IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 2 August 2022, accepted 19 September 2022, date of publication 22 September 2022, date of current version 30 September 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3208907

RESEARCH ARTICLE

A Straight Construction Algorithm for a Delay Invariant Convolutional Network Coding on Cyclic Networks

XUBO ZHAO AND XIAOPING LI
College of Sciences, China University of Petroleum, Qingdao, Shandong 266580, China
Corresponding author: Xubo Zhao (zhaoxubo@upc.edu.cn)

This work was supported in part by the Shandong Provincial Natural Science Foundation of China under Grant ZR2019MF070, in part by the National Natural Science Foundation of China under Grant 61902429, in part by the Fundamental Research Funds for the Central Universities under Grant 20CX05012A and Grant 22CX03015A, and in part by the Major Scientific and Technological Projects of CNPC under Grant ZD2019-183-008.

ABSTRACT Delay invariant convolutional network codes (abbreviated DI-CNCs) can guarantee multicast communication at asymptotically optimal rates in networks with any delay profile. For a cyclic network, it has been shown that one can associate it with an acyclic network consisting of nodes in five layers, and the acyclic algorithm of $\mathbb{F}$-linear multicast can be employed to construct a DI-$\mathbb{F}$-CNC, as long as the field size is larger than the number of sinks in the cyclic network. In this paper, we present a directly feasible construction algorithm for a DI-$\mathbb{F}$-CNC over a cyclic network. Complexity of code construction and theoretical guarantees of algorithm implementation are also investigated in detail. The advantage of the straight construction algorithm is that for an existing code, when some sink nodes and associated edges are added, our algorithm just modifies the new assigned coding coefficients in an efficient localized manner, without the necessity to construct again the code in its expanding network.

INDEX TERMS Network coding, cyclic network, delay invariant convolutional network codes, construction algorithm.

I. INTRODUCTION

Network coding is an efficient paradigm in information transmission, which can improve the capacity achievability for both wireless and wired networks [1], [2], [3], [4], [5]. Convolutional network coding (CNC) was considered in [6], [7], [8], [9], [10], [11], [12], and [13], it is a form of linear network coding which deals with a pipeline of messages as a whole rather than individually. Due to the bidirectional communications, in practical setting, most networks are cyclic. Over a cyclic network [14], [15], the propagation and encoding of sequential data symbols convolve together, and the propagation delay becomes an inseparable issue in network coding. Thus, CNC is naturally adopted to ensure causal data propagation around cycles. As shown in [7] and [16], if the transmission delay is nonzero along every cycle, then data propagation in a causal manner can be assured in the cyclic network by a CNC. The optimal CNCs are demonstrated to achieve the maximum transmission rate from the source node to the set of eligible receiver nodes. For a multicast network, an optimal CNC can be constructed by the deterministic algorithm [16] or the random one [17] with respect to a certain delay pattern.

Delay-sensitive traffic systems (typical systems of this kind include multimedia services and satellite communications) have expressed a phenomenal growth in recent years [18], [19], [20], [21]. One feature of these systems is that information bits traversing the network have a strict deadline. In addition to delay constraints, real-time communication networks also require achieving the maximum transmission rate from the source node to its receiver nodes timely. Under the above delay-sensitive traffic network setting, especially

The associate editor coordinating the review of this manuscript and approving it for publication was Marco Martalo.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

over a cyclic network, one can consider deploying delay invariant CNCs (DI-CNCs) into it. DI-CNCs are firstly introduced by Sun *et al.* in [22], they are a new class of CNCs for multicast networks, which are not dependent on the delays of the network. If a DI-CNC is deployed into the network, any delay changes incurred by inappropriate synchronization or other issues have no impact on the multicast capacity of that network.

Sun *et al.* in [22] proved the existence of a DI-CNC over any symbol field, and showed a random coding technique suffices to construct a DI-CNC with high probability. Moreover, an algorithm was devised to construct a DI-CNC with scalar coding coefficients as long as the symbol field was no smaller than the number of receivers. And the algorithm converted a cyclic network to a responding equivalent acyclic network, and then the acyclic algorithm of linear multicast [23] can be employed to construct a DI-CNC with scalar coding coefficients.

However, if dynamic behavior of the network is considered, for example, new sinks are added in the network, the DI-CNC could be re-distributed by the indirect algorithm in [22]. To address this issue, in this paper, we present a straight construction algorithm for a $\mathbb{F}$-linear multicast, which qulifies to be a DI-CNC on cyclic networks. Complexity of code construction and the theoretical guarantees for algorithm implementation are also provided. The advantage of the straight construction algorithm is that if we add some sinks and associated edges in an existing code, our algorithm just modifies the new assigned coding coefficients in an efficient localized manner, without the necessity to construct again the code in its expanding network.

The rest of this paper is organized as follows. In Section II, some basic definitions and properties about network coding, convolutional network coding and delay invariant convolutional network coding are reviewed. In Section III, the DI-$\mathbb{F}$-CNC construction algorithm and the analyses of algorithm are provided. The conclusion of this paper is given in Section IV.

II. PRELIMINARIES

We adopt the following convention as in [7], [16], and [22].

A *network* can be modeled as a finite directed multigraph $N = (V, E)$, or simply N, where V is the set of nodes and E is the set of edges. N contains a unique node with no incoming edges. This node is called the *source*, denoted by s. No edge loops around a node. Every edge represents a transmission channel of unit capacity.

For the directed edge $e = (u, v)$ from node u to node v, denote $u = \text{tail}(e)$ and $v = \text{head}(e)$. For every node v, denote the sets of its incoming and outgoing edges by $\text{In}(v)$ and $\text{Out}(v)$, respectively. An ordered pair (d, e) of edges is called an *adjacent pair* when there is a node v such that $d \in \text{In}(v)$ and $e \in \text{Out}(v)$.

Outgoing edges from s are called *data-generating edges*. Abbreviate $|\text{Out}(s)|$ as ω, which represents the (fixed) data generating rate from the source. In this paper, we assume that the ordering on edges of E is led by data-generating edges.

A *sink* means a non-source node r to which there are ω edge-disjoint paths from s. The set of τ sink nodes is denoted by $R = \{r_1, r_2, \cdots, r_\tau\} \subset V$.

Let $\mathbb{F}$ be a finite field, which represents the symbol alphabet, and $\mathbb{F}_q$ the finite field with q elements. Similarly to [22], let $\mathbf{P}$ be a principal ideal domain (PID), which represents the general ensemble of data units. Definition below describes a linear network coding where the base field is generalized to a PID.

Definition 1: A $\mathbf{P}$-*linear network coding* ($\mathbf{P}$-LNC) on a network $N = (V, E)$ means the assignment of an element $l_{d,e}$ in $\mathbf{P}$ to every pair (d, e) of edges such that $l_{d,e} = 0$ when (d, e) is not an adjacent pair. The element $l_{d,e}$ is called the *coding coefficient* or *local encoding kernel* (LEK) for the pair (d, e).

Generally, $\mathbf{P} = \mathbb{F}$ in conventional network coding, and $\mathbf{P} = \mathbb{F}\langle z\rangle$ in CNC, where $\mathbb{F}\langle z\rangle$ is a rational power series ring, z is the dummy variable that represents a unit-time delay [24], [25]. An $\mathbb{F}$-CNC means an $\mathbb{F}\langle z\rangle$-LNC $(l_{d,e})$ with $(l_{d,e}) \in \mathbb{F}\langle z\rangle$.

For linear network coding, any edge e has a global coding vector associated with it as follows.

Definition 2: A set of *coding vectors* or *global encoding kernel* (GEK) for a $\mathbf{P}$-linear network coding with LEKs means an assignment of an $\omega-$dimensional column vector $\mathbf{g}_e$ over $\mathbf{P}$ to each edge e such that

(i) $\mathbf{g}_e = \sum_{d \in \text{In}(v)} l_{d,e}\mathbf{g}_d$, where $e \in \text{Out}(v)$, v is a nonsource node,

(ii) the vectors $\mathbf{g}_e$, $e \in \text{Out}(s)$, forms the standard basis of the free module $\mathbf{P}^\omega$.

Denote by K the $|E|\times|E|$ matrix $[l_{d,e}]_{d,e\in E}$, where rows and columns are indexed according to the ordering of edges. Then the conditions (i) and (ii) in Definition 2 can be combined into the matrix equation

$$[\mathbf{g}_e]_{e\in E} = [\mathbf{g}_e]_{e\in E} \cdot K + H_s, \tag{1}$$

where $[\mathbf{g}_e]_{e\in E}$ is the $\omega \times |E|$ matrix obtained by juxtaposing the GEKs $\mathbf{g}_e$ with $e \in E$, H_s an $\omega \times |E|$ matrix formed by appending $|E| - \omega$ columns of zeroes to the $\omega \times \omega$ identity matrix I_ω. And this can be expressed in the following equivalent forms:

$$[\mathbf{g}_e]_{e\in E} \cdot (I_{|E|} - K) = H_s, \tag{2}$$

namely,

$$\det(I_{|E|} - K)[\mathbf{g}_e]_{e\in E} = H_s \cdot (I_{|E|} - K)^*, \tag{3}$$

where $\det(I_{|E|} - K)$, $(I_{|E|} - K)^*$ are the discriminant and the adjugate of the matrix $I_{|E|} - K$, respectively.

According to (3), if the discriminant $\det(I_{|E|} - K)$ is zero, then for $\mathbf{g}_e$, none or multiple solutions exsit.

Definition 3: The *discriminant* of a $\mathbf{P}$-linear network coding on a network $N = (V, E)$ is $\det(I_{|E|} - K)$. The code is said to be *nonsingular* when the discriminant is nonzero. A nonsingular code is said to be *normal* when it determines a unique set of coding vectors.

Normality of a **P**-linear network coding is a prerequisite to the notion of data propagation via the code, a sufficient condition for normality of a **P**-linear network coding is that $\det(I_{|E|} - K)$ is a unit in **P** [6].

Definition 4: A normal **P**-linear network coding with the coding vectors $\mathbf{g}_e$ is called a **P**-*linear multicast* when

$$\text{rank}_{\mathbf{P}}(\text{span}\{\mathbf{g}_e : e \in \text{In}(r)\}) = \omega \text{ for every sink } r. \quad (4)$$

A linear multicast is an optimal network coding, which enables every eligible sink node receiving date from source s at the full rate ω.

Definition 5: A *delay function* on the network is a nonnegative integer function t, defined over the set of adjacent pairs such that, along every cycle, there is at least one pair (d, e) with $\text{t}(d, e) > 0$.

In order to assure causality of data transmission over a cyclic network, each cycle in the network should contain a positive delay.

