Deterministic Network Model Revisited: An Algebraic Network Coding Approach

Elona Erez, MinJi Kim, Student Member, IEEE, Yun Xu, Student Member, IEEE,
Edmund M. Yeh, Senior Member, IEEE, and Muriel Médard, Fellow, IEEE

Abstract—The capacity of multiuser networks has been a long-standing problem in information theory. Recently, Avestimehr et al. have proposed a deterministic network model to approximate multiuser wireless networks. This model, known as the ADT network model, takes into account the broadcast nature as well as the multiuser interference inherent in the wireless medium. For the types of connections we consider, we show that the results of Avestimehr et al. under the ADT model can be reinterpreted within the algebraic network coding framework. Using this framework, we propose an efficient distributed linear code construction for the deterministic wireless multicast relay network model. Unlike several previous coding schemes, we do not attempt to find flows in the network. Instead, for a layered network, we maintain an invariant where it is required that at each stage of the code construction, certain sets of codewords are linearly independent.

Index Terms—Network coding, deterministic network, algebraic coding, multicast, non-multicast, code construction.

I. INTRODUCTION

FINDING the capacity as well as the code construction for multi-user wireless networks are generally open problems. Even the relatively simple relay network with one source, one sink, and one relay, has not been fully characterized. There are two sources of disturbances in multi-user wireless networks – channel noise and interference among users in the network. In order to better approximate the Gaussian multi-user wireless networks, [4]–[6] proposed a binary linear deterministic network model (known as the ADT model), which takes into account the multi-user interference but not the noise. A node within the network receives the bit if the signal is above the noise level; multiple bits that simultaneously arrive at a node are superposed.

In essence, references [4]–[6] consider the high SNR regime, in which interference is the dominating factor. In the high SNR regime, the Cover-Wyner region may be well approximated by the combination of two regions, one square and one triangular, as in Figure 1. The square (shaded) part can be modeled as parallel links for the users, since they do not interfere. The triangular (unshaded) part can be considered as that of a time-division multiplexing (TDM) channel, which is equivalent to using noiseless finite-field additive MAC [9]. This result holds not only for binary field additive MAC, but also for higher field size additive MAC [9].

References [4]–[6] showed that, for a multicast connection where a single source wishes to transmit the same data to a set of destinations, the achievable rate is equal to the minimal cut between the source and any of the destinations. In addition, they showed that the minimal cut between the source and a destination is equal to the minimal rank of incidence matrices of all cuts between the two nodes. This can be viewed as the equivalent of the Min-cut Max-flow criterion for network coding in wireline networks [10], [11]. It has been shown that for several networks, the gap between the capacity of the deterministic ADT model and that of the corresponding Gaussian network is bounded by a constant number of bits, which does not depend on the specific channel fading parameters [4], [12], [13].

We make a connection between the ADT network and network coding – in particular, algebraic network coding introduced by Koetter and Médard [11]. Other approaches to operations in high SNR networks have been proposed [7].

1In high SNR, analog network coding, which allows/encourages strategic interference, is near optimal [7]. Analog network coding is a physical layer coding technique, introduced by [8], in which intermediate nodes amplify-and-forward the received signals without decoding. Thus, the nodes amplify not only the superposed signals from different transmitters but also the noise.
However, we do not compare these different approaches but rather build upon the given model proposed by [6]. We show that the multicast communication scenarios considered in the context of the ADT network model can be described within the algebraic network coding framework. The algebraic framework allows us to recover the same results as those in [4]–[6] using a different interpretation, and to provide further generalizations.\(^2\)

Using this insight, we propose an efficient linear code construction for multicasting in layered ADT networks that guarantees decodability, if such a code exists. Note that Avestimehr et al.’s proposed code construction is not guaranteed to be efficient and may potentially involve an infinite block length. Unlike several previous coding schemes [14]–[16], we do not attempt to find flows in the network. Instead, we maintain an invariant where it is required that at each stage of the code construction, certain sets of codewords are linearly independent. We assume that any node in the network can potentially be a destination. We design the code such that if the min-cut from the source to a certain node is at least the required rate, then the node will be able to reconstruct the data of the source using matrix inversion. In addition, when normalized by the number of sinks, our code construction has a complexity which is comparable to those of previous coding schemes for a single sink.

The paper is organized as follows. We provide a brief overview of the related literature in Section II. We present the network model and an algebraic formulation of the ADT network in Section III. In Section IV, we present our code construction for multicasting in ADT networks, and analyze its performance. In Section V, we briefly discuss possible extensions to ADT networks. Although this paper focuses on multicast connections in ADT networks, the algebraic formulation allows us to apply the results from network coding [11] to ADT networks. By doing so, we can provide new capacity characterizations of ADT networks with a more general set of traffic requirements and ADT networks with cycles and erasures. Finally, we conclude in Section VI.

II. BACKGROUND

Avestimehr et al. introduced the ADT network model to better approximate wireless networks [6]. In the same work, they characterized the capacity of the ADT networks to be the minimum of all cut capacities between two nodes. Reference [17] showed that, under some special assumptions, the min-cut can be characterized by an optimization over a submodular function; thus, in these special cases, the min-cut of an ADT network may be computed efficiently. Reference [6] generalized the Min-cut Max-flow theorem for graphs to ADT networks for single unicast/multicast connections.

