}_t(\mathbf{C}_L)\cdot(\mathbf{I}_\omega \otimes \tilde{\mathbf{I}}_L)$, where every block entry in the matrix $\mathbf{D}_t(\mathbf{C}_L)$ can be written as $\sum_{j=0}^{L-1} a_j \mathbf{C}_L^j$ with at most $\frac{L-1}{2}$ nonzero coefficients a_j. Thus, it takes $L\left(\frac{L-1}{2}\omega - 1\right)\omega$ binary operations to compute $[\mathbf{m}_e]_{e\in In(t)} \cdot \mathbf{D}_t(\mathbf{C}_L)$ and additional ωL binary operations to further obtain $[\mathbf{m}_e]_{e\in In(t)} \cdot \mathbf{D}_t(\mathbf{C}_L)\cdot(\mathbf{I}_\omega \otimes \tilde{\mathbf{I}}_L)$. In total, the number of binary operations is $\frac{1}{2}\omega^2 L(L-1)$.

For a more transparent and fairer comparison, consider a scalar linear solution over GF(2^m), an $(m, m+1)$ circular-shift linear solution of degree $\frac{m}{2}$, and an $(L, L+1)$ circular-shift linear solution of degree 1, where $m+1$, $L+1$ are primes with primitive root 2 and $|T| \leq L+2 \leq 2^m$. In this setting, such three linear solutions can be efficiently constructed according to Corollary 5. Table I lists the respective number of binary operations per bit for encoding at v and decoding at t.

TABLE I

NUMBER OF BINARY OPERATIONS PER BIT FOR ENCODING AND DECODING

	Encoding	Decoding
Scalar over GF(2^m)	$> 2\eta m$	$> \omega(2m+1)$
$(m, m+1)$ circular-shift of degree $\frac{m}{2}$	$\frac{1}{2}\eta m$	$\frac{1}{2}\omega(m+1)$
$(L, L+1)$ circular-shift of degree 1	$\eta - 1$	$\frac{1}{2}\omega(L+1) < \frac{1}{2}\omega 2^m$

It can be seen that for the considered circular-shift linear solution of degree $\delta = \frac{m}{2}$, the number of required binary operations per bit for both encoding and decoding can be approximately reduced by $3/4$ compared with the scalar linear solution. When the degree of the circular-shift linear solution decreases from $\frac{m}{2}$ to 1, the encoding complexity will decrease and the decoding complexity will increase. To our knowledge, this interesting tradeoff between encoding and decoding complexities for efficient construction of LNC schemes are new, and it makes circular-shift LNC more flexible to be applied in networks with different computational constraints.

One may observe that for the two circular-shift linear solutions in Table I, when δ decreases from $\frac{m}{2}$ to 1, the increasing rate of the decoding complexity is faster than the decreasing rate of the encoding complexity. The reason is that for the method proposed in this paper, the necessary block length $m+1$ for efficiently constructing a circular-shift linear solution of degree $\frac{m}{2}$ is $\lceil \log_2 |T| \rceil$, but the necessary block length $L+1$ for efficiently constructing a circular-shift linear solution of degree 1 is $|T|$. How to efficiently construct a circular-shift linear solution of degree 1 with a shorter block length deserves further investigation in the future work.

V. Random Circular-shift LNC on Multicast Networks

A. Randomly Analysis

As we have not known whether there are infinitely many primes with primitive root 2 yet, the results established in Theorem 4 and Corollary 5 are insufficient to imply that every multicast network is *asymptotically circular-shift linearly solvable*, that is, for any $\epsilon > 0$, it has an (L', L) circular-shift linear solution with $L'/L > 1 - \epsilon$. This motivates us to further study circular-shift LNC by random coding and to show, from a probabilistic perspective, that every multicast network is asymptotically circular-shift linearly solvable.

In this section, we concentrate on circular-shift LNC of degree 1, that is, all local encoding kernels are chosen from $\mathcal{C}_1 = \{\mathbf{0}, \mathbf{I}_L, \mathbf{C}_L, \ldots, \mathbf{C}_L^{L-1}\}$. Note that it can be regarded as a special class of *permutation-based* LNC schemes studied in [9], in which the local encoding kernels are chosen from $L!$ permutation matrices of size L as well as matrix $\mathbf{0}$.