Definition 6: A **P**-linear network coding is said to be t-*causal* if the coding coefficient for every adjacent pair (d, e) is divisible by $z^{\text{t}(d,e)}$.

The following definition presents a special CNC.

Definition 7: An $\mathbb{F}$-CNC $(l_{d,e})$ is called a delay invariant $\mathbb{F}$-CNC (DI-$\mathbb{F}$-CNC) if for any delay function t, the code $(l_{d,e}z^{\text{t}(d,e)})$ is a t-causal $\mathbb{F}$-convolutional multicast.

One of the merits of DI-CNC is that the code design is independent of delay functions.

It is known that over a cyclic network, there exists an $\mathbb{F}$-linear multicast with all coding coefficients belonging to any sufficiently large subset $\Phi \subset \mathbb{F}$ [7]. Also notice that every $\mathbb{F}$-linear multicast is a DI-$\mathbb{F}$-CNC ([22], Proposition 5). Thus, if we can construct an $\mathbb{F}$-linear multicast $(l_{d,e})$ on a cyclic network, then for any delay function t, the code $(l_{d,e}z^{\text{t}(d,e)})$ is a DI-$\mathbb{F}$-CNC.

Unlike the design scheme of DI-CNC in the associated acyclic network of the original network $N = (V, E)$ [22], we consider DI-CNC construction in the directed line graph of $N = (V, E)$.

The *directed line graph* of $N = (V, E)$ is defined as $\mathcal{N}(\mathcal{V}, \mathcal{E})$, or simply $\mathcal{N}$, with vertex set $\mathcal{V} = E \cup s \cup R$ and edge set $\mathcal{E} = \{(d, e) \in E^2 : \text{head}(d) = \text{tail}(e)\} \cup \{(s, e) : e \in \text{Out}(s)\} \cup \{(e, r_i) : e \in \text{In}(r_i), 1 \le i \le \tau\}$. We denote nodes of $\mathcal{N}(\mathcal{V}, \mathcal{E})$ as $e \in \mathcal{V}$, and the edges as $(d, e) \in \mathcal{E}$. To be specific, the nodes in $\mathcal{N}$ are the edges of N, the source s, and the sink nodes $r_i \in R, i = 1, 2, \cdots, \tau$. There is an edge (d, e) in $\mathcal{N}$ if in N, (d, e) is an adjacent pair. There is an edge (s, e) in $\mathcal{N}$ if in N, e is an outgoing edge of source node s. There is an edge (e, r_i) in $\mathcal{N}$ if in N, e is an incoming edge of sink node r_i. Clearly, the LEK $l_{d,e}$ for the adjacent pair (d, e) in N corresponds to the LEK for the edge (d, e) in $\mathcal{N}$, and the associated GEK $\mathbf{g}_e$ of edge e in N just corresponds to the associated GEK of node e in $\mathcal{N}$. The graphical representation of LEKs and GEKs on line graph $\mathcal{N}$ can be found in [11]. It is also obviously that if there are ω edges disjoint paths between source s and sink r in N, there are corresponding ω nodes disjoint paths in $\mathcal{N}$.

In our construction, we will employ partial encoding kernels (PEKs) to maintain the regularity of every basis of sink node. Before giving the definition of the PEK, we need the following notation.

A set ξ_i of exactly ω edges (nodes) in $\text{In}(r_i)$, is called a *basis* of the sink r_i in N (the associated line graph $\mathcal{N}$), if there are ω edges (nodes) disjoint paths $\mathcal{P}_i = \{P_1^i, \cdots, P_k^i, \cdots, P_\omega^i\}$, P_k^i is the k-th path of $\mathcal{P}_i$, $k \in \{1, 2, \cdots, \omega\}$, starting from imaginary edges (nodes) and ending at edges (nodes) in ξ_i, $i = 1, 2, \cdots, \tau$. We call this set of ω edges (nodes) disjoint paths $\mathcal{P}_i = \{P_1^i, \cdots, P_k^i, \cdots, P_\omega^i\}$, $k \in \{1, 2, \cdots, \omega\}$, an *associated flow* of basis ξ_i. A basis ξ_i is *regular* if $\text{rank}(\text{span}\{\mathbf{g}_e : e \in \xi_i\})$ (the rank of the linear span by all GEKs in set ξ_i) is equal to ω.

Let $O_i = \{e_1^i, \cdots, e_k^i, \cdots, e_\omega^i\}$ be a set of ω-nodes for associated flow $\mathcal{P}_i = \{P_1^i, \cdots, P_k^i, \cdots, P_\omega^i\}$, where each node e_k^i belongs to a different path P_k^i in $\mathcal{P}_i$, $k \in \{1, 2, \cdots, \omega\}$. Denote A_k^i be the subset of the path P_k^i, which is consisted of all nodes following the node $e_k^i \in O_i$ (not including e_k^i). And define B_k^i be the set of LEKs of the edges with tail in A_k^i and head in $\mathcal{N} \setminus A_k^i$ (an illustration of A_k^i and B_k^i can be found in [26, Fig. 1].

Now, we can give the definition of the PEK.

Definition 8 [16]: For any node e in the line graph $\mathcal{N}$, without loss of generality, assume that $e \triangleq e_k^i \in O_i$, then the PEK $\mathbf{u}_e(z)$ of the node e satisfies the conditions (i) and (ii) in Definition 2 (with $\mathbf{g}_d(z)$, $\mathbf{g}_e(z)$ replaced by $\mathbf{u}_d(z)$, $\mathbf{u}_e(z)$, respectively) when the LEKs in B_k^i are set to zero.

Notice that the difference between GEK $\mathbf{g}_e(z)$ and PEK $\mathbf{u}_e(z)$ is that for $\mathbf{g}_e(z)$, the LEKs in B_k^i may not currently be zero, since they can be determined in previous steps.

For the sake of clarity, we summarize some notations in Table 1.

III. DI-$\mathbb{F}$-CNC CONSTRUCTION ALGORITHM

In this section, we will present the DI-$\mathbb{F}$-CNC construction algorithm, which is shown in Algorithm 1. And our algorithm assumes that the code designer has full knowledge of the network. By the definition of $\mathbb{F}$-linear multicast, after the code construction, we must make sure that

(I) the code is normal, namely, $\det(I - K)$ is nonzero,

(II) all bases of sink nodes r_i, $i = 1, 2, \cdots, \tau$, are regular, namely, $\text{rank}(\text{span}\{\mathbf{g}_e : e \in \xi_i\}) = \omega$ for $i = 1, 2, \cdots, \tau$.

A. STATEMENT OF DI-$\mathbb{F}$-CNC CONSTRUCTION ALGORITHM

We consider the code construction on the line graph $\mathcal{N}(\mathcal{V}, \mathcal{E}) = \bigcup_{i=1}^{\tau} \mathcal{N}(\mathcal{P}_i)$ and process every basis ξ_i of sink node r_i, $i = 1, \cdots, \tau$, one after another. Initially, all LEKs in $\mathcal{N}(\mathcal{V}, \mathcal{E})$ are assigned to zeros, thus, $\det(I - K) = 1 \neq 0$, during the code construction, we will remain that $\det(I - K) \neq 0$. Note that a basis may have more than one associated flows, we choose any one of them, only the bases and their determined associated flows are taken into account in our algorithm. For any edge in the original network N, which does not take part in any of the associated flows, we can assign the

TABLE 1. Notations.

$N=(V,E)$, or N	a network modeled as a finite directed multigraph, where V is the set of nodes and E is the set of edges
s	the source node of $N=(V,E)$
$e=(u,v)$	the directed edge from node u to node v, denote $u=$ tail(e) and $v=$ head(e)
In(v) (Out(v))	the set of incoming (outgoing) edges to (from) the node $v\in V$
adjacent pair (d,e)	an ordered pair (d,e) of edges when there exists $v\in V$ with $d\in$ In(v) and $e\in$ Out(v)
$R=\{r_1,r_2,\cdots,r_\tau\}\subset V$	the set of τ sink nodes
$\mathcal{N}(\mathcal{V},\mathcal{E})$, or $\mathcal{N}$	the directed line graph of $N=(V,E)$ with vertex set $\mathcal{V}=E\cup s\cup R$ and edge set $\mathcal{E}=\{(d,e)\in E^2:$ head(d) = tail(e)$\}\cup\{(s,e):e\in$ Out(s)$\}\cup\{(e,r_i):e\in$ In(r_i)$,1\le i\le\tau\}$
$l_{d,e}$	the coding coefficient or local encoding kernel (LEK) for the pair (d,e)
K	the $\lvert E\rvert\times\lvert E\rvert$ local encoding kernel matrix $[l_{d,e}]_{d,e\in E}$, where rows and columns are indexed according to the ordering of edges
$\mathbf{g}_e$	the global encoding kernel (GEK) for e
ξ_i	the basis of the sink r_i in N ($\mathcal{N}$), where ξ_i consists of exactly ω edges in In(r_i), and there are ω edge (nodes) disjoint paths starting from imaginary edges (nodes) and ending at edges (nodes) in ξ_i
$\mathcal{P}_i=\{P_1^i,\cdots,P_k^i,\cdots,P_\omega^i\}$	the determined associated flow of basis ξ_i in $\mathcal{N}$, each $\mathcal{P}_i$ is consisted of ω node-disjoint paths in $\mathcal{N}$, where P_k^i is the k-th path of $\mathcal{P}_i$, $k\in\{1,2,\cdots,\omega\}$
$O_i=\{e_1^i,\cdots,e_k^i,\cdots,e_\omega^i\}$	a set of ω-nodes for associated flow $\mathcal{P}_i$, where each node e_k^i belongs to a different path P_k^i in $\mathcal{P}_i$
A_k^i	the subset of the path P_k^i, it includes all nodes following the node $e_k^i\in O_i$ (not including e_k^i)
B_k^i	the set of LEKs of the edges with tail in A_k^i and head in $\mathcal{N}\setminus A_k^i$
$\mathbf{u}_e$	the partial encoding kernel (PEK) for e

zero encoding coefficient to it or remove it directly from the network. According to [1], this will not affect achieving the optimal rate.

Also notice that U_i defines the set of PEKs of the nodes in O_i, we regard the linear independence for the PEKs of nodes in $\mathcal{N}(\mathcal{V},\mathcal{E})$, rather than GEKs. The GEKs do not always work to make every basis be regular, since their former values can be updated by the most recently assigned LEKs over a cyclic network (a detailed example of a bad code designed only using GEKs can be found in [16]). For nodes in $\mathcal{P}_i$, their PEKs may be different from their GEKs, however, in terms of the definition of PEKs, if we reach the last node in $\mathcal{P}_i$, the GEKs of ξ_i are exactly the PEKs of it. Note that at this time, rank(span$\{V(\xi_i)\}$) $=\omega$, namely, we ensure that basis ξ_i is full rank.