It has been shown that for several networks, the ADT network model approximates the capacity of the corresponding Gaussian network to within a constant number of bits. For instance, [4] considered the single relay channel and the diamond network, and showed that the gap between the capacity of the ADT model and that of the Gaussian network is within 1 bit and 2 bits, respectively. Reference [12] considered many-to-one and one-to-many Gaussian interference networks. The networks in [12] are special cases of the interference network with multiple users, where the interference are either experienced (many-to-one) or caused by (one-to-many) a single user. It was shown that in these cases, the gap between the capacity of the Gaussian interference channel and the corresponding deterministic interference channel is again bounded by a constant number of bits. The work in [12] provided an alternative proof to [18] on the existence of a scheme that can achieve a constant gap from the capacity for all values of channel parameters. In [13], the half-duplex butterfly network was considered. They showed that the deterministic model approximates the symmetric Gaussian butterfly network to within a constant.

There has been significant interest in finding an efficient code construction algorithm for the ADT network model. In the case of unicast communication, a number of code constructions have been proposed. It is important to observe that in the code constructions for unicast communication, routing [15] or one-bit operations [14], [19] are sufficient for achieving the capacity of the deterministic model. Amaudruz and Fragouli [14] proposed an algorithm which can be viewed as an application of the Ford and Fulkerson flow construction to the deterministic model. The complexity of the algorithm was shown to be \(O(|V||E| R^3)\), where \(V\) is the set of nodes in the network, \(E\) is the set of edges, and \(R\) is the rate of the code. Reference [19] improved on the work of [14]. In [15], another algorithm for finding the flow for unicast networks was developed. The algorithm is based on an extension of the Rado-Hall transversal theorem for matroids and on Edmonds’ theorem. The transmission scheme in [15] extracts at each relay node a subset of the input vectors and sets the outputs to the same values as that subset. In [16], it was shown that the deterministic model can be viewed as a special case of a more abstract flow model, called linking network, which is based on linking systems and matroids. Using this approach, [16] achieved a code complexity \(O(\lambda N_{layer}^3 \log N_{layer})\), where \(\lambda\) is the number of layers in the layered network, and \(N_{layer}\) is the maximal number of nodes in a layer. Note that linear network coding is known to be matroidal [20]; thus, the fact that ADT networks are matroidal [16] is consistent with our result.

In the case of multicast communication, however, routing or one-bit operations may not be sufficient to achieve the capacity in the ADT model. This can be shown by considering the example in Figure 2, which is given in [21]–[23] for network coding. From the analysis for network coding, it follows that in the case of the deterministic model, the maximal rate 2 can be achieved simultaneously for all sinks only with an alphabet size which is at least 3. To see this, observe that to achieve rate 2 the source has to transmit at its outputs two statistically independent symbols \(x_1, x_2\). For node \(v_1^2\), \(1 \leq i \leq 4\), at the second layer, the transmitted symbol is a certain function of the symbols \(x_1, x_2\), given by \(y_1 = f_i(x_1, x_2)\). Node \(v_2^3\), \(1 \leq i \leq 4\), at the third layer transmits at its outputs two functions of \(y_i\),
given by $f_i^1(y_i), f_i^2(y_i)$. Sink $t_i, 1 \leq i \leq 6$, receives at its two inputs symbols of the form $f_i^1(y_j), f_i^2(y_j) + f_i^2(y_k)$ for some $1 \leq j \leq 4, 1 \leq k \leq 4, j \neq k$. It follows that without rate loss, we can always assume $f_i^1(y_j) = y_j$ for each $1 \leq j \leq 4$. In that case, the sink $t_i$ receives $y_j$ at its upper input and can therefore find $f_i^2(y_j)$ and eliminate it from its second received symbol. Thus, it is equivalent to the situation in which the sink receives $y_j, y_k$. This in turn is exactly the situation in [22] (Theorem 3.1) for network coding. Since the channels are all binary in the deterministic model, it follows that the minimal required alphabet size is in fact $2^2 = 4$, and therefore the minimal vector length is $\log_2(4) = 2$. Thus, for multicasting in ADT networks, we need to either operate in a larger field, $\mathbb{F}_q, q \geq 2$, or use vector coding (or both).

References [24]–[26], independently from [1]–[3], proposed a polynomial time algorithm for multicasting in ADT networks. Note that the algorithm proposed in [24]–[26] focuses on minimizing the field size, and is therefore a centralized algorithm. The complexity of the algorithm was shown to be $O(|T| \alpha^2 \log_2 \alpha + \alpha^2 |\mathcal{E}|^2 + (|T| \log_2 |\mathcal{E}|)^2 \mu(S) \lambda^3)$ where $\alpha = |\mathcal{E}|^2 + \mu(S), \mu(S)$ is the number of source random processes to be transmitted, $T$ is the set of destinations, $\mathcal{E}$ is the set of edges, and $\lambda$ is the longest path between the source $S$ and a destination in $T$. In particular, [25] extended the algebraic network coding result [11] to vector network coding, and showed that constructing a valid vector code is equivalent to certain algebraic conditions. This result [25] is supported by the result from [27]. Reference [27] showed that codes in higher field size $\mathbb{F}_q$ can be mapped to binary-vector codes without loss in performance. This insight, combined with that of [11], suggests that an algebraic property of a scalar code may translate into another algebraic property of the corresponding vector code.