Theorem 7. Let $\epsilon_L > 0$ be any function of L such that $\lim_{L\to\infty} \epsilon_L = 0$ and $\lim_{L\to\infty} \log \frac{2^{L\epsilon_L}}{L+1} = \infty$. Randomly construct an (L', L) circular-shift linear code, where $L' = \frac{\omega - |E|\epsilon_L}{\omega} L$, as follows:

- The $\omega L' \times \omega L$ coding matrix $\mathbf{G}_s$ operated at source s is uniformly and randomly chosen from all $\omega L' \times \omega L$ binary matrices.
- Every local encoding kernel is uniformly and randomly chosen from $\mathcal{C}_1 = \{\mathbf{0}, \mathbf{I}_L, \mathbf{C}_L, \ldots, \mathbf{C}_L^{L-1}\}$.

Then, the probability of this code forming an (L', L) linear solution is greater than $1 - 2^{-L\epsilon_L + \log(L+1) + \log|T||E|}$.

Proof. First, observe that for every receiver t, if $rank(\mathbf{G}_s[\mathbf{F}_e]_{e\in In(t)}) \geq \omega L'$, then there must exist an $\omega L \times \omega L'$ matrix $\mathbf{D}_t$ over GF(2) such that $\mathbf{G}_s[\mathbf{F}_e]_{e\in In(t)}\mathbf{D}_t = \mathbf{I}_{\omega L'}$, that is, receiver t can successfully recover the $\omega L'$ source data symbols. Thus, the probability of the randomly constructed code to be an (L', L)-fractional linear solution is lower bounded by

$$\begin{aligned} & Pr(rank(\mathbf{G}_s[\mathbf{F}_e]_{e\in In(t)}) \geq \omega L') \\ \geq & Pr(rank([\mathbf{F}_e]_{e\in In(t)}) \geq r) \cdot Pr(rank(\mathbf{G}_s[\mathbf{F}_e]_{e\in In(t)}) \geq \omega L' | rank([\mathbf{F}_e]_{e\in In(t)}) \geq r) \end{aligned}$$

for an arbitrary $r \geq \omega L'$.

Consider an arbitrary receiver t in the multicast network. As the maximum flow for t is ω, there are ω edge-disjoint paths from s to t. Let $E_t \subset E$ denote the set of edges in the ω edge-disjoint paths and index the edges in E_t as $e_1, e_2, \ldots, e_{|E_t|}$. Assume that there is an upstream-to-downstream order of E_t with $\{e_1, \ldots, e_\omega\} = Out(s)$ and $\{e_{|E_t|-\omega+1}, \ldots, e_{|E_t|}\} = In(t)$. Iteratively consider an set E_ω, which always consists of ω consecutive edges in E_t. Initially, $E_\omega = \{e_1, \ldots, e_\omega\}$ and by definition, $[\mathbf{F}_e]_{e\in E_\omega} = \mathbf{I}_{\omega L}$. In $i^{th} \geq 1$ iteration, based on the current setting E_ω which contains $e_{i+\omega-1}$ as the least ordered edge, define a new set $E'_\omega = E_\omega \backslash \{e_j\} \cup \{e_{i+\omega}\}$, where $(e_j, e_{i+\omega})$ forms an adjacent pairs of edges. Based on Lemma 8 to be presented in the sequel, it can be deduced (See Appendix-B) that

$$Pr(rank([\mathbf{F}_e]_{e\in E_\omega}) - rank([\mathbf{F}_e]_{e\in E'_\omega}) > L\epsilon_L) \leq 2^{-L\epsilon_L + \log(L+1)}. \tag{9}$$

Then, reset E_ω equal to E'_ω and proceed to the next iteration. In the final iteration, $E_\omega = In(t)$. As the number of iterations conducted for E_ω to change from $Out(s)$ to $In(t)$ is upper bounded by $|E| - \omega$, the following can be readily obtained by a union bound on (9):

$$Pr\{rank([\mathbf{F}_e]_{e\in In(t)}) \geq r\} \geq (1 - 2^{-L\epsilon_L + \log(L+1)})^{|E|-\omega} > 1 - (|E| - \omega) \cdot 2^{-L\epsilon_L + \log(L+1)}, \tag{10}$$

where r is set to be $\omega L - L\epsilon_L(|E| - \omega)$.