We traverse the nodes of O_i in topological order of $\mathcal{P}_i$. Initially, the set O_i is consisted of ω imaginary nodes, and it will be updated following subsequent processes. Assume that e_k^i is the first processing node, and $\overline{e}_k^i$ is the next processing node. We update O_i and U_i by $\overline{O}_i=(O_i\cup\overline{e}_k^i)\backslash e_k^i$ and $\overline{U}_i^C=\{$PEKs of $\overline{O}_i$ with LEK $l^C_{e_k^i,\overline{e}_k^i}\}$, respectively, where $l^C_{e_k^i,\overline{e}_k^i}$ is the current LEK between e_k^i and $\overline{e}_k^i$.

The transfer function $T_{d,e}$ from node d to node e on line graph $\mathcal{N}$ means the sum of all Π_j, where Π_j is the product of all LEKs encountered in tracing the jth path starting from d and ending at e. By convention, $T_{d,e}=0$ if there is no path starting from d and ending at e. We note that in the beginning, on line graph $\mathcal{N}$, $T_{e,e}=0$ for any node e.

Algorithm 1 The DI-$\mathbb{F}$-CNC Algorithm

Input: The original network N
Output: All LEKs of N

1: find the associated flow $\mathcal{P}_i$ between the source node s and the sink node r_i, $i=1,2,\cdots,\tau$;
2: find the line graph $\mathcal{N}(\mathcal{P}_i)$ for $\mathcal{P}_i$, $i=1,2,\cdots,\tau$;
// $\mathcal{N}(\mathcal{V},\mathcal{E})=\bigcup_{i=1}^{\tau}\mathcal{N}(\mathcal{P}_i)$
3: **for all** edges in $\mathcal{N}(\mathcal{V},\mathcal{E})$ **do**
4: let the LEKs of all edges be $0'$s;
5: **end for**
6: **for** $i=1$ to τ **do**
7: $O_i\leftarrow\{e_1^i,\cdots,e_k^i,\cdots,e_\omega^i\}$;
// initially, O_i is the set of ω imaginary nodes in $\mathcal{N}(\mathcal{P}_i)$
8: $U_i\leftarrow\{\mathbf{u}_{e_1^i},\cdots,\mathbf{u}_{e_k^i},\cdots,\mathbf{u}_{e_\omega^i}\}=\{\varepsilon_1^i,\cdots,\varepsilon_k^i,\cdots,\varepsilon_\omega^i\}$;
// ε_k^i is an ω-dim column vector, the k-th element is 1, the others are $0'$s, $\mathbf{u}_{e_k^i}$ is the PEK of node e_k^i, namely, U_i is initialized by a set of ω-dim unit column vectors, and the cardinality of U_i is ω
9: isLast$\leftarrow$false;
10: **while** isLast = false **do**
11: traverse the nodes in $\mathcal{P}_i$ by topological order;
// e_k^i is the first processing node, $\overline{e}_k^i$ is the next one after e_k^i
12: $\overline{O}_i\leftarrow(O_i\cup\overline{e}_k^i)\backslash e_k^i$;
// $\overline{e}_k^i$ is the next processing node after node e_k^i
13: $\overline{U}_i^C\leftarrow\{$PEKs of $\overline{O}_i$ with LEK $l^C_{e_k^i,\overline{e}_k^i}\}$;
// $l^C_{e_k^i,\overline{e}_k^i}$ is the current LEK of $(e_k^i,\overline{e}_k^i)$
14: **if** det$(I-K)=0$, or $T_{e_k^i,e_k^i}=1$, or rank(span$\{\overline{U}_i^C\})\neq\omega$, or rank(span$\{\mathbf{g}_e:e\in\xi_j\})\neq\omega$ for ξ_j, $1\le j\le i-1$ **then**
15: $(l^C_{e_k^i,\overline{e}_k^i},\overline{U}_i^C)\leftarrow$Pick a new LEK;
16: **end if**
17: $(O_i,U_i,e_k^i)\leftarrow(\overline{O}_i,\overline{U}_i^C,\overline{e}_k^i)$;
18: **if** e_k^i is the last processing node of $\mathcal{P}_i$ **then**
19: $V(\xi_i)\leftarrow U_i$;
20: isLast$\leftarrow$true;
21: **end if**
22: **end while**
23: **end for**

The main challenge of our construction algorithm lies in picking eligible LEKs (lines 14 to 16 in Algorithm 1). The LEK $l^C_{e_k^i,\overline{e}_k^i}$ should satisfy the following conditions,

(a). det$(I-K)\neq0$,

(b). $T_{e_k^i,e_k^i}\neq1$, where $T_{e_k^i,e_k^i}$ is the total transfer function from e_k^i to e_k^i,

(c). rank(span$\{\overline{U}_i^C\})=\omega$,

(d). rank(span$\{\mathbf{g}_e:e\in\xi_j\})=\omega$ for all $j<i$.

And Theorems 1-4 (in subsection III-D) guarantee that choosing LEKs in finite field $\widetilde{\mathbb{F}}$ is sufficient to maintain the

DI-$\mathbb{F}$-CNC conditions, where $|\widetilde{\mathbb{F}}| \geq \tau + 3$, and τ is the total number of all sink nodes.

B. COMPLEXITY OF THE CODE CONSTRUCTION

Firstly, we find the associated flow $\mathcal{P}_i$ between the source node s and the sink node r_i, $i = 1, 2, \cdots, \tau$. By Ford and Fulkerson algorithm [27], the total complexity of this step is $O(\tau|E|\omega)$, where ω is equal to the size of the minimal individual min-cut between s and the sink node r_i, $i = 1, 2, \cdots, \tau$.

Note that for the line graph $\mathcal{N}(\mathcal{V}, \mathcal{E})$ the $|E| \times |E|$ matrix K is given by

$$K = \begin{cases} l_{d,e}, & (d, e) \in \mathcal{E}, \\ 0, & \text{otherwise.} \end{cases}$$

We can label the line graph, and initialize the LEKs $l_{d,e}$, where $(d, e) \in \mathcal{E}$ in total complexity $O(|E|^2)$.

In the for-loop steps (lines 6-23 in Algorithm 1), for the ith for-loop step, $i \in \{1, 2, \cdots, \tau\}$, we have the following results.

(1) The complexity of initializing O_i and U_i is $O(\omega^2)$.

(2) For node e_k^i in line graph $\mathcal{N}$, the complexity of computing $\mathbf{u}_{e_k^i}$ is $O(|\text{In}(e_k^i)|\omega)$ (generally, $|\text{In}(e_k^i)| < |E|$).

(3) If we employ the linear independent test vector method given in [23, Lemma 5] or in [28], the complexities of checking the singularity of matrix $I - K$ and the full-rank condition of set $\overline{U}_i^C$ are $O(|E|)$ and $O(\omega)$, respectively.

(4) The transfer function $T_{e_k^i, e_k^i}$ from node e_k^i to itself is required to be computed. To do this, as in [16], we can substitute the relevant values of LEKs in matrix K, and it can be finished with complexity $O(|E|^2 \log\tau)$. Similar to [16] or [23], for brevity, we neglect the logarithmic factor, then the complexity is $O(|E|^2)$.

(5) In terms of [23, Lemma 7], determining the regularity of the relevant basis ξ_j, $1 \leq j \leq i-1$, can be implemented in complexity $O(|E|\tau\omega^2)$.

Notice that in the for-loop steps (lines 6-23 in Algorithm 1), we should successively process the nodes on the associated flow $\mathcal{P}_i$ in line graph $\mathcal{N}$. Thus, there are $|E|\tau$ iterations at most. Consequently, based on the above complexity analyses, the DI-$\mathbb{F}$-CNC construction in Algorithm 1 can be finished in expected complexity $O(\max\{\ |E|^3\tau,\ |E|^2\tau^2\omega^2\ \})$. In comparison, the indirect algorithm in [22] has complexity $O(|E|^3\tau)$ to construct a DI-$\mathbb{F}$-CNC. Therefore, if $\tau\omega^2 \leq |E|$, the two algorithms possess the same complexities. However, the overhead of network converting by the indirect algorithm in [22] is avoided by our direct construction.

C. AN EXAMPLE NETWORK TO ILLUSTRATE DI-$\mathbb{F}$-CNC CONSTRUCTION ALGORITHM

We will illustrate DI-$\mathbb{F}$-CNC by means of a simple example network. We depict it in Fig.1, our target is to assign appropriate LEKs such that the network is a $\mathbb{F}$-linear multicast. The associated line graph of Fig.1 is given in Fig.2(a). For simplicity, below we discuss the simplified line graph $\mathcal{N}(\mathcal{V}, \mathcal{E})$ in Fig.3, that is, we omit the nodes which have a single input and a single output, since these nodes would simply receive and forward the same symbol. The bases of sinks r_1, r_2 are $\xi_1 = \{e_1, e_9\}$ and $\xi_2 = \{e_2, e_6\}$, respectively. Note that in Fig.1, the data generating rate from source s is $\omega = 2$. For convenience, source s in original network $N(V, E)$ is split into ω source s_i, $i = 1, \cdots, \omega$, in the associated line graph $\mathcal{N}(\mathcal{V}, \mathcal{E})$ of $N(V, E)$. Thus, the associated flow of ξ_1 is $\mathcal{P}_1 = \{P_1^1,\ P_2^1\} = \{s_1 \to e_1,\ s_2 \to e_2 \to e_6 \to e_9\}$. And the associated flow of ξ_2 is $\mathcal{P}_2 = \{P_1^2,\ P_2^2\} = \{s_2 \to e_2,\ s_1 \to e_1 \to e_9 \to e_6\}$ (lines 1-2). At the beginning, all LEKs in $\mathcal{N}$ are set to zeros (line 3-5). Assume that we go through

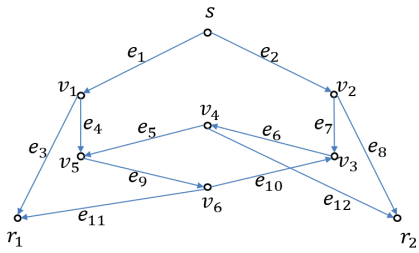

FIGURE 1. A cyclic network with one source s and two sinks r_1, r_2.