### III. Network Model

The ADT network model uses binary channels, and thus, binary additive MACs are used to model interference. Prior to [4] and [5], Effros et al. presented an additive MAC over a finite field $\mathbb{F}_q [28]$. The Min-cut Max-flow theorem holds for all the above cases. It may seem that the ADT network model differs greatly from that of [28] owing to the difference in the field sizes used. In general, codes in $\mathbb{F}_q$ subsume binary codes, i.e., binary-vector codes in $(\mathbb{F}_2)^m$. However, for point-to-point links with memory (or equivalently by allowing nodes to code across time), we can convert a code in $(\mathbb{F}_2)^m$ to a code in a larger field $\mathbb{F}_q$ and vice versa by normalizing to an appropriate time unit. Therefore, our work in part shows an equivalence of higher field size codes and binary-vector codes in ADT networks.

We can achieve a higher field size in ADT networks in two ways. First, we can combine multiple binary channels. In other words, consider two nodes $V_1$ and $V_2$ with two binary channels connecting $V_1$ to $V_2$. Now, instead of considering them as two binary channels, we can “combine” the two channels as one with capacity of 2 bits. In this case, instead of using $\mathbb{F}_2$, we can use a larger field size of $\mathbb{F}_q$. Thus, selecting a larger field $\mathbb{F}_q, q > 2$, in an ADT network results in fewer but higher capacity parallel channels. Reference [27] also provides a conversion from a code in a larger field $\mathbb{F}_q$ to a binary-vector scheme in $\mathbb{F}_2^m$ where $q \leq 2^m$. Therefore, a solution in $\mathbb{F}_q$ may be converted back to a binary-vector scheme, which may be more appropriate for the original ADT model. Furthermore, it is known that to achieve capacity for multicast connections, $\mathbb{F}_2$ is not sufficient [2]; thus, we need to operate in a higher field size. Therefore, we shall not restrict ourselves to $\mathbb{F}_2$. Second, we can treat the code in blocks – i.e., we can increase the field size by processing bits across multiple time units. For example, consider two nodes $V_1$ and $V_2$ with one binary channel connecting them. If we wish to use a field size of $\mathbb{F}_4$, then we can do so by sending one symbol (2 bits) over two time units.

We now proceed to define the network model precisely. A wireless network is modeled using a directed graph $G = (\mathcal{V}, \mathcal{E})$ with a supernode set $\mathcal{V}$ and an edge set $\mathcal{E}$ as shown in Figure 3. A supernode $V \in \mathcal{V}$ is a node of the original network. We use the term supernode to emphasize the fact that supernode $V$ consists of input ports $I(V)$ and output ports $O(V)$, as shown in Figure 4. Let $S, T \subseteq \mathcal{V}$ be the set of source and destination supernodes. We assume that each supernode...
\[ V \in \mathcal{V} \] contains \( n \) input ports and \( n \) output ports, where
\[
    n = \left\lceil \frac{1}{2} \log \max_{(V_i, V_j) \in \mathcal{E}} \text{SNR}(V_i, V_j) \right\rceil. 
\]
(1)

Therefore, noise is embedded or hard-coded in the structure of the ADT network. The number of edges between two supernodes \( V_i \) and \( V_j \) represents the channel quality (equivalently, the noise) between the two supernodes, i.e., there are
\[
    \left\lceil \frac{1}{2} \log \text{SNR}(V_i, V_j) \right\rceil 
\]
edges between nodes \( V_i \) and \( V_j \).

We denote the input ports of supernode \( V \) by
\[
    I(V) = \{x^V_0, \ldots, x^V_n\}, \quad (3) 
\]
and the output ports by
\[
    O(V) = \{y^V_0, \ldots, y^V_n\}. \quad (4) 
\]

An edge \((y^V_0, x^V_i) \in \mathcal{E}\) may exist from an output port \( y^V_i \in O(V_i) \) to an input port \( x^V_0 \in I(V_0) \), for any \( V_1, V_2 \in \mathcal{V} \). All edges are of unit capacity, where capacity is normalized with respect to the symbol size of \( \mathbb{F}_q \).

A source supernode \( S \in \mathcal{S} \) has independent random processes \( \mathcal{X}(S) = \{X(S, 1), X(S, 2), \ldots, X(S, \mu(S))\} \), \( \mu(S) \leq \lfloor O(S) \rfloor \), which it wishes to communicate to a set of destination supernodes \( \mathcal{T}(S) \subseteq \mathcal{T} \). In other words, we want \( T \in \mathcal{T}(S) \) to replicate a subset of the random processes, denoted by \( \mathcal{X}(S, T) \subseteq \mathcal{X}(S) \), with the aid of the network. Note that the algebraic formulation is not restricted to multicast connections; different sources may wish to communicate to different subsets of destinations. We define a connection \( c \) as a triple \((S, T, \mathcal{X}(S, T))\), and the rate of \( c \) is defined as
\[ R(c) = \sum_{(S, i) \in \mathcal{X}(S, T)} H(X(S, i)) = |\mathcal{X}(S, T)| \text{ (symbols)}. \]