Under the condition that $rank([\mathbf{F}_e]_{e\in In(t)}) \geq r$, it can be further deduced (See Appendix-B) that

$$Pr\{rank(\mathbf{G}_s[\mathbf{F}_e]_{e\in In(t)}) \geq \omega L' \mid rank([\mathbf{F}_e]_{e\in In(t)}) \geq r\} > 1 - \omega L' 2^{-\omega L \epsilon_L}. \tag{11}$$

Then, by combining (10) and (11),

$$\begin{aligned} & Pr(rank(\mathbf{G}_s[\mathbf{F}_e]_{e\in In(t)}) \\ > & 1-(|E|-\omega)\cdot 2^{-L\epsilon_L+\log(L+1)}-\omega L' 2^{-\omega L\epsilon_L} \\ > & 1-[(L+1)(|E|-\omega)+\omega L']\cdot 2^{-L\epsilon_L} \\ > & 1-(L+1)|E|(1-\epsilon_L)2^{-L\epsilon_L}. \end{aligned} \tag{12}$$

By taking a union bound on (12) for all receivers, the desired lower bound for the probability of the randomly constructed circular-shift linear code to be an (L', L)-fractional linear solution can be obtained. □

As a result, for an arbitrary multicast network, the probability for random circular-shift LNC to yield an asymptotic linear solution tends to 1 with block length L increasing to infinity. One may notice that in the work of [11], it was also proved proved that on a multicast network, the probability for a randomly constructed $(L-1, L)$ circular-shift linear code (of degree 1) to be a linear solution is lower bounded by $(1-|T|/L)^{\sum_{v:node}|In(v)||Out(v)|}$. Compared with this bound, the one obtained in Theorem 7 converges much faster as L appears as an exponent parameter instead of as a denominator parameter. In addition, it is essentially the same as the lower bound obtained in [9] for the probability of a randomly constructed permutation-based linear code to be a linear solution. This connection is particular interesting because the coding operations provided by circular-shift are much less than permutation operations. Thus, the asymptotic linear solvability characterization in Theorem 7 is stronger than the results in [9]. Next lemma is a key reason for the lower bound derived in [9] to essentially hold for circular-shift LNC.

Lemma 8. For an $L\times L$ matrix $\mathbf{A}$ uniformly and randomly chosen from $\{\mathbf{I}_L, \mathbf{C}_L, \ldots, \mathbf{C}_L^{L-1}\}$, a uniformly and randomly chosen $L\times L$ binary matrix $\mathbf{L}$, and any $\epsilon > 0$, the probability for the rank of $\mathbf{L}+\mathbf{A}$ to be less than $L(1-\epsilon)$ is at most $2^{-L\epsilon+log(L+1)}$.

Proof. See Appendix-C. □

B. Circular-shift LNC vs Permutation-based LNC

In the previous section, we showed that the circular-shift LNC and permutation-based LNC share an essentially same lower bound of successful probability to yield an asymptotic linear solution. As only when the block length L is sufficiently long, the bound can start yielding

TABLE II

SUCCESSFUL PROBABILITY OF RANDOMLY GENERATING AN (L', L)-FRACTIONAL LINEAR SOLUTION ON $(4, 2)$ COMBINATION NETWORK

(L', L)	Circular-shift	Permutation
$(15, 16)$	0.1055	0.0168
$(30, 32)$	0.5894	0.3358
$(60, 64)$	0.7031	0.9349
$(120, 128)$	0.9996	0.9998

a positive value, it does not shed light on the asymptotic behavior for shorter block lengths. We next attempt to numerically analyze the successful probability of randomly generating a circular-shift and a permutation-based linear solution of the same rate $r = L'/L = 15/16$ on the $(4, 2)$-Combination Network, as shown in Table II. It can be seen that even though the successful probability for permutation-based LNC converges faster than the one for circular-shift LNC, for moderate block length $L = 128$, the successful probabilities for both have no big difference and are very close to 1.

Though permutation-based LNC can be regarded as a generalization of circular-shift LNC (of degree 1), the above numerical result indicates that the much more local encoding kernel candidates it brings in ($L!$ vs $L + 1$) do not obviously help increase the successful probability of randomly constructing a solution. In addition, as to be shown in the next proposition, for both the Combination Networks and Swirl Networks, which do not have an (L, L) circular-shift linear solution for any L as proved in Proposition 6, permutation-based LNC is insufficient to achieve the exact multicast capacity either.

Proposition 9. For $n \geq 4$, both the $(n, 2)$-Combination Network depicted in Fig. 1 and the Swirl Network depicted in Fig. 2 with parameter $\omega = n$ do not have an L-dimensional vector linear solution over GF(2) with local encoding kernels chosen from $L!$ possible permutation matrices of size L and matrix $\mathbf{0}$, for any block length L.

Proof. See Appendix-D. □

It turns out that for multicast LNC, compared with permutation operations, circular-shifts do not lose much in terms of linear solvability, while have much less implementation complexity.