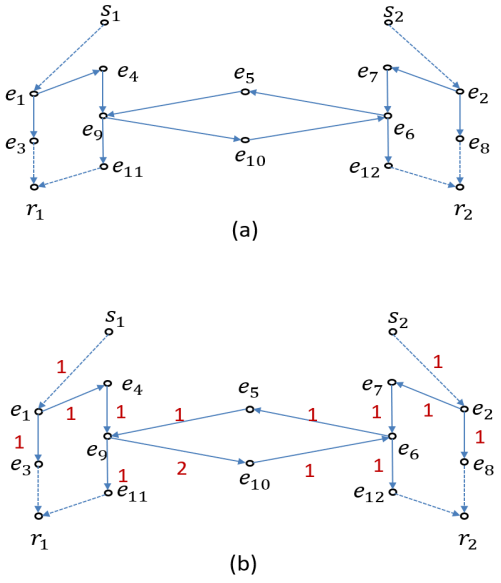

FIGURE 2. (a) The associated line graph of Fig.1. (b) The associated line graph of Fig.1, which sign the LEKs to construct a DI-$\mathbb{F}$-CNC in Fig.1.

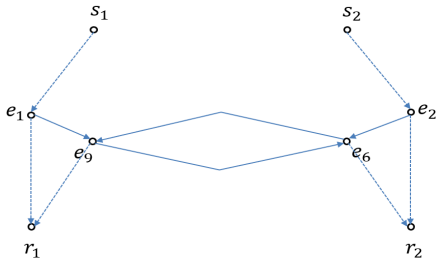

FIGURE 3. The simplified line graph of Fig.2(a).

the nodes $s_1, s_2, e_1, e_2, e_6, e_9$ in $\mathcal{P}_1$ in topological order, the nodes $s_1, s_2, e_1, e_2, e_9, e_6$ in $\mathcal{P}_2$ in topological order.

Initially, $O_1 = \{s_1, s_2\}$, $U_1 = \{(1\ 0)^\top, (0\ 1)^\top\}$ (lines 7-8), where the superscript $\top$ represents the transposed symbol. The next processing node should be e_1 with the current LEK coefficient $l_{s_1,e_1} = 0$. Thus, we have $\overline{O}_1 = \{e_1, s_2\}$ and $\overline{U}_1^C = \{(0\ 0)^\top, (0\ 1)^\top\}$ (lines 11-13). Note that when we process the nodes in $\mathcal{P}_1$, the matrix $I - K$ is upper triangular and on its diagonal all elements are equal to 1, then $\det(I - K) = 1 \neq 0$. Also note that there are no cycle in $\mathcal{P}_1$, and ξ_1 is the first processing basis, we do not need to check the regularities of other bases. Thus, we just need to compute rank(span$\{\overline{U}_1^C\}$). Since $\overline{U}_1^C$ is not full rank, we should pick a new LEK for (s_1, e_1) (lines 14-15). By Theorem 3, any non-zero value in $\widetilde{\mathbb{F}}$ ($|\widetilde{\mathbb{F}}| \geq 5$, assume that $|\widetilde{\mathbb{F}}| = 5$) will work, and assume that we select $l_{s_1,e_1}^C = 1$. Now, we can obtain that $\overline{U}_1^C = \{(1\ 0)^\top, (0\ 1)^\top\}$, which is full rank.

We update the sets O_1, U_1 by $O_1 = \{e_1, s_2\}$, $U_1 = \{\mathbf{u}_{e_1}, \mathbf{u}_{s_2}\} = \{(1\ 0)^\top, (0\ 1)^\top\}$, respectively (line 17). The next processing node in $\mathcal{P}_1$ is e_2, with the LEK $l_{s_2,e_2}^C = 0$ (return to line 11). After a similar discussion to e_2, we can obtain the updated values $l_{s_2,e_2}^C = 1$, $O_1 = \{e_1, e_2\}$, $U_1 = \{\mathbf{u}_{e_1}, \mathbf{u}_{e_2}\} = \{(1\ 0)^\top, (0\ 1)^\top\}$.

In the next steps, we will deal with nodes e_6, e_9, successively. After similar discussions, we can assign $l_{e_2,e_6}^C = 1$, $l_{e_6,e_9}^C = 1$, and get $\mathbf{u}_{e_6} = (0\ 1)^\top$, $\mathbf{u}_{e_9} = (0\ 1)^\top$. At this time, $O_1 = \xi_1 = \{e_1, e_9\}$, so we have processed the last node in $\mathcal{P}_1$, and we obtain $\{\mathbf{g}_{e_1}, \mathbf{g}_{e_9}\} = U_1 = \{\mathbf{u}_{e_1}, \mathbf{u}_{e_9}\} = \{(1\ 0)^\top, (0\ 1)^\top\}$, namely, rank(span$\{\mathbf{g}_e : e \in \xi_1\}$) $= \omega = 2$.

We recall that the associated flow of $\xi_2 = \{e_2, e_6\}$ is $\mathcal{P}_2 = \{P_1^2, P_2^2\} = \{s_2 \to e_2,\ s_1 \to e_1 \to e_9 \to e_6\}$. We process the nodes $s_1, s_2, e_1, e_2, e_9, e_6$ in $\mathcal{P}_2$ in topological order. Accordingly, we deal with sets $O_2 = \{s_1, s_2\}$, $O_2 = \{s_1, e_2\}$, $O_2 = \{e_1, e_2\}$, $O_2 = \{e_9, e_2\}$, $O_2 = \{e_6, e_2\}$, successively.

For $\xi_2 = \{e_2, e_6\}$, it is easy to see that we can maintain the values of $l_{s_2,e_2}^C = 1$, $l_{s_1,e_1}^C = 1$, and the current $O_2 = \{e_1, e_2\}$, $U_2 = \{\mathbf{u}_{e_1}, \mathbf{u}_{e_2}\} = \{(1\ 0)^\top, (0\ 1)^\top\}$. Now, following the topological order of nodes in $\mathcal{P}_2$, we process $O_2 = \{e_9, e_2\}$. The current value of l_{e_1,e_9}^C is 0, then $\overline{U}_2^C = \{\mathbf{u}_{e_2}, \mathbf{u}_{e_9}\} = \{(0\ 1)^\top, (0\ 1)^\top\}$, which is not full rank. Thus, by Theorem 3, we can assign $l_{e_1,e_9}^C = 1$. Now, $\overline{U}_2^C = \{\mathbf{u}_{e_2}, \mathbf{u}_{e_9}\} = \{(0\ 1)^\top, (1\ 1)^\top\}$, which is full rank. Note that at this time, $\det(I - K) = 1 \neq 0$, $T_{e_1,e_1} = 0 \neq 1$. Also notice that changing the value of l_{e_1,e_9}^C may affect the regularity of basis ξ_1, in which the GEK $\mathbf{g}_{e_9}$ has been determined with $l_{e_1,e_9}^C = 0$. We can compute $\mathbf{g}_{e_9} = (1, 1)^\top$ with new LEK $l_{e_1,e_9} = 1$, and it does not change the regularity of basis ξ_1.

Now we process the last node e_6 in $\mathcal{P}_2$ in topological order. At present, $O_2 = \{e_2, e_6\}$, $l_{e_9,e_6}^C = 0$, and $\overline{U}_2^C = \{\mathbf{u}_{e_2}, \mathbf{u}_{e_6}\} = \{(0\ 1)^\top, (0\ 1)^\top\}$, which is not full rank. Thus, by Theorem 3, we can assign $l_{e_9,e_6}^C = 1$. Now, $\overline{U}_2^C = \{\mathbf{u}_{e_2}, \mathbf{u}_{e_6}\} = \{(0\ 1)^\top, (1\ 1)^\top\}$, which is full rank. However, $T_{e_9,e_9} = 1$ with $l_{e_9,e_6}^C = 1$. By Theorem 2, we need to assign a new value of l_{e_9,e_6}^C. Assume that we set $l_{e_9,e_6}^C = 2$, $2 \in \widetilde{\mathbb{F}}$ ($|\widetilde{\mathbb{F}}| = 5$). Now, $T_{e_9,e_9} = 2$, and $\det(I - K) = 3 \neq 0$. By Eq.(2) or Eq.(3), it is also easy to compute that $\overline{U}_2^C = \{\mathbf{u}_{e_2}, \mathbf{u}_{e_6}\} = \{(0\ 1)^\top, (-2\ -1)^\top\} = \{\mathbf{g}_{e_2}, \mathbf{g}_{e_6}\}$, and $\{\mathbf{g}_{e_1}, \mathbf{g}_{e_9}\} = \{(1\ 0)^\top, (-1\ -1)^\top\}$.

Thus, by setting LEKs $l_{s_1,e_1}, l_{s_2,e_2}, l_{e_1,e_9}, l_{e_2,e_6}, l_{e_6,e_9}$ to 1, and l_{e_9,e_6} to 2, we can successfully construct a DI-$\mathbb{F}$-CNC in Fig.3, so do Fig.2 and Fig.1 (we sign the LEKs of the original network $N(V, E)$ in Fig.2(b)).

When we add some non-source nodes and associated edges in the original network, they can produce newly additional bases ξ_j and associated flows $\mathcal{P}_j$ ($j > \tau$). Assume that the already processed node prior to the new one follows the topological orders of associated flows $\mathcal{P}_j$ ($j > \tau$). Then constructing a DI-$\mathbb{F}$-CNC on the expanding network just corresponds to adding some new for-loop steps (lines 6-19) in Algorithm 1, namely, our algorithm can modify the already assigned LEKs in a localized manner.

For example, we add a sink r_3 in Fig.4 (the gray node in Fig.4), and connect it with the original network by directed edges e_{13}, e_{14}. The associated line graph of the extending network is depicted in Fig.5(a) (the added edges are also marked by gray dashed arrows). The associated flow for the new basis $\xi = \{e_{13}, e_{14}\}$ of sink r_3 is $\mathcal{P}_3 = \{P_1^3, P_2^3\} = \{s_1 \to e_1 \to e_4 \to e_{13},\ s_2 \to e_2 \to e_7 \to e_6 \to e_{14}\}$.

In Fig.5(a), we prefer the topological order for the nodes in $\mathcal{P}_3$ following the rule that the already processed nodes in original line graph are prior to the new nodes in the extending line graph. Thus, we process the nodes $s_1, s_2, e_1, e_2, e_4, e_7, e_{13}, e_6, e_{14}$ in $\mathcal{P}_3$ in topological order. Accordingly, we deal with sets $O_3 = \{s_1, s_2\}$, $O_3 = \{e_1, s_2\}$, $O_3 = \{e_1, e_2\}$, $O_3 = \{e_4, e_2\}$, $O_3 = \{e_4, e_7\}$, $O_3 = \{e_{13}, e_7\}$, $O_3 = \{e_{13}, e_6\}$, $O_3 = \{e_{13}, e_{14}\}$, successively. As the cases of $i = 1$ or 2 in the for-loop steps (lines 6-19), a similar analysis can be applied on the case $i = 3$. We conclude that a DI-$\mathbb{F}$-CNC can be constructed on the extending network in Fig.4, with all already assigned LEKs in Fig.2(b) remain unchanged and new LEKs $l_{e_4,e_{13}} = l_{e_6,e_{14}} = 1$, $l_{e_5,e_{13}} = 0$ (see Fig.5(b)).