Information is transmitted through the network in the following manner. A supernode \( V \) sends information through \( y^V \in O(V) \) at a rate at most one symbol per time unit. Each supernode has coding vectors associated with it, where coding vector represents the coding coefficients used to generate the linear combination. Denote the coding vectors of the input ports of supernode \( V \) by
\[
    x^{(V)} = \{x^V_0, \ldots, x^V_n\}, \quad (5) 
\]
where \( x^V_i \in \mathbb{F}_q^n \) is the coding vector associated with the input port \( x^V_i \). Similarly, we denote the coding vectors of the output ports of supernode \( V \) by
\[
    y^{(V)} = \{y^V_0, \ldots, y^V_n\}, \quad (6) 
\]
where \( y^V_i \in \mathbb{F}_q^n \) is the coding vector associated with the output port \( y^V_i \). The coding vector \( y^V_i \) is given by
\[
    y^V_i = \sum_{i=1}^{n} \beta(\mathcal{X}(S, i)) x^V_i, \quad 1 \leq j \leq n, \quad (7) 
\]
where \( \beta(\mathcal{X}(S, i)) \in \mathbb{F}_q \) are the supernode internal coding coefficients. For a source supernode \( S \),
\[
    y^S = \sum_{i=1}^{n} \beta(\mathcal{X}(S, i)) x^S_i + \sum_{(X(S, i) \in \mathcal{X}(S))} a(i, y^V_i) X(S, i). \quad (8) 
\]
where \( a(i, y^V_i) \in \mathbb{F}_q \) are the source super node coding coefficients.

Finally, the destination \( T \) receives a collection of input processes \( y^T_i \) for \( i \in [1, n] \). Supernode \( T \) generates a set of random processes \( \mathcal{Z}(T) = \{Z(T, 1), Z(T, 2), \ldots, Z(T, v(T))\} \) where
\[
    Z(T, i) = \sum_{x^T_i \in \mathcal{I}(T)} \epsilon(x^T_i, T, y^T_i) x^T_i. \quad (9) 
\]

where the coefficients \( \epsilon(x^T_i, T, y^T_i) \) are elements of \( \mathbb{F}_q \).

An output port \( y^T_i \) of a supernode \( V \) sends the same symbol or broadcasts to the set of input ports \( x^T_j \) connected to it (i.e., an edge \((y^T_i, x^T_j) \in \mathcal{E}) \). In addition, superposition occurs at the input port \( x^T_i \), i.e.,
\[
    x^T_i = \sum_{(y^T_j, y^T'_j) \in \mathcal{E}} y^T_j. \quad (10) 
\]
A connection \( c = (S, T, \mathcal{X}(S, T)) \) is established successfully if \( \mathcal{X}(S) = \mathcal{Z}(T) \). For a given network \( G \) and a set of connections \( C \), we say that \((G, C)\) is solvable if it is possible to establish successfully all connections \( c \in C \).

A. An Interpretation of the Network Model

The ADT network model uses multiple channels from an output port to model broadcast. In Figure 3, there are two edges from output port \( y_i \) to input ports \( x^V_0 \) and \( x^V_1 \); however, due to the broadcast constraint, the two edges \((y_i, x^V_0)\) and \((y_i, x^V_1)\) carry the same information \( a \). This introduces considerable complexity in constructing a network code as well as computing the min-cut of the network \([4, 5, 14, 16]\). This is because multiple edges from a port do not capture the broadcast dependencies. Furthermore, the broadcast dependencies have to be propagated through the network.

In our approach, we remedy this situation by introducing the use of hyperedges, as shown in Figure 5. An output port’s decision to transmit affects the entire hyperedge; thus, the output port transmits to all the input ports connected to the hyperedge simultaneously.

B. Algebraic Network Coding Formulation

We define a system matrix \( M \) to describe the relationship between the source’s random processes \( \mathcal{X}(S) \) and the destinations’ processes \( \mathcal{Z} \). Thus, we want to characterize \( M \) where
\[
    \mathcal{Z} = \mathcal{X}(S) \cdot M. \quad (11) 
\]
The matrix \( M \) is composed of three matrices, \( A \), \( F \), and \( B \).

Given \( G \), we define the adjacency matrix \( F \) as follows:
\[
    F(p, q) = \begin{cases} 
    1 & \text{if } (p, q) = (y^V_i, x^V_j) \in \mathcal{E}, \\
    \beta(\mathcal{X}(S, i)) & \text{if } (p, q) = (x^V_i, y^V_j) \text{ for } V \in \mathcal{V}, \\
    0 & \text{otherwise.} 
    \end{cases} \quad (12) 
\]
Each entry of \( F \) represents the input-output relationships of the ports. A zero entry indicates that the ports are not directly
can show the following remark. In other words, using the above formulation, we directly applied to ADT networks, and it can be shown that results from \([11]\) can be used at \(T\) as follows:

\[
\begin{bmatrix}
\alpha_{(i,j)}^{(s)} & 0 & \cdots & 0 \\
0 & \alpha_{(j,k)}^{(s)} & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \alpha_{(t,v)}^{(s)}
\end{bmatrix}
\]

where \( \alpha_{(i,j)}^{(s)} \) are free variables, representing the coding coefficients used at \( V \) to map the input port processes to the output port processes.

Note that \( F^k \), the \( k \)-th power of an adjacency matrix of a graph \( G \), shows the existence of paths of length \( k \) between any two nodes in \( G \). Therefore, the series \( (I - F)^{-1} = I + F + F^2 + F^3 + \cdots \) represents the connectivity of the network.