TABLE III

OVERHEADS OF RANDOM LNC SCHEMES UNDER ALPHABET SIZE 2^L

Schemes	Overheads
Scalar LNC	ωL bits
Circular-Shift LNC	ωL bits
Permutation-based LNC	$\Omega(\omega L \log_2 L)$
Vector LNC	ωL^2 bits

C. Overhead Analysis

In the practical implementation of random LNC, every packet transmitted along the network usually consists of a batch of data units (See, e.g., [18]). All data units belong to the same alphabet and all data units in the same packet correspond to the same global encoding kernel. As the global encoding kernel for a packet will be dynamically updated to indicate how the packet is linearly formed from the source packets, its information must be stored as part of the packet header. For a scalar linear code over GF(2^L), as the global encoding kernels are ω-dimensional vectors over GF(2^L), the overhead to store the information of a global encoding kernel is theoretically ωL bits. On the other hand, for random vector LNC, under the same block length L, the global encoding kernel becomes an $\omega L \times L$ matrix over GF(2) and thus the overhead to store the corresponding information theoretically extends to ωL^2 bits.

Proposition 10. Under the same block length L, for a random circular-shift linear code (of degree 1) and a random permutation-based linear code, where local encoding kernels are respectively randomly chosen from $\mathcal{C}_1 = \{\mathbf{0}, \mathbf{I}_L, \mathbf{C}_L, \ldots, \mathbf{C}_L^{L-1}\}$ and $L \times L$ permutation matrices, the overheads to store the global encoding kernel information are ωL and $\Omega(\omega L \log_2 L)$ bits, respectively.

Proof. Recall that $[\mathbf{F}_e]_{e \in out(s)} = \mathbf{I}_{\omega L}$, and for an outgoing edge e from a non-source node v, the global encoding kernel $\mathbf{F}_e$ can be expressed as $\mathbf{F}_e = \sum_{d \in In(v)} \mathbf{F}_d \mathbf{K}_{d,e}$. Then, when $\mathbf{F}_e$ is regarded as an ω-dimensional vector with each component being an $L \times L$ matrix, each of these ω matrices can be recursively written as a function of local encoding kernels, which are randomly chosen from $\mathcal{C} = \{\mathbf{0}, \mathbf{I}, \mathbf{C}_L, \ldots, \mathbf{C}_L^{L-1}\}$. As $\mathcal{C}$ is closed under multiplication by elements in $\mathcal{C}$, each of the ω components in $\mathbf{F}_e$ is a summation of some matrices in $\mathcal{C}$. Thus, the number of possible $L \times L$ matrices to appear in each component of $\mathbf{F}_e$ is

$$\binom{L}{0} + \binom{L}{1} + \ldots + \binom{L}{L} = 2^L,$$

which can be represented by L bits. In all, the total number of bits required to store the information of $\mathbf{F}_e$ is ωL.

For an L-dimensional permutation-based linear code, the number of local encoding kernel candidates is $L! = \Omega(\omega L \log_2 L)$. As the number of possibilities for every block entry in a global encoding kernel $\mathbf{F}_e$ is at least the number of local encoding kernels, the overhead to store the information of $\mathbf{F}_e$ is $\Omega(\omega L \log_2 L)$ bits. □

Table III summarizes the required overheads for global encoding kernels among the aforementioned four types linear network coding schemes. The table shows that under the same alphabet size, the overhead required by random circular-shift LNC is as small as that required by conventional scalar LNC, and is much smaller than that of permutation-based LNC and vector LNC. The results established in this section show that circular-shift LNC also has advantages on shorter overheads for random coding and suggest a new direction of practical implementation of LNC using efficient, randomized circular-shift operations.

VI. Concluding Remarks

In this work, we formulate circular-shift LNC of degree δ as a special type of vector LNC with local encoding kernels restricted to be summation of at most δ cyclic permutation matrices. The results subsequently obtained under this framework suggest the potential of circular-shift LNC to be deployed with lower implementation complexities in both deterministic and randomized manners, compared with the conventional scalar LNC and permutation-based LNC. From a theoretical point of view, we prove that circular-shift LNC is insufficient to achieve the exact capacity of certain multicast networks, but whether every multicast network is asymptotically circular-shift linearly solvable remains open. Other interesting future works include the design of (fractional) circular-shift linear solutions with block lengths different from primes with primitive root 2, and the study of circular-shift LNC over an arbitrary base field GF(p).