Therefore, by the above example, using our construction algorithm, constructing a DI-$\mathbb{F}$-CNC on the expanding network just corresponds to adding some new for-loop steps (lines 6-19) in Algorithm 1. And the number of LEKs need-

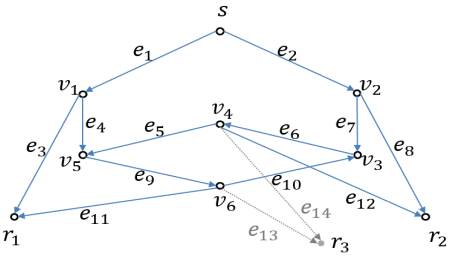

FIGURE 4. The expanding networks of Fig.1.

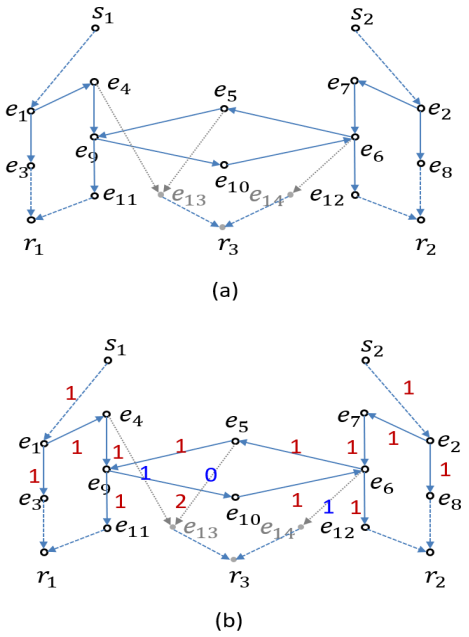

FIGURE 5. (a) The associated line graph of the expanding network in Fig.4. (b) The associated line graph of the expanding network in Fig.4, which sign the LEKs to construct a DI-$\mathbb{F}$-CNC in Fig.4.

ing to be changed or assigned in a practical extending network is generally small. If we use the existing DI-$\mathbb{F}$-CNC construction algorithm in [22] on the expanding network, one firstly should convert the cyclic extending network to a new corresponding equivalent acyclic one, and then apply the acyclic algorithm of linear multicast in [23] on the new acyclic network, so the entire code may be reconstructed.

D. THE THEORETICAL GUARANTEE FOR ALGORITHM IMPLEMENTATION

This subsection provides the theoretical guarantees for Algorithm 1.

As we have mentioned in Algorithm 1, when $\det(I-K)=0$ with the current LEK l, we need to replace it with a new value. Theorem 1 ensure that if $\det(I-K)=0$ with LEK l, then $\det(I-K)\neq 0$ with LEK l', where $l'\neq l$.

Theorem 1: In the line graph $\mathcal{N}$ of a given network N, assume that $(e,\bar{e})$ is any adjacent pair with the LEK $l_{e,\bar{e}}$, K is the current LEKs matrix and $\det(I-K)\neq 0$. We change $l_{e,\bar{e}}$ to a new LEK $l'_{e,\bar{e}}$, and define the associated LEKs matrix by K'. If $\det(I-K')=0$, then any other new LEK $l''_{e,\bar{e}}$ will make its associated LEKs matrix K'' holding that $\det(I-K'')\neq 0$.

Proof: Expand $\det(I-K)$ in terms of the row (or column) that the element $l_{e,\bar{e}}$ lies in. Then we can get

$$\det(I-K)=l_{e,\bar{e}}A^*_{e,\bar{e}}+B, \tag{5}$$

where $A^*_{e,\bar{e}}$ is the cofactor of element $l_{e,\bar{e}}$, and B is the sum of the cofactor expansions of other elements in the same row (or column) with $l_{e,\bar{e}}$.

In a similar way, we have

$$\det(I-K')=l'_{e,\bar{e}}A^*_{e,\bar{e}}+B, \tag{6}$$

$$\det(I-K'')=l''_{e,\bar{e}}A^*_{e,\bar{e}}+B. \tag{7}$$

By assumption, $\det(I-K)\neq 0$. If $\det(I-K')=0$, we say that $\det(I-K'')\neq 0$. Otherwise, assume to the contrary that

$$\det(I-K')=\det(I-K'')=0, \tag{8}$$

substituting it into (6) and (7), we can obtain that

$$A^*_{e,\bar{e}}=0,\ B=0. \tag{9}$$

After substituting (9) into (5), we have $\det(I-K)=0$, which contradicts to our assumption. Thus, Theorem 1 holds. □

The following Theorem shows that we can always pick an eligible LEK to maintain the transfer function condition.

Theorem 2: Let e be any node in the line graph $\mathcal{N}$ of the original network N, and $S(e)$ be the set of all successors of the node e. Assume that $\bar{e}\in S(e)$ is a successor of the node e, T_{ee} is the the total transfer function from e to e with the LEK $l_{e,\bar{e}}$, and $T_{ee}\neq 1$. We change $l_{e,\bar{e}}$ to a new LEK $l'_{e,\bar{e}}$, and define the associated transfer function by T'_{ee}. If $T'_{ee}=1$, then any other new LEK $l''_{e,\bar{e}}$ will make its associated transfer function T''_{ee} holding that $T''_{ee}\neq 1$.

Proof: Let T_1 be the transfer function from e to e, which traces $S(e)\backslash\bar{e}$. And $l_{e,\bar{e}}T_2$ represents the transfer function from e to e, which traces $\bar{e}$. Then we have

$$T_{ee}=T_1+l_{e,\bar{e}}T_2. \tag{10}$$

Similarly, we obtain that

$$T'_{ee}=T_1+l'_{e,\bar{e}}T_2, \tag{11}$$

$$T''_{ee}=T_1+l''_{e,\bar{e}}T_2. \tag{12}$$

By assumption, $T_{ee}\neq 1$. If $T'_{ee}=1$, we say that $T''_{ee}\neq 1$. Otherwise, assume to the contrary that

$$T'_{ee}=T''_{ee}=1, \tag{13}$$

substituting it into (11) and (12), we can obtain that

$$T_1=1,\ T_2=0. \tag{14}$$

After substituting (14) into (10), we have $T_{ee}=1$, which contradicts to our assumption. This completes the proof. □

In order to derive the the regularity condition of the set of PEKs in Algorithm 1, we need the following Lemma.

Lemma 1: In the line graph $\mathcal{N}$ of a given network N, assume that $E=\{e_1,\cdots,e_i,\cdots,e_\omega\}$ is a set of nodes in $\mathcal{N}$, and $W=\{\mathbf{w}(e_1),\cdots,\mathbf{w}(e_i),\cdots,\mathbf{w}(e_\omega)\}$ is the set of GEKs (or PEKs) of the associated nodes in E. Suppose that for a picked index i, $T_{e_ie_i}\neq 1$ with the current LEKs, where $T_{e_ie_i}$ is the transfer function from node e_i to e_i. Let $\widetilde{W}=\{\widetilde{\mathbf{w}}(e_1),\cdots,\widetilde{\mathbf{w}}(e_i),\cdots,\widetilde{\mathbf{w}}(e_\omega)\}$ be the set of GEKs (or PEKs) of the associated nodes in E, in which $l_{e_i,e}=0$ for any node $e\in\mathcal{N}$. Then the vectors $\mathbf{w}(e_i)$, $i=1,2,\cdots,\omega$, in W have the same linearly independence as the vectors $\widetilde{\mathbf{w}}(e_i)$, $i=1,2,\cdots,\omega$, in $\widetilde{W}$.

Proof: By the constraint relation (i) in Definition 1, it is not difficult to get the following expressions,

$$\mathbf{w}(e_i)=\widetilde{\mathbf{w}}(e_i)+T_{e_ie_i}\mathbf{w}(e_i), \tag{15}$$

$$\mathbf{w}(e_j) = \widetilde{\mathbf{w}}(e_j) + T_{e_ie_j}\mathbf{w}(e_i), \tag{16}$$

where $i \neq j$, $i, j \in \{1, 2, \cdots, \omega\}$, and $T_{e_ie_j}$ is the transfer function from e_i to e_j. Rearranging (15) and (16), we have

$$\widetilde{\mathbf{w}}(e_i) = (1 - T_{e_ie_i})\mathbf{w}(e_i), \tag{17}$$

$$\widetilde{\mathbf{w}}(e_j) = \mathbf{w}(e_j) - T_{e_ie_j}\mathbf{w}(e_i), \tag{18}$$

where $i \neq j$, $i, j \in \{1, 2, \cdots, \omega\}$. The matrix form of the above expressions (17) and (18) is as follows

$$\begin{pmatrix} \widetilde{\mathbf{w}}(e_1) \\ \widetilde{\mathbf{w}}(e_2) \\ \vdots \\ \widetilde{\mathbf{w}}(e_i) \\ \vdots \\ \widetilde{\mathbf{w}}(e_\omega) \end{pmatrix} = \begin{pmatrix} 1 & 0 & \cdots & -T_{e_ie_1} & \cdots & 0 \\ 0 & 1 & \cdots & -T_{e_ie_2} & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & 1 - T_{e_ie_i} & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & -T_{e_ie_\omega} & \cdots & 1 \end{pmatrix} \cdot \begin{pmatrix} \mathbf{w}(e_1) \\ \mathbf{w}(e_2) \\ \vdots \\ \mathbf{w}(e_i) \\ \vdots \\ \mathbf{w}(e_\omega) \end{pmatrix}. \tag{19}$$

Therefore, the vectors $\mathbf{w}(e_i)$, $i = 1, 2, \cdots, \omega$, have the same linearly independence as the vectors $\widetilde{\mathbf{w}}(e_i)$, $i = 1, 2, \cdots, \omega$, since the determinant of the $\omega \times \omega$ matrix in the right hand of (19) is equal to $1 - T_{e_ie_i} \neq 0$. □

Let the symbols be the same as before, we can get the following Theorem about the regularity condition of the set of PEKs.