Matrix \( A \) represents the encoding operations performed at \( S \). We define a \(|X(S)| \times |E|\) encoding matrix \( A \) as follows:

\[
A_{(i,q)} = \begin{cases} 
\alpha_{(i,j)}^{(s)} & \text{if } q = y_j^S \in O(S) \\
0 & \text{otherwise,}
\end{cases}
\]

Matrix \( B \) represents the decoding operations performed at the destination \( T \in T \). Since there are \(|T|\) destination nodes, \( B \) is a matrix of size \(|Z| \times |E|\) where \( Z \) is the set of random processes derived at the destination supernodes. We define the decoding matrix \( B \) as follows:

\[
B_{(T_j,k),q} = \begin{cases} 
\epsilon_{(i,j)}^{(T_j)} & \text{if } q = x_i^T \in I(T_j) \\
0 & \text{otherwise,}
\end{cases}
\]

Then, using similar argument as \([11]\), it can be shown the system matrix \( M \) is given by

\[
M = A(1 - F)^{-1}B^T,
\]

and the capacity of the given ADT network, equivalently the minimum value of all \( S - T \) cuts mincut \((S, T)\), is

\[
\text{mincut}(S, T) = \max_{a_{(i,j)}, \epsilon_{(i,j)}^{(T)}} \text{rank}(M).
\]

Given this algebraic formulation, results from \([11]\) can be directly applied to ADT networks, and it can be shown that linear operations are sufficient to achieve capacity for multicast connections. In other words, using the above formulation, we can show the following remark.

**Remark 3.1:** Given an acyclic network \( G \) with a single multicast connection \( c = (S, T, X(S, T)) \) of rate \( R(c) = |X(S, T)| \), the following are equivalent:

1. A connection \( c \) is feasible.
2. mincut \((S, T) \) \( \geq R(c) \) for all \( T \in T \).
3. There exists an assignment of \( \alpha_{(i,j)}^{(s)} \), \( \epsilon_{(i,j)}^{(T)} \), and \( \beta_{(i,j)}^{(s)} \) such that processes \( X(S, T) \) are recovered at all \( T \in T \).

Therefore, the algebraic framework provides yet another way to show the linearity result in \([4]\)–\([6]\), \([16]\) and \([14]\).

A direct consequence of this observation is that random linear network coding achieves capacity for multicast connections in ADT networks. However, random linear network coding guarantees decodability with high probability.

In this paper, we provide an efficient linear code construction for multicasting in layered ADT networks that guarantees decodability, if such a code exists. Note that Avestimehr et al.’s proposed code construction is not guaranteed to be efficient and may potentially involve an infinite block length. Unlike several previous coding schemes \([14]\)–\([16]\), we do not attempt to find flows in the network. Instead, we maintain an invariant where it is required that at each stage of the code construction, certain sets of codewords are linearly independent.

**IV. CODE CONSTRUCTION FOR MULTICAST CONNECTION**

Our construction can be viewed as a non-straightforward generalization of the algorithm in \([29]\) for the construction of linear codes for multicast wireline networks. Each sink receives on its incoming edges a linear transformation of the source. The generalization of the code construction to the ADT network model is not straightforward, due to the broadcast constraint and the interference constraint, which are embedded in the ADT network model.

In this section, we propose an efficient partially distributed code construction for a multicast connection in an ADT network, which guarantees that all destination supernodes can decode if the connection is feasible. This code construction requires only that there be some local coordination among neighboring supernodes.

We let

\[
R = \min_{T_i \in T} \text{mincut}(S, T_i).
\]
For the given multicast rate $R$, we define the set

$$\mathcal{V}(R) = \{ V \in \mathcal{V} \mid \text{mincut}(S, V) \geq R \}.$$  

(18)

A property of our code construction is that all supernodes in $\mathcal{V}(R)$ will be able to decode the data, including those that are not in the set of destination supernodes $T$. We also note that the code designer may be oblivious of the exact location of the nodes in $T$ or $\mathcal{V}(R)$.

The network is assumed to be layered with $\lambda$ layers, where all links are from layer $l$ to layer $l+1$ for $l = 1, \ldots, \lambda - 1$. The source is at layer 1. We assume that there are at most $N_{\text{layer}}$ nodes at any layer $l \in [1, \lambda]$. Since the network is acyclic, we can arrange all the ports in a topological order. The input ports of a certain supernode always precede the output ports of the same supernode. In addition, we adopt the convention that ports of supernodes in layer $l$ precede all the ports of supernodes in layer $l+1$, for $l = 1, \ldots, \lambda - 1$. We make the assumption that within a single layer, the supernodes are ordered from top to bottom. Also, within each supernode, ports are arranged from top to bottom.

We denote $P(x^V_l)$ to be the set of output ports that have links incoming into the input port $x^V_l$ of a supernode $V$ in layer $l$. By assumption, $0 \leq |P(x^V_l)| \leq N_{\text{layer}}$ i.e. there are at most $N_{\text{layer}}$ edges incoming to $x^V_l$ from output ports in layer $l-1$. Let $C(y^V_l)$ be the set of input ports in layer $l+1$ that have links incoming from output port $y^V_l$ of supernode $V$ in layer $l$. Note that $0 \leq |C(y^V_l)| \leq N_{\text{layer}}$ since $y^V_l$ may be adjacent to input ports of different supernodes in layer $l+1$, but may not be adjacent to more than one input port per supernode in layer $l+1$.