Appendix

A. Proof of Lemma 3

As $0 = \alpha^L + 1 = (\alpha+1)(\alpha^{L-1} + \ldots + \alpha + 1)$ and $\alpha \neq 0$, $f(\alpha) = 0$. Consequently, $f(\alpha^{2^j}) = f(\alpha)^{2^j} = 0$ for all $j \geq 0$. As the multiplicative order of 2 modulo L is $L-1$, $\alpha, \alpha^2, \ldots, \alpha^{2^{L-2}}$ are distinct elements, and thus constitute the $L-1$ roots of $f(x)$. This implies that $f(x)$ is irreducible over GF(2), so $\alpha \in \mathrm{GF}(2^{L-1})$.

Because $f(x)$ is irreducible over GF(2) and $f(\alpha) = 0$, $\{1, \alpha, \ldots, \alpha^{L-2}\}$ is a basis of GF(2^{L-1}) over GF(2). Thus, every element $k \in \mathrm{GF}(2^{L-1})$ can be uniquely written as $a_0 + a_1\alpha + \ldots + a_{L-2}\alpha^{L-2}$ with binary coefficients a_j, $0 \leq j \leq L-2$. Additionally set a_{L-1} to be 0. If the number of nonzero coefficients a_j is no larger than $\frac{L-1}{2}$, then $g(x) = a_{L-1}x^{L-1} + \ldots + a_1 x + a_0$ is the polynomial satisfying $(*)$. Otherwise, set $a'_j = 1 \oplus a_j$ for all $0 \leq j \leq L-1$. In this way, $g(x) = a'_{L-1}x^{L-1} + \ldots + a'_1 x + a'_0$ is the polynomial satisfying $(*)$. As there are total 2^{L-1} polynomials over GF(2) of degree at most $L-1$ where at most $\frac{L-1}{2}$ coefficients are nonzero, the polynomial subject to $(*)$ is unique.

As the multiplicative order of 2 modulo L is $L-1$, for each $1 \leq j \leq L-1$, there exists $i \geq 1$ such that $\alpha^j = \alpha^{2^i}$. Thus, when $g_1(\alpha^{k_1}) = g_2(\alpha^{k_2})$,

$$g_1(\alpha^{jk_1}) = g_1(\alpha^{2^i k_1}) = g_1(\alpha^{k_1})^{2^i} = g_2(\alpha^{k_2})^{2^i} = g_2(\alpha^{2^i k_2}) = g_2(\alpha^{jk_2}).$$

B. Justification of Bounds (9) and (11)

In this appendix, we provide a detailed proof on obtaining the bounds (9) and (11). Adopt the same notations as in the proof sketch following Theorem 7.

First we shall prove inequality (9). Recall that in the i^{th} round of the iterative process, E'_ω is formed from E_ω via substituting e_j by $e_{i+\omega}$, where $(e_j, e_{i+\omega})$ forms an adjacent pair of edges. Let $\hat{\mathbf{F}}$ be any $\omega L \times K$ submatrix of $[\mathbf{F}_e]_{e \in E_\omega}$ with $rank([\mathbf{F}_e]_{e\in E_\omega}) = rank(\hat{\mathbf{F}}) = K$. Write $\hat{\mathbf{F}} = [\hat{\mathbf{F}}_1\ \hat{\mathbf{F}}_0]$, where $\hat{\mathbf{F}}_0$, $\hat{\mathbf{F}}_1$ respectively consist of columns in $\mathbf{F}_{e_j}$ and $[\mathbf{F}_e]_{e \in E_\omega \backslash \{e_j\}}$ that are contained in $\hat{\mathbf{F}}$. Because $rank([\mathbf{F}_e]_{e\in E_\omega\backslash\{e_j\}}) \geq rank(\hat{\mathbf{F}}_1)$ and $[\mathbf{F}_e]_{e\in E'_\omega} = [[\mathbf{F}_e]_{e\in E_\omega\backslash\{e_j\}}\ \ \mathbf{F}_{e_{i+\omega}}]$,

$$\begin{aligned} & rank([\mathbf{F}_e]_{e\in E_\omega}) - rank([\mathbf{F}_e]_{e\in E'_\omega}) \\ \leq & rank([\hat{\mathbf{F}}_1\ \hat{\mathbf{F}}_0]) - rank([\hat{\mathbf{F}}_1\ \mathbf{F}_{e_{i+\omega}}]). \end{aligned}$$