Theorem 3: In a line graph $\mathcal{N}$, let $E = \{e_1, \cdots, e_i, \cdots, e_\omega\}$ and $\overline{E} = \{e_1, \cdots, \overline{e}_i, \cdots, e_\omega\}$ be two sets of ω nodes, where $\overline{e}_i$ is a successor of e_i. Suppose that $\text{rank}(\text{span}\{U^C\}) = \omega$ and $\text{rank}(\text{span}\{\overline{U}^C\}) \neq \omega$, where U^C, $\overline{U}^C$ are the set of PEKs of the associated ω nodes in E and $\overline{E}$, with the current LEK $l^C_{e_i,\overline{e}_i}$, respectively. And the transfer function from e_i to e_i with the current LEK $l^C_{e_i,\overline{e}_i}$ holding that $T^C_{e_ie_i} \neq 1$. Then after changing $l^C_{e_i,\overline{e}_i}$ to a new LEK $l^N_{e_i,\overline{e}_i}$ such that $T^N_{e_ie_i} \neq 1$, we have $\text{rank}(\text{span}\{\overline{U}^N\}) = \omega$, where $T^N_{e_ie_i}$ and $\overline{U}^N$ are the transfer function from e_i to e_i, and the PEKs of the associated ω nodes in $\overline{E}$, with the new LEK $l^N_{e_i,\overline{e}_i}$, respectively.

Proof: By the definition of PEK, we denote U^C, $\overline{U}^C$ and $\overline{U}^N$ as follows,

$U^C = \{\mathbf{u}^C(e_1), \cdots, \mathbf{u}^C(e_i), \cdots, \mathbf{u}^C(e_\omega)\}$, with the LEK $l^C_{e_i,\overline{e}_i}$ and $B^{e_i} = 0$,

$\overline{U}^C = \{\overline{\mathbf{u}}^C(e_1), \cdots, \overline{\mathbf{u}}^C(\overline{e}_i), \cdots, \overline{\mathbf{u}}^C(e_\omega)\}$, with the LEK $l^C_{e_i,\overline{e}_i}$ and $B^{\overline{e}_i} = 0$,

$\overline{U}^N = \{\overline{\mathbf{u}}^N(e_1), \cdots, \overline{\mathbf{u}}^N(\overline{e}_i), \cdots, \overline{\mathbf{u}}^N(e_\omega)\}$, with the LEK $l^N_{e_i,\overline{e}_i}$ and $B^{\overline{e}_i} = 0$.

By the conditions in Theorem 3, we know that the PEKs in set U^C are linearly independent, and the PEKs in set $\overline{U}^C$ are linearly dependent. And we need to prove that the PEKs in set $\overline{U}^N$ are linearly independent. Let

$\widetilde{U}^C = \{\widetilde{\mathbf{u}}^C(e_1), \cdots, \widetilde{\mathbf{u}}^C(\overline{e}_i), \cdots, \widetilde{\mathbf{u}}^C(e_\omega)\}$, with the LEK $l^C_{e_i,\overline{e}_i}$ and $B^{e_i} = 0$,

$\widetilde{U}^N = \{\widetilde{\mathbf{u}}^N(e_1), \cdots, \widetilde{\mathbf{u}}^N(\overline{e}_i), \cdots, \widetilde{\mathbf{u}}^N(e_\omega)\}$, with the LEK $l^N_{e_i,\overline{e}_i}$ and $B^{e_i} = 0$.

By Lemma 1, the PEKs in set $\widetilde{U}^C$ are linearly dependent just like the PEKs in set U^C, and the PEKs in set $\widetilde{U}^N$ have the same linear dependence with the PEKs in set $\overline{U}^N$.

Note that $B^{e_i} = 0$ for sets U^C, $\widetilde{U}^C$ and $\widetilde{U}^N$. Thus, when we change the LEK of the adjacent pair $(e_i, \overline{e}_i)$ from $l^C_{e_i,\overline{e}_i}$ to $l^N_{e_i,\overline{e}_i}$, it does not alter the PEKs of nodes $e_1, \cdots, e_{i-1}, e_{i+1}, \cdots, e_\omega$ in sets U^C, $\widetilde{U}^C$ and $\widetilde{U}^N$. In other words, we have

$$U^C \backslash \mathbf{u}^C(e_i) = \widetilde{U}^C \backslash \widetilde{\mathbf{u}}^C(\overline{e}_i) = \widetilde{U}^N \backslash \widetilde{\mathbf{u}}^N(\overline{e}_i). \tag{20}$$

By the constraint relation (i) in Definition 1, it is not difficult to obtain that

$$\widetilde{\mathbf{u}}^N(\overline{e}_i) - \widetilde{\mathbf{u}}^C(\overline{e}_i) = (l^N_{e_i,\overline{e}_i} - l^C_{e_i,\overline{e}_i})\mathbf{u}^C(e_i). \tag{21}$$

Since the PEKs in $\widetilde{U}^C$ are linearly dependent, we have

$$\widetilde{\mathbf{u}}^C(\overline{e}_i) \in \text{span}\{\widetilde{U}^C \backslash \widetilde{\mathbf{u}}^C(\overline{e}_i)\} = \text{span}\{U^C \backslash \mathbf{u}^C(e_i)\}. \tag{22}$$

Thus, $\widetilde{\mathbf{u}}^C(\overline{e}_i)$ can be linearly represented by $\mathbf{u}^C(e_1)$, $\cdots$, $\mathbf{u}^C(e_{i-1})$, $\mathbf{u}^C(e_{i+1})$, $\cdots$, $\mathbf{u}^C(e_\omega)$, namely,

$$\widetilde{\mathbf{u}}^C(\overline{e}_i) = \sum_{j=1, j\neq i}^{\omega} a_j \mathbf{u}^C(e_j), \tag{23}$$

where $a_j \in \mathbb{F}, j = 1, \cdots, i-1, i+1, \cdots, \omega$. If

$$\widetilde{\mathbf{u}}^N(\overline{e}_i) \in \text{span}\{\widetilde{U}^N \backslash \widetilde{\mathbf{u}}^N(\overline{e}_i)\} = \text{span}\{U^C \backslash \mathbf{u}^C(e_i)\}, \tag{24}$$

namely, $\widetilde{\mathbf{u}}^N(\overline{e}_i)$ also can be determined by a linear combination of $\mathbf{u}^C(e_1)$, $\cdots$, $\mathbf{u}^C(e_{i-1})$, $\mathbf{u}^C(e_{i+1})$, $\cdots$, $\mathbf{u}^C(e_\omega)$ as follows,

$$\widetilde{\mathbf{u}}^N(\overline{e}_i) = \sum_{j=1, j\neq i}^{\omega} b_j \mathbf{u}^C(e_j), \tag{25}$$

where $b_j \in \mathbb{F}, j = 1, \cdots, i-1, i+1, \cdots, \omega$. Notice that $\mathbf{u}^C(e_1)$, $\cdots$, $\mathbf{u}^C(e_i)$, $\cdots$, $\mathbf{u}^C(e_\omega)$ are linearly independent, after substituting (23) and (25) into (21), we can get

$$a_j = b_j,\ j \in \{1, \cdots, i-1, i+1, \cdots, \omega\},\ l^N_{e_i,\overline{e}_i} = l^C_{e_i,\overline{e}_i}, \tag{26}$$

which contradicts to the assumption that $l^N_{e_i,\overline{e}_i} \neq l^C_{e_i,\overline{e}_i}$. Therefore, $\widetilde{\mathbf{u}}^N(\overline{e}_i) \notin \text{span}\{\widetilde{U}^N \backslash \widetilde{\mathbf{u}}^N(\overline{e}_i)\}$, namely, the PEKs in set $\widetilde{U}^N$ are linearly independent, and so do the PEKs in set $\overline{U}^N$. □

Now, we discuss the regularity condition for the bases of sink nodes.

Theorem 4: In the line graph $\mathcal{N}$ of a given network N, we denote $\xi = \{e_1, e_2, \cdots, e_\omega\}$ be any regular basis of sink node, which has been determined before. For any node e in $\mathcal{N}$, denote $\overline{e}$ be a successor of the node e. Let the current LEK between the node e and the node $\overline{e}$ be $l^C_{e,\overline{e}}$ and the transfer function $T^C_{ee} \neq 1$ with the current LEK $l^C_{e,\overline{e}}$. We change $l^C_{e,\overline{e}}$ into any new value $l^N_{e,\overline{e}}$ such that the associated transfer function $T^N_{ee} \neq 1$, then at most one value of the $l^N_{e,\overline{e}}$ will make the basis ξ not be regular.

Proof: Let $S(e)$ be the set of all successors of the node e, then $\bar{e}\in S(e)$. Thus we have

$$T_{ee}^{C} = T_1 + l_{e,\bar{e}}^{C} T_2, \tag{27}$$

$$T_{ee}^{N} = T_1 + l_{e,\bar{e}}^{N} T_2, \tag{28}$$

where T_1 is the transfer function from e to e, which traces $S(e)\backslash\bar{e}$, and $l_{e,\bar{e}}^{C}T_2$ ($l_{e,\bar{e}}^{N}T_2$) represents the associated transfer function from e to e with LEK $l_{e,\bar{e}}^{C}$ ($l_{e,\bar{e}}^{N}$), which traces $\bar{e}$.

Let $\mathbf{g}_e^C$, $\mathbf{g}_e^N$ be the GEKs of node e with current LEK $l_{e,\bar{e}}^{C}$ and new LEK $l_{e,\bar{e}}^{N}$ respectively. And denote $\bar{\mathbf{g}}_e$ be the GEK of the node e, when $l_{e,\bar{e}} = 0$ for all $\bar{e} \in S(e)$.