### A. Regular Sets and Virtual Sinks

For the algorithm we describe in the following sections, we will use an invariant which will be maintained throughout the algorithm. Prior to defining this invariant, we need to introduce the concepts of regular sets and virtual sinks.

Consider supernodes at a certain layer $l$. We consider a set $W$ of $R$ ports, where for any supernode $V$ the set $W$ may contain a subset of $V$’s output ports, or a subset of $V$’s input ports, but not both. The set $W$ may contain ports of several supernodes. Now consider the following. If $W$ contains output ports of supernode $V$ then we connect each of the output ports to a “virtual sink” $T(W)$, which is a supernode consisting of its own ports. If $W$ contains $p$ input ports of supernode $V$, then we disconnect all the input ports of supernode $V$ that are not in $W$. We connect the $p$ upper output ports of $V$ to the virtual sink $T(W)$. The order in which the output ports are connected to $T(W)$ is not important. For consistency, however, we assume that the output ports of layer $l$ that are connected to sink $T(W)$, are connected to the input ports of $T(W)$ from top to bottom, each output port to a different input port of $T(W)$. See Figure 7 for an illustration.

**Definition 4.1 (Regular Set):** The set $W$ is said to be regular if $\text{mincut}(S, T(W)) = R$.

In Figure 7, we show an example of $W$ where $R = 3$, and a virtual sink $T(W)$. The set $W$ is not regular since $T(W)$ receives only rate of 2.

The important property of regular sets, as we shall observe, is that there exists a network code such that the coding vectors of the regular sets are linearly independent. We shall exploit this property for our code construction.

### B. Overview of Coding Scheme

We proceed through the ports in the topological order, and for each port we reach, we choose the coding coefficients, taken from $\mathbb{F}_q$, where $q$ is the field size to be determined.

We refer to the coding operation in Equation (7) as “forward coding”. Once the coding vector $y^V_l$ of the output port is determined, we can multiply it by the coding coefficient $k_j$. We refer to this step as “virtual coding”. The “virtual coding” can be incorporated into the “forward coding”. However, we separate the coding into two distinct phases for purposes of presentation.

For an input port $x^V_l$, let $y_1, \ldots, y_p$ be the coding vectors of the output ports in the set $P(x^V_l)$. Then, the coding vector of the input port $x^V_l$ is given by

$$x^V_l = \sum_{j=1}^p k_j y_j,$$

(19)

where $k_j \in \mathbb{F}_q$ are the virtual coding coefficients. Note that only one coefficient $k_j$ is chosen for all ports in $C(y_j)$, which requires that the supernodes in the same layer coordinate locally when determining $k_j$’s.

**Definition 4.2 (Cut of Ports):** Consider the coding scheme, which assigns coding coefficients in a topological order, as mentioned above. Let $t$ be the time index. We denote $C_t = (\hat{S}, \hat{S}^c)$ to be the current cut of the algorithm, where $\hat{S}$ is the set of ports whose coding coefficients have been determined, and $\hat{S}^c$ the set of remaining ports. An output port $y^V_l$ is in
Fig. 8. An example of $Q_{C_l}$ (in black).
Since the input ports in \( I(V) \) will be replaced by \( p \) of the output ports from \( O(V) \). Without loss of generality, assume that the \( p \) ports in \( W \) that are also in \( I(V) \) are \( \{w_1, \ldots, w_p\} = \{x_1^V, \ldots, x_p^V\} \). We choose a set of size \( p \) from \( O(V) \) and denote the set by \( \{w_1', \ldots, w_p'\} \). There are \( \binom{n}{p} \) such possible sets.

We now update list \( L_i \) to \( L_{i+1} \). In \( L_{i+1} \), we replace \( W = \{w_1, \ldots, w_p\} \) with \( W' = \{w_1', \ldots, w_p', w_{p+1}, \ldots, w_R\} \). There are \( \binom{n}{p} \) subsets of this form in \( L_{i+1} \). In order for the invariant to be maintained, we require the coding vectors of all \( \binom{n}{p} \) new subsets to be simultaneously linearly independent.

**Lemma 4.3:** Consider a subset \( W \in L_i \), which contains \( p \geq 1 \) input ports from \( I(V) \). If the field size \( q \geq 2 \), then there exists a set of coding coefficients \( \beta_{(x_i^V, y_1^V)} \in \mathbb{F}_q \) for \( 1 \leq i \leq n \), \( 1 \leq j \leq n \) such that the coding vectors of a subset \( W' \in L_{i+1} \) are linearly independent.

**Proof:** The coding vectors of \( W' \) are

\[
W' = \left\{ \sum_{i=1}^{p} \beta_{x_i^V, y_1^V} x_i^V + v_1, \ldots, \sum_{i=1}^{p} \beta_{x_i^V, y_p^V} x_i^V + v_p, \right. \\
\left. w_{p+1}, \ldots, w_R \right\},
\]

where \( v_i, i = 1, \ldots, p, \) are the contributions of the coding vectors of the input ports in \( I(V) \setminus W \). The \( v_i \)'s are assumed to be fixed. Since \( W \in L_i \), invoking the inductive hypothesis, the set of vectors \( W = \{x_1^V, \ldots, x_p^V, w_{p+1}, \ldots, w_R\} \) is a basis.

We need to determine under which conditions the subset \( W' \) is also a basis.