In order to prove the bound (9) for general cases, it suffices to prove (9) under the assumption that the columns in $\mathbf{F}_{e_{i+\omega}}$ are only linearly dependent on column vectors in $\hat{\mathbf{F}}$. Then, there must exist $L \times L$ matrices $\hat{\mathbf{L}}_1$, $\hat{\mathbf{L}}_2$, and a randomly generated cyclic-permutation matrix $\mathbf{K}_{e_j,e_{i+\omega}}$ (the local encoding kernel for adjacent pair $(e_j, e_{i+\omega})$) such that

$$[\hat{\mathbf{F}}_1\ \mathbf{F}_{e_{i+\omega}}] = [\hat{\mathbf{F}}_1\ \hat{\mathbf{F}}_0] \begin{bmatrix} \mathbf{I}_{K-\hat{L}} & \hat{\mathbf{L}}_1 \\ \mathbf{0} & \hat{\mathbf{L}}_2 + \mathbf{K}_{e_j,e_{i+\omega}} \end{bmatrix},$$

where $\hat{L}$ refers to the number of columns in $\hat{\mathbf{F}}_0$. Subsequently,

$$\begin{aligned} &Pr(rank([\mathbf{F}_e]_{e\in E_\omega}) - rank([\mathbf{F}_e]_{e\in E'_\omega}) > L\epsilon_L) \\ \leq &Pr(rank([\hat{\mathbf{F}}_1\ \hat{\mathbf{F}}_0]) - rank([\hat{\mathbf{F}}_1\ \mathbf{F}_{e_{i+\omega}}]) > L\epsilon_L) \\ = &Pr(rank(\hat{\mathbf{L}}_2 + \mathbf{K}_{e_j,e_{i+\omega}}) < L - L\epsilon_L), \end{aligned}$$

and hence inequality (9) is a direct consequence of Lemma 8.

Next, we shall prove inequality (11). Assume that $r = \omega L - L\epsilon_L(|E| - \omega)$. Under this condition, the number of choices for the $\omega L' \times \omega L$ binary matrix $\mathbf{G}_s$ satisfying $rank(\mathbf{G}_s[\mathbf{F_e}]_{e\in In(t)}) \geq \omega L'$ is equal to

$$(2^{\omega L} - 2^{\omega L - r})(2^{\omega L} - 2^{\omega L - r + 1}) \ldots (2^{\omega L} - 2^{\omega L - r + \omega L' - 1}).$$

As $\mathbf{G}_s$ is uniformly and randomly chosen from all $2^{(\omega L')(\omega L)}$ possible $\omega L' \times \omega L$ binary matrices,

$$\begin{aligned} &Pr(rank(\mathbf{G}_s[\mathbf{F}_e]_{e\in In(t)}) \geq \omega L' \mid rank([\mathbf{F}_e]_{e\in In(t)}) \geq r) \\ &\quad = \frac{(2^{\omega L} - 2^{\omega L - r}) \ldots (2^{\omega L} - 2^{\omega L - r + \omega L' - 1})}{2^{(\omega L)(\omega L')}} \\ &\quad = (1 - 2^{-r})(1 - 2^{-r+1}) \ldots (1 - 2^{-r+\omega L' - 1}) \\ &\quad > (1 - 2^{-r+\omega L' - 1})^{\omega L'} \\ &\quad = (1 - 2^{-\omega L\epsilon_L - 1})^{\omega L'} \\ &\quad > 1 - \omega L' 2^{-\omega L\epsilon_L}. \end{aligned}$$

Inequality (11) has thus been established.

C. Proof of Lemma 8

For a fixed L-dimensional vector $\mathbf{v}$ over GF(2), the probability that $\mathbf{v}$ is in the null-space of $\mathbf{L}+\mathbf{A}$ is

$$\begin{aligned}&Pr[(\mathbf{L}+\mathbf{A})\mathbf{v}=\mathbf{0}]\\ =&Pr[\mathbf{L}\mathbf{v}=\mathbf{A}\mathbf{v}]\\ =&\begin{cases}\frac{1}{\binom{L}{w_H(\mathbf{v})}}, & w_H(\mathbf{L}\mathbf{v})=w_H(\mathbf{v})\\ 0, & otherwise\end{cases}\end{aligned}$$

where $\mathbf{v}'=\mathbf{L}\mathbf{v}$, and $w_H(\cdot)$ stands for the Hamming weight of a vector. The reason for the last equality to hold is as follows. First, note that since $\mathbf{A}$ acts as a random circular-shift operation on $\mathbf{v}$, $\mathbf{v}'=\mathbf{A}\mathbf{v}$ only if $\mathbf{v}'$ and $\mathbf{v}$ have the same Hamming weight. Next, when $\mathbf{A}$ is chosen from $\{\mathbf{I}_L,\mathbf{C}_L,\ldots,\mathbf{C}_L^{L-1}\}$, there are $l\le L$ vectors $\mathbf{v}'$ subject to $\mathbf{v}'=\mathbf{A}\mathbf{v}$. As it is possible that $\mathbf{C}^i\mathbf{v}=\mathbf{C}^j\mathbf{v}$ for some $0\le i<j\le L-1$, l can be strictly smaller than L. For the i^{th} possible vector $\mathbf{v}'$ subject to $\mathbf{v}'=\mathbf{A}\mathbf{v}$, let t_i be the number of matrices $\mathbf{C}_L^i$, $0\le i\le L-1$ subject to $\mathbf{v}'=\mathbf{C}_L^i\mathbf{v}$. Apparently, $\sum_{i=1}^l t_i=L$. Then,