Thus, by the constraint relation (i) in Definition 1, we have

$$\mathbf{g}_e^C = T_{ee}^{C}\mathbf{g}_e^C + \bar{\mathbf{g}}_e = (T_1 + l_{e,\bar{e}}^{C}T_2)\mathbf{g}_e^C + \bar{\mathbf{g}}_e, \tag{29}$$

and

$$\mathbf{g}_e^N = T_{ee}^{N}\mathbf{g}_e^N + \bar{\mathbf{g}}_e = (T_1 + l_{e,\bar{e}}^{N}T_2)\mathbf{g}_e^N + \bar{\mathbf{g}}_e. \tag{30}$$

Note that $T_{ee}^{C} \neq 1$ and $T_{ee}^{N} \neq 1$, rearranging (29) and (30), we have

$$\mathbf{g}_e^C = \frac{\bar{\mathbf{g}}_e}{1 - T_1 - l_{e,\bar{e}}^{C}T_2}, \tag{31}$$

$$\mathbf{g}_e^N = \frac{\bar{\mathbf{g}}_e}{1 - T_1 - l_{e,\bar{e}}^{N}T_2}. \tag{32}$$

Again, by the constraint relation (i) in Definition 1, we can obtain the following expression,

$$\mathbf{g}_{e_i}^N - \mathbf{g}_{e_i}^C = (R_{1i} + l_{e,\bar{e}}^{N}R_{2i})\mathbf{g}_e^N - (R_{1i} + l_{e,\bar{e}}^{C}R_{2i})\mathbf{g}_e^C, \tag{33}$$

where R_{1i} is the transfer function from e to e_i tracing the nodes in $S(e)\backslash\bar{e}$, $l_{e,\bar{e}}^{C}R_{2i}$ ($l_{e,\bar{e}}^{N}R_{2i}$) is the transfer function from $\bar{e}$ to e_i with LEK $l_{e,\bar{e}}^{C}$ ($l_{e,\bar{e}}^{C}$) tracing the node $\bar{e}$, and $i = 1, \cdots, \omega$. Substituting (31) and (32) into (33), we can get

$$\mathbf{g}_{e_i}^N - \mathbf{g}_{e_i}^C = X(l^N, l^C)R_i\bar{\mathbf{g}}_e, \tag{34}$$

where

$$X(l^N, l^C) = \frac{l_{e,\bar{e}}^{N} - l_{e,\bar{e}}^{C}}{(1 - T_1 - l_{e,\bar{e}}^{N}T_2)(1 - T_1 - l_{e,\bar{e}}^{C}T_2)}, \tag{35}$$

and

$$R_i = R_{2i} + T_2R_{1i} - T_1R_{2i}. \tag{36}$$

By assumption, the current GEKs $\mathbf{g}_{e_1}^C, \cdots, \mathbf{g}_{e_i}^C, \cdots, \mathbf{g}_{e_\omega}^C$ of basis ξ are linearly independent with LEK $l_{e,\bar{e}}^{C}$. Therefore, $\bar{\mathbf{g}}_e$ can be linearly expressed by $\mathbf{g}_{e_i}^C$, $i = 1, \cdots, \omega$. That is,

$$\bar{\mathbf{g}}_e = \alpha_1\mathbf{g}_{e_1}^C + \cdots + \alpha_i\mathbf{g}_{e_i}^C + \cdots + \alpha_\omega\mathbf{g}_{e_\omega}^C. \tag{37}$$

By (34) and (37) (for succinctness, denote $X(l^N, l^C)$ by X), we have

$$\begin{aligned}\mathbf{g}_{e_i}^N &= \mathbf{g}_{e_i}^C + XR_i\bar{\mathbf{g}}_e \\ &= XR_i\alpha_1\mathbf{g}_{e_1}^C + \cdots + (1 + XR_i\alpha_i)\mathbf{g}_{e_i}^C + \cdots + XR_i\alpha_\omega\mathbf{g}_{e_\omega}^C,\end{aligned} \tag{38}$$

where $i = 1, 2, \cdots, \omega$. (38) can be written in matrix form as

$$\begin{aligned}&(\mathbf{g}_{e_1}^N, \cdots, \mathbf{g}_{e_i}^N, \cdots, \mathbf{g}_{e_\omega}^N)^\top \\ &= \begin{pmatrix} 1+XR_1\alpha_1 & \cdots & XR_1\alpha_i & \cdots & XR_1\alpha_\omega \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ XR_i\alpha_1 & \cdots & 1+XR_i\alpha_i & \cdots & XR_i\alpha_\omega \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ XR_\omega\alpha_1 & \cdots & XR_\omega\alpha_i & \cdots & 1+XR_\omega\alpha_\omega \end{pmatrix} \\ &\quad \cdot \begin{pmatrix} \mathbf{g}_{e_1}^C \\ \vdots \\ \mathbf{g}_{e_i}^C \\ \vdots \\ \mathbf{g}_{e_\omega}^C \end{pmatrix}.\end{aligned} \tag{39}$$

If $\alpha_i = 0$ for all $i \in \{1, 2, \cdots, \omega\}$, then we have $\mathbf{g}_{e_i}^N = \mathbf{g}_{e_i}^C$, $i = 1, 2, \cdots, \omega$. That is, the basis ξ is regular with GEKs $\mathbf{g}_{e_1}^N, \cdots, \mathbf{g}_{e_i}^N, \cdots, \mathbf{g}_{e_\omega}^N$.

If there exists $i \in \{1, 2, \cdots, \omega\}$ such that $\alpha_i \neq 0$, without loss of generality, we assume that $\alpha_1 \neq 0$. Denote the $\omega \times \omega$ matrix in the right hand of (39) by M. Applying elementary column operations (replace the jth column by the sum of that column and $-\frac{\alpha_j}{\alpha_1}$ multiple of the 1st column, and then replace the 1st column by the sum of that column and $-XR_j\alpha_1$ multiple of the jth column, where $j = 2, \cdots, \omega$) on matrix M, we can get an equivalent matrix $\widetilde{M}$ of M,

$$\begin{aligned}&M \sim \widetilde{M} \\ &= \begin{pmatrix} 1+X\sum_{i=1}^{\omega} R_i\alpha_i & \cdots & -\alpha_i/\alpha_1 & \cdots & \alpha_\omega/\alpha_1 \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ 0 & \cdots & 0 & \cdots & 1 \end{pmatrix}.\end{aligned} \tag{40}$$

Thus,

$$\det(\widetilde{M}) = 1+X\sum_{i=1}^{\omega} R_i\alpha_i. \tag{41}$$

If $\det(\widetilde{M}) \neq 0$, then $\mathrm{rank}(M) = \mathrm{rank}(\widetilde{M}) = \omega$. In this case, in terms of (38), the GEKs $\mathbf{g}_{e_1}^N, \cdots, \mathbf{g}_{e_i}^N, \cdots, \mathbf{g}_{e_\omega}^N$ are linearly independent just as the GEKs $\mathbf{g}_{e_1}^C, \cdots, \mathbf{g}_{e_i}^C, \cdots, \mathbf{g}_{e_\omega}^C$.

If $\det(\widetilde{M}) = 0$, substituting (35) into (41), we can get the following expression

$$(1 - T_1 - l_{e,\bar{e}}^{N}T_2)(1 - T_1 - l_{e,\bar{e}}^{C}T_2) + (l_{e,\bar{e}}^{N} - l_{e,\bar{e}}^{C})R = 0, \tag{42}$$

where $R = \sum_{i=1}^{\omega} R_i\alpha_i$. Rearranging (42), we can get

$$\begin{aligned}&(1-T_1)^2+(T_1T_2-T_2-R)l_{e,\bar{e}}^{C}+(T_1T_2-T_2-R+T_2^2l_{e,\bar{e}}^{C})l_{e,\bar{e}}^{N} \\ &\qquad =0.\end{aligned} \tag{43}$$

We say that $T_1T_2-T_2-R+T_2^2l^C_{e,\bar{e}} \neq 0$. Otherwise, assume to the contrary that

$$T_1T_2-T_2-R+T_2^2l^C_{e,\bar{e}} = 0. \tag{44}$$

It is easy to get the value of R,

$$R = T_2-T_1T_2-T_2^2l^C_{e,\bar{e}}. \tag{45}$$

Substituting (45) into (43) and rearranging it, we have

$$(1 - T_1 - l^C_{e,\bar{e}}T_2)^2 = 0. \tag{46}$$

Thus,

$$1 - T_1 - l^C_{e,\bar{e}}T_2 = 0. \tag{47}$$

Substituting (47) into (42), we have $(l^N_{e,\bar{e}} - l^C_{e,\bar{e}})R = 0$. Since $R \neq 0$ (otherwise, $\det(\widetilde{M}) = 1 \neq 0$), we have $l^N_{e,\bar{e}} = l^C_{e,\bar{e}}$, which contradicts to our assumption. Therefore, $T_1T_2-T_2-R+T_2^2l^C_{e,\bar{e}} \neq 0$.

Now, in terms of (43), we can get

$$l^N_{e,\bar{e}} = \frac{(1-T_1)^2+(T_1T_2-T_2-R)l^C_{e,\bar{e}}}{T_1T_2-T_2-R+T_2^2l^C_{e,\bar{e}}}. \tag{48}$$

This demonstrates that there exists at most one $l^N_{e,\bar{e}}$ with the exact expression (48), such that $\det(\widetilde{M}) = 0$, namely, only in this case, the basis ξ with the new LEK $l^N_{e,\bar{e}}$ is not regular.

Thus, we finish the proof of this theorem. □

Remark: By the construction algorithm of DI-$\mathbb{F}$-CNC, we must determine all LEKs such that every basis of the sink node is regular. As we have mentioned in Algorithm 1, when (1) $\det(I - K) = 0$, or (2) $T_{e^i_k,e^i_k} = 1$, or (3) $\text{rank}(\text{span}\{\overline{U}^C_i\}) \neq \omega$ with $l^C_{e^i_k,\bar{e}^i_k}$, we need to replace the current coefficient $l^C_{e^i_k,\bar{e}^i_k}$ with a new value $l^N_{e^i_k,\bar{e}^i_k}$. By Theorems 1-3, we know that if $|\widetilde{\mathbb{F}}| \geq 4$, then we can select an eligible LEK from $\widetilde{\mathbb{F}}$. Also notice that the new picked LEK may change the regularity of previously determined bases of sink nodes on cyclic network. Thus, there is a possibility that we can not find an eligible LEK, such that the choosing procedures success. However, Theorem 4 can ensure that there exists at most one bad LEK such that the already determined basis is not regular. Also note that if we substitute $l^C_{e^i_k,\bar{e}^i_k}$ with $l^N_{e^i_k,\bar{e}^i_k}$, the bases ξ_j of sink node r_j, $j = 1, \cdots, i-1$ may be effected. Thus, the number of bases to be checked is at most $\tau - 1$. Therefore, if we have $4+\tau - 1 = \tau + 3$ LEKs coefficients to be selected, in terms of Theorems 1-4, we can always pick an appropriate LEK for lines 14-16 in Algorithm 1.

We have theoretically shown that if the size of the symbol field is no smaller than $\tau + 3$, where τ is the number of sink nodes in the network, then we can always construct a DI-$\mathbb{F}$-CNC over a cyclic network. Intuitively, it is possible to reduce the required symbol field size for the existence of DI-$\mathbb{F}$-CNC over a cyclic network with special network topology. In what follows, we verify this intuitive conjecture through numerical results in MATLAB on a class of special random directed cyclic networks.