Consider the equation

\[
\varphi_1 \left( \sum_{i=1}^{p} \beta_{x_i^V, y_1^V} x_i^V + v_1 \right) + \cdots + \varphi_p \left( \sum_{i=1}^{p} \beta_{x_i^V, y_p^V} x_i^V + v_p \right) + \varphi_{p+1} w_{p+1} + \cdots + \varphi_R w_R = 0. \tag{23}
\]

The set \( W \) is a basis if and only if \( \varphi_1 = \varphi_2 = \cdots = \varphi_R = 0 \) is the only solution to (23). In the following, we find a sufficient condition for \( \varphi_1 = \varphi_2 = \cdots = \varphi_p = 0 \) to be the only solution to (23). First, we express \( v_1 \) in the basis \( W \) as

\[
v_1 = \gamma_1, x_1^V + \cdots + \gamma_p, x_p^V + \gamma_{p+1}, w_{p+1} + \cdots + \gamma_R, w_R,
\]

where \( \gamma \) are not all zeros. Substituting and rearranging the terms of (23) yields

\[
\varphi_1 (\beta_{x_1^V, y_1^V} + \gamma_1, 1) + \cdots + \varphi_p (\beta_{x_p^V, y_p^V} + \gamma_1, 1) \sum_{i=1}^{p} \beta_{x_i^V, y_1^V} x_i^V + \cdots + \sum_{i=1}^{p} \beta_{x_i^V, y_p^V} x_i^V + \varphi_{p+1} \gamma_{p+1}, w_{p+1} + \cdots + \gamma_{p+1}, w_{p+1} + \cdots + \gamma_R, w_R = 0. \tag{24}
\]

Since \( W \) is a basis, it follows that

\[
\varphi_1 (\beta_{x_1^V, y_1^V} + \gamma_1, 1) + \cdots + \varphi_p (\beta_{x_p^V, y_p^V} + \gamma_1, 1) = 0
\]

\[
\varphi_1 (\beta_{x_1^V, y_1^V} + \gamma_p, 1) + \cdots + \varphi_p (\beta_{x_p^V, y_p^V} + \gamma_p, 1) = 0 \tag{25}
\]

This can be written in matrix form as

\[
\begin{pmatrix}
\beta_{x_1^V, y_1^V} + \gamma_1, 1 & \cdots & \beta_{x_p^V, y_p^V} + \gamma_1, 1 \\
\vdots & \ddots & \vdots \\
\beta_{x_1^V, y_1^V} + \gamma_p, 1 & \cdots & \beta_{x_p^V, y_p^V} + \gamma_p, 1
\end{pmatrix}
\begin{pmatrix}
\varphi_1 \\
\vdots \\
\varphi_p
\end{pmatrix}
= 0. \tag{26}
\]

We note that \( \varphi_1 = \cdots = \varphi_p = 0 \) is the only solution of (23) if and only if the matrix in (26) is non-singular. For a \( p \times p \) matrix over a field \( \mathbb{F}_q \), the total number of matrices is \( q^{p^2} \).

Using a combinatorial argument, the number of non-singular matrices is

\[
(q^p - 1)(q^p - q)(q^p - q^2) \cdots (q^p - q^{p-1}) = q^{p^2} \left(1 - \frac{1}{q^p}\right) \geq q^{p^2} \left(1 - \frac{1}{q}\right)^p. \tag{27}
\]

Equation (27) can be explained as follows. For the first column of the matrix, we can choose any vector, except the zero vector. There are \( q^{p-1} \) such vectors. For the second column, we can choose any vector, except any multiple of the first column (which includes the zero vector). Thus, there are \( q^p - q \) choices. In general, there are \( q^p - q^{i-1} \) choices for the \( i \)th column.

So far, we have shown the conditions for \( \varphi_1 = \cdots = \varphi_p = 0 \) to be the only solution to (26). If these conditions are maintained, then (23) becomes

\[
\varphi_{p+1} w_{p+1} + \cdots + \varphi_R w_R = 0. \tag{28}
\]

The only solution to this relation is \( \varphi_{p+1} = \cdots = \varphi_R = 0 \) since the vectors \( w_{p+1}, \ldots, w_R \) are in the basis \( W \) and are therefore linearly independent.

We conclude that if the matrix in (26) is non-singular, then the vectors in \( W' \) are linearly independent. If \( q \geq 2 \), then the number of non-singular matrices is positive, and we can choose the set of coding coefficients \( \beta_{x_i^V, y_j^V} \) for \( 1 \leq i \leq n, 1 \leq j \leq n \), such that the matrix is non-singular.

**Lemma 4.4:** If alphabet size \( q > n(\binom{n}{p}) \left( q^{p^2} \right) \), then there exists a set of coding coefficients \( \beta_{x_i^V, y_j^V} \in \mathbb{F}_q \) for \( 1 \leq i \leq n, 1 \leq j \leq n \), such that all the subsets in \( L_{i+1} \) have linearly independent coding vectors simultaneously.