$$\begin{aligned}Pr[\mathbf{L}\mathbf{v}=\mathbf{A}\mathbf{v}]&=\sum_{i=1}^{l}\frac{1}{\binom{L}{w_H(\mathbf{v})}}\times\frac{t_i}{L}\\ &=\frac{1}{L\binom{L}{w_H(\mathbf{v})}}\sum_{i=1}^{l}t_i\\ &=\frac{L}{L\binom{L}{w_H(\mathbf{v})}}\\ &=\frac{1}{\binom{L}{w_H(\mathbf{v})}}\end{aligned}$$

Now let $\mathbf{v}$ be chosen uniformly and randomly from L-dimensional binary vectors. Then the probability that $\mathbf{v}$ is in the null-space of $\mathbf{L}+\mathbf{A}$ is

$$\begin{aligned}Pr[(\mathbf{L}+\mathbf{A})\mathbf{v}=\mathbf{0}]\le&\frac{1}{2^L}\Sigma_{\mathbf{v}}\frac{1}{\binom{L}{w_H(\mathbf{v})}}\\ =&\frac{1}{2^L}\Sigma_{i=0}^{L}\binom{L}{w_H(\mathbf{v})}\frac{1}{\binom{L}{w_H(\mathbf{v})}}\\ =&\frac{L+1}{2^L},\end{aligned}$$

where the first equality holds due to the partitioning of the set of all L-dimensional binary vectors $\mathbf{v}$ into $L+1$ classes of different Hamming weights. Since there are L random choices for $\mathbf{A}$ and 2^L random choices for $\mathbf{v}$, the number of $(\mathbf{v}, \mathbf{A})$ pairs satisfying $\mathbf{Lv} = \mathbf{Av}$ is bounded by

$$L \times 2^L Pr[(\mathbf{L}+\mathbf{A})\mathbf{v} = \mathbf{0}] \leq L2^L \times \frac{L+1}{2^L} = L(L+1). \tag{13}$$

Let k denote the number of choices for $\mathbf{A}$ such that

$$rank(\mathbf{L}+\mathbf{A}) < L(1-\epsilon). \tag{14}$$

For each $\mathbf{A}$ subject to (14), the number of vectors $\mathbf{v}$ in the null space of $\mathbf{L}+\mathbf{A}$ is at least $2^{L(1-\epsilon)}$, i.e., the number of $(\mathbf{v}, \mathbf{A})$ pairs satisfying $\mathbf{Lv} = \mathbf{Av}$ is at least $2^{L(1-\epsilon)}$. Thus, as a consequence of (13),

$$k < \frac{L(L+1)}{2^{L(1-\epsilon)}}.$$

Since there are L possible choices for $\mathbf{A}$ in total, the desired probability is upper bounded by $[L(L+1)/2^{L\epsilon}]/L = (L+1)/2^{L\epsilon} = 2^{-L\epsilon+\log(L+1)}$

D. Proof of Proposition 9

Same as in the proof of Proposition 6, we start the proof from the following necessary condition for both the $(n,2)$-Combination Network and the Swirl Network with $|Out(s)| = n$ to be L-dimensional vector linearly solvable over GF(2): there are two $L \times L$ invertible matrices $\mathbf{A}_i, \mathbf{A}_j$ over GF(2) such that

$$rank(\mathbf{A}_i - \mathbf{A}_j) = L \tag{15}$$

It suffices to show that $rank(\mathbf{A}_i - \mathbf{A}_j) < L$ for two arbitrary permutation matrices of size L. First note that each of $\mathbf{A}_i$ and $\mathbf{A}_j$ has exactly one non-zero entry in every row and every column. In the case that $\mathbf{A}_i$ and $\mathbf{A}_j$ have a non-zero entry at a same position, $\mathbf{A}_i - \mathbf{A}_j$ has at least one zero row or zero column. Thus, $\det(\mathbf{A}_i - \mathbf{A}_j) = 0$ and $rank(\mathbf{A}_i - \mathbf{A}_j) < L$. It remains to prove, by induction, that $rank(\mathbf{A}_i - \mathbf{A}_j) < L$ in the case that $\mathbf{A}_i - \mathbf{A}_j$ has exactly two non-zero entries in each row and each column.