E. NUMERICAL RESULTS

We construct a class of special random directed cyclic networks in MATLAB. For a given number of nodes, we number all nodes, with source as node 1, and sinks as nodes with the largest numbers. A random adjacency matrix $A = (a_{ij})$ of the nodes is generated according to the following rules. The elements in the fist column of A are zeroes, which ensures source node has no incoming edges. The last few rows are zeroes which ensures each sink node has no outgoing edges. All the diagonal elements of A are zeroes, which ensures no self-loop in the network. Other elements of A are assigned 0 with probability 0.85, and 1 with probability 0.15, such allocations of edge connection ensure that the random directed network is not so intricate and cycles can exist with positive probabilities. We throw away instances that the number of edge-disjoint paths from source to any sink is less than 2 and the number of outgoing edges from source is greater than $\tau + 1$, so we can focus on the scenarios of data generating rate $2 \leq \omega \leq \tau + 1$ (τ is the number of sink nodes in the network).

Figs.6 and 7 depict the mean of the required field size for the existence of DI-$\mathbb{F}$-CNC in the aforementioned special random directed cyclic networks (each statistic data is based on 50 eligible random graphs). Fig.6 shows the case where the number of sinks is fixed to 2, and the number of nodes changes from 10 to 20 with network including k circles, $k \in \{1, 2, 3\}$. Fig.7 shows the case where the number of circles is fixed to 1, and the number of intermediate nodes changes from 8 to 16 with network including k sinks, $k \in \{3, 4, 5\}$.

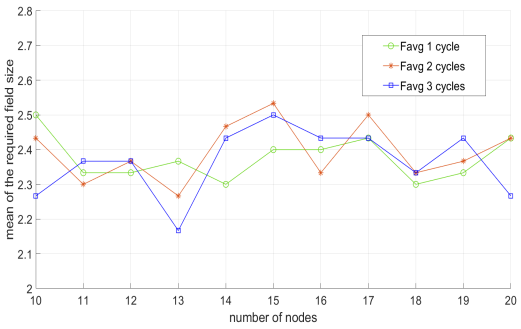

FIGURE 6. Mean of the required field size for the case where the number of sinks is fixed to 2, and the number of nodes changes from 10 to 20 with network including k circles, $k \in \{1, 2, 3\}$.

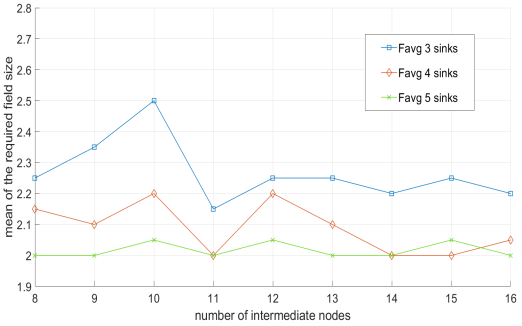

FIGURE 7. Mean of the required field size for the case where the number of circles is fixed to 1, and the number of intermediate nodes changes from 8 to 16 with network including k sinks, $k \in \{3, 4, 5\}$.

In both cases, the mean of the required field size is less than 3, and by numerical results, the practical field size for the existence of DI-$\mathbb{F}$-CNC is 2 or 3. These verify the intuitive conjecture that based on special network topology structure, the theoretically required field size $\tau + 3$ (τ is the number of sink nodes in the network) for the existence of DI-$\mathbb{F}$-CNC over a cyclic network can be reduced. Since the required symbol field size is closely related to the implementation of such coding constriction scheme in terms of computational complexity and storage requirement, we will investigate an improved lower bound on the required field size of DI-$\mathbb{F}$-CNC for the future work.

IV. CONCLUSION

In this paper, we investigate a straight construction algorithm for a field-based linear multicast network coding, which actually qualifies as a DI-$\mathbb{F}$-CNC over a cyclic network. We conclude that if the size of the symbol field is no smaller than $\tau + 3$, where τ is the number of sink nodes in the network, then we can always construct a DI-$\mathbb{F}$-CNC. This straight construction algorithm is beneficial to the scenario that the network is locally dynamic changing.

How to design DI-CNCs over cyclic networks with multiple sources is an interesting direction for future reasearch. Another intriguing work would be how to investigate the design and analysis of DI-CNCs over cyclic networks in the presence of noise or interception.

CONFLICTS OF INTEREST

The authors declare that they have no conflicts of interest.

REFERENCES

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.

[2] A. Cohen, R. G. L. D'Oliveira, S. Salamatian, and M. Medard, "Network coding-based post-quantum cryptography," *IEEE J. Sel. Areas Inf. Theory*, vol. 2, no. 1, pp. 49–64, Mar. 2021.

[3] U. Martinez-Penas and F. R. Kschischang, "Reliable and secure multishot network coding using linearized Reed–Solomon codes," *IEEE Trans. Inf. Theory*, vol. 65, no. 8, pp. 4785–4803, Aug. 2019.

[4] M. Tan, R. W. Yeung, S. T. Ho, and N. Cai, "A unified framework for linear network coding," *IEEE Trans. Inf. Theory*, vol. 57, no. 1, pp. 416–423, Jan. 2011.

[5] Q. T. Sun, "On construction of variable-rate and static linear network codes," *IEEE Access*, vol. 6, pp. 22249–22256, 2018.

[6] S. Y. R. Li and Q. T. Sun, "Network coding theory via communtative algebra," *IEEE Trans. Inf. Theory*, vol. 57, no. 1, pp. 403–415, Jan. 2011.

[7] S.-Y. R. Li, Q. T. Sun, and Z. Shao, "Linear network coding: Theory and algorithms," *Proc. IEEE*, vol. 99, no. 3, pp. 372–387, Mar. 2011.

[8] S. Y. R. Li and R. W. Yeung, "On convolutional network coding," in *Proc. IEEE Int. Symp. Inf. Theory*, Seattle, WA, USA, Jul. 2006, pp. 1743–1747.

[9] S.-Y.-R. Li and S. T. Ho, "Ring-theoretic foundation of convolutional network coding," in *Proc. 4th Workshop Netw. Coding, Theory Appl.*, Hong Kong, Jan. 2008, pp. 1–6.

[10] J. Huang, L. Wang, T. Zhang, and H. Li, "Unified construction algorithm of network coding in cyclic networks," in *Proc. 15th Asia–Pacific Conf. Commun.*, Wuhan, China, Oct. 2009, pp. 749–753.

[11] X. Zhao and W. Guo, "Equivalent conditions to determine the GEKs by the LEKs in a convolutional network code over a cyclic network," *IEICE Trans. Fundamentals Electron., Commun. Comput. Sci.*, vol. E95.A, no. 9, pp. 1570–1576, 2012.

[12] M. Lvov and H. H. Permuter, "Initialization algorithms for convolutional network coding," *IEEE Trans. Inf. Theory.*, vol. 64, no. 7, pp. 5277–5295, Jul. 2018.

[13] W. Guo, W. Zhang, W. Zhang, and B. Bai, "Convolutional network coding based on systematic RS code for low-latency guarantee," in *Proc. Comput., Commun. IoT Appl. (ComComAp)*, Nov. 2021, pp. 232–237.

[14] E. Erez and M. Feder, "Efficient network codes for cyclic networks," in *Proc. Int. Symp. Inf. Theory*, Adelaide, SA, Australia, Sep. 2005, pp. 1982–1986.

[15] A. I. Barbero and Y. Ytrehus, "Cycle-logical treatment for 'cyclopathic' networks," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2795–2804, Jun. 2006.

[16] E. Erez and M. Feder, "Efficient network code design for cyclic networks," *IEEE Trans. Inf. Theory*, vol. 56, no. 8, pp. 3862–3877, Aug. 2010.

[17] W. Guo, X. Shi, N. Cai, and M. Medard, "Localized dimension growth: A convolutional random network coding approach to managing memory and decoding delay," *IEEE Trans. Commun.*, vol. 61, no. 9, pp. 3894–3905, Sep. 2013.

[18] T. Abräo, L. Sampaio, S. Yang, K. T. K. Cheung, P. J. E. Jeszensky, and L. Hanzo, "Energy efficient OFDMA networks maintaining statistical QoS guarantees for delay-sensitive traffic," *IEEE Access*, vol. 4, pp. 774–791, 2016.

[19] Z. Ji, S. Cao, S. Wu, and W. Wang, "Delay-aware satellite-terrestrial backhauling for heterogeneous small cell networks," *IEEE Access*, vol. 8, pp. 112190–112202, 2020.

[20] D. Jiang, F. Wang, Z. Lv, S. Mumtaz, S. Al-Rubaye, A. Tsourdos, and O. Dobre, "QoE-aware efficient content distribution scheme for satellite-terrestrial networks," *IEEE Trans. Mobile Comput.*, early access, Apr. 22, 2021, doi: 10.1109/TMC.2021.3074917.

[21] I. Shrem, B. Grinboim, and O. Amrani, "Dynamic network-code design for satellite networks," 2022, *arXiv:2204.01433*.

[22] Q. T. Sun, S. Jaggi, and S.-Y.-R. Li, "Delay invariant convolutional network codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Petersburg, Russia, Jul. 2011, pp. 2492–2496.

[23] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. M. G. M. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Inf. Theory*, vol. 51, no. 6, pp. 1973–1982, Jun. 2005.

[24] R. W. Yeung, S.-Y. R. Li, N. Cai, and Z. Zhang, *Network Coding Theory*. Boston, MA, USA: Now Press, 2006.

[25] R. W. Yeung, *Information Theory and Network Coding*. Hong Kong: Springer, 2008.

[26] X. Zhao, "An efficient basic convolutional network code construction algorithm on cyclic networks," *AEU-Int. J. Electron. Commun.*, vol. 67, no. 12, pp. 1072–1078, Dec. 2013.

[27] H. C. Thomas, E. L. Charles, L. R. Ronald, and S. Clifford, *Introduction to Algorithms*. Cambridge, MA, USA: MIT Press, 2001.

[28] X. Zhao, X. Li, and T. Yan, "An improved independence test method for the convolutional multicast algorithm," *IEICE Trans. Fundamentals Electron., Commun. Comput. Sci.*, vol. E100.A, no. 9, pp. 2044–2047, 2017.

XUBO ZHAO received the M.S. degree in mathematics from Guangxi University, Nanning, in 2006, and the Ph.D. degree from the School of Telecommunications Engineering, Xidian University, Xi'an, China, in 2014. Her research interests include coding theory and network coding and their applications. She is the Reviewer of the journal IEEE TRANSACTIONS ON COMMUNICATIONS, *Finite Field and Their Applications*, and IEEE COMMUNICATIONS LETTERS.

XIAOPING LI received the M.S. degree in mathematics from Jiangxi Normal University, Nanchang, in 2004, and the Ph.D. degree in cryptography from Xidian University, Xi'an, China, in 2014. He is currently a Faculty Member with the College of Science, China University of Petroleum (East China), Qingdao, China. He has authored or coauthored around 20 publications, including refereed IEEE/IEICE/Springer/Elsevier journals and conference papers. His research interests include cryptography, coding theory, and network coding and their applications.