When $L = 2$, there are only 2! permutation matrices to be considered. Obviously, $rank(\mathbf{A}_i - \mathbf{A}_j) < 2$. Assume that when $L = m$, $rank(\mathbf{A}_i - \mathbf{A}_j) < m$. When $L = m+1$, assume that the $(i,1)$ and $(j,1)$ entries are 1 in the first column and then add the entire i^{th} row to the j^{th} row in $\mathbf{A}_i - \mathbf{A}_j$. Remove the row and column where $(i,1)$ entry locates and form a new matrix of size $\mathbf{M}$ of size m. Note that $\det(\mathbf{A}_i - \mathbf{A}_j) = \det(\mathbf{M})$. In addition, the j^{th} row in $\mathbf{M}$ either has all zero entries or contains exactly two non-zero entries. In the former case, $\det(\mathbf{M}) = 0$. In the latter case, $\mathbf{M}$ has exactly two non-zero entries in each column and each row. By induction assumption, $rank(\mathbf{M}) < m$, and hence $\det(\mathbf{M}) = 0$. We conclude that $\det(\mathbf{A}_i - \mathbf{A}_j) = 0$ and (15) does not hold for any L. This completes the proof.

References

[1] S.-Y. R. Li, R. W. Yeung, and N. Cai, “Linear network coding,” *IEEE Trans. Inf. Theory*, vol. 49, no. 2, Feb. 2003.

[2] R. Koetter and M. Médard, “An algebraic approach to network coding,” *IEEE/ACM Trans. Netw.*, vol. 11, No. 5, Oct. 2003.

[3] M. Médard, M. Effros, D. Karger, T. Ho, “On coding for non-multicast networks,” *Annual ALLERTON Conference*, 2003.

[4] J. B. Ebrahimi, C. Fragouli, “Algebraic algorithm for vecor network coding” *IEEE Trans. Inf. Theory*, vol. 57, no. 2, Feb. 2011.

[5] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, “Polynomial time algorithms for multicast network code construction,” *IEEE Trans. Inf. Theory*, vol. 51, no. 6, Jun. 2005.

[6] M. Langberg, A. Sprintson, and J. Bruck, “Network coding: a computational perspective,” *IEEE Trans. Inf. Theory*, vol. 55, no. 1, Jan. 2009.

[7] Q. T. Sun, X. Yang, K. Long, X. Yin, and Z. Li, “On vector linear solvability of multicast networks,” *IEEE Trans. Comm.*, Dec. 2016.

[8] T. Etzion and A. Wachter-Zeh, “Vector network coding based on subspace codes outperforms scalar linear network coding,” arXiv:1512.06352, 2016.

[9] S. Jaggi, Y. Cassuto, and M. Effros, “Low complexity Encoding for Network Codes,” *IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2006.

[10] M. Xiao, M. Médard, and T. Aulin, “A binary coding approach for combination networks and general erasure networks,” *IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2007.

[11] A. Keshavarz-Haddad and M. A. Khojastepour, “Rotate-and-add coding: A novel algebraic network coding scheme,” *IEEE ITW*, Ireland, 2010.

[12] H. Hou, K. W. Shum, M. Chen and H. Li, “BASIC codes: low-complexity regenerating codes for distributed storage systems,” *IEEE Trans. Inf. Theory*, vol. 62, no. 6, Jun. 2016.

[13] J. Connelly and K. Zeger, “A class of non-linearly solvable networks,” *IEEE Trans. Inf. Theory*, vol. 63, no. 1, Jan. 2017.

[14] J. Connelly and K. Zeger, “Linear network coding over rings part II: vector codes and non-commutative alphabets,” *IEEE Trans. Inf. Theory*, 2017.

[15] Q. T. Sun, X. Yin, Z. Li and K. Long, “Multicast network coding and field sizes,” *IEEE Trans. Inf. Theory*, vol. 61, no. 11, pp. 6182-6191, Nov. 2015.

[16] N. J. A. Sloane, “Primes with primitive root 2,” *The On-Line Encyclopedia of Integer Sequences*, https://oeis.org/A001122.

[17] C. K. Ngai, R. W. Yeung, “Network coding gain of combination networks,” *IEEE Inf. Theory Workshop (ITW)*, Oct. 2004.

[18] P. Chou, Y. Wu, K. Jain, “Practical network coding,” *Annual ALLERTON Conference*, 2003.