**Proof:** The subset \( W \in L_i \) contains \( p \geq 1 \) input ports from \( I(V) \). From (27), it follows that for a specific subset \( W' \) in \( L_{i+1} \), the number of non-singular matrices is at least

\[
q^{p^2} \left(1 - \frac{1}{q}\right)^p \geq q^{p^2} \left(1 - \frac{p}{q}\right), \tag{29}
\]

where the last inequality follows from Bernoulli inequality which holds when \( p > 0 \) and \( q \geq 1 \). Thus, the number of singular matrices is at most

\[
q^{p^2} - q^{p^2} \left(1 - \frac{p}{q}\right) = pq^{p^2 - 1} - nq^{p^2 - 1}. \tag{30}
\]

In \( L_{i+1} \), there are at most \( \binom{n}{p} \) subsets. For each subset, there are at most \( nq^{p^2 - 1} \) choices of a set of coding coefficients \( \beta_{x_i^V, y_j^V} \) for \( 1 \leq i \leq n, 1 \leq j \leq n \), such that the coding vectors associated with the subset are linearly dependent. Therefore by the union bound, there are at most \( nq^{p^2 - 1} \) choices
of sets of coding coefficients such that at least one of the subsets in \( L_{l+1} \) can have dependent coding vectors. The total number of choices of coding coefficients is \( q^{n_R} \). Therefore, if \( q > n^{(n_{layer})} \), then we will have at least one set of coding coefficients such that all the subsets in \( L_{l+1} \) have linearly independent coding vectors simultaneously.

We note that for each supernode, the coding vectors of the output ports can be viewed as columns of a parity check matrix of a Maximum Distance Separable (MDS) code with parameters \((n, k = p)\).

**Theorem 4.1:** The invariant of the algorithm is maintained by the stage of coding for the output ports.

**Proof:** By assumption, the invariant is maintained for the set \( Q(O_l) \), which contains the input ports of the supernodes in layer \( l \). We need to show that the invariant is maintained for the set \( Q(I_{l+1}) \), which contains the output ports of the supernodes in layer \( l \), where \( t(I) < t(O) \). This follows by induction from Lemma 4.4.

The average complexity of this stage is computed using arguments similar to those in [29] for the network code construction. According to Lemma 4.4, we can choose the alphabet size at this stage to be \( q = 2n^{(n_{layer})} \). It follows from the proof of Lemma 4.4 that the probability of failure when the coding coefficients are chosen randomly is upper bounded by

\[
P_f \leq \frac{(n_{layer})nq^{n^2-1}}{q^n} = \frac{(n_{layer})n}{q} = \frac{1}{2}.
\] (31)

Therefore, the expected number of trials until the vectors in \( W \) form a basis is at most 2. A single layer has at most \( N_{layer} \) supernodes. The total number of edges connecting input and output ports of a certain supernode is \( n^2 \). It follows that the total number of edges at the layer is bounded by \( n_{layer}n^2 \). In our case, the equivalent to the number of sinks [7] in [29] is the size of \( L_l \), which is at most \( (n_{layer})^2 \). Therefore, similarly to the complexity in [29] for network coding, the average complexity for a layer is \( O((n_{layer})N_{layer}n^2R) \). If the total number of layers is \( \lambda \), then the total average complexity of finding the coding coefficients of the output ports is \( O((n_{layer})N_{layer}n^2R\lambda) \).

2) **Coding for Input Ports:** The coding for the input ports is performed jointly over all supernodes in the same layer. Assume that the coding coefficients of the output ports of layer \( l \) have already all been updated according to Section IV-C1. We need to update the coding coefficients of the input ports of layer \( l+1 \). The list \( L_l \) contains ports from layer \( l \) only. From the list \( L_l \), we choose an arbitrary subset

\[
W = \{w_1, \ldots, w_R\} \in L_l.
\] (32)

The set \( W \) is a subset of \( R \) input ports at layer \( l + 1 \). Let \( B(W, W') \) be the bipartite graph with vertices \((W, W')\) and edges from ports in \( W \) to ports in \( W' \). Let \( G(W, W') \) denote the incidence matrix for \( B(W, W') \). If \( G(W, W') \) is full rank, then we shall show that we can find coding coefficients such that the coding vectors of the ports in \( W \) are linearly independent.

If we cannot find a set \( W' \) such that \( G(W, W') \) is full rank, then we remove \( W \) from the list \( L_l \) and do not replace it with a new set \( W' \). Nevertheless, we shall show that whenever \( W \) is regular, we can always find a regular \( W' \) such that \( G(W, W') \) is full rank.

In Figure 9, we see the sets \( W, W' \). It can be verified that \( R = 3 \); however, the rank of the incidence matrix, \( \text{rank}(G(W, W')) = 2 \). The set \( W' \) is not regular according to Definition 4.1, since the upper and the lower ports in \( W' \) always receive the same symbol.

**Lemma 4.5:** If \( W' \) is a regular set containing \( R \) input ports of supernodes at layer \( l+1 \) for some \( 1 \leq l \leq \lambda - 1 \), then there exists a regular set \( W \) containing \( R \) output ports on supernodes at layer \( l \) such that the incidence matrix \( G(W, W') \) is full rank.

**Proof:** The result follows from [14]–[16]. Specifically, in [14], a path is defined as a disjoint set of edges \((e_1, \ldots, e_\mu)\) where \( e_1 \) starts from the source, \( e_\mu \) enters a certain sink, and \( e_i \) enters the same supernode from which \( e_{i+1} \) emerges. In this proof, we consider the virtual sink \( T(W') \) as our sink.

In [14], linearly independent (LI) paths are defined. Consider the subgraph \( G' \) of the network \( G \) consisting of \( K \) paths from source \( S \) to \( T(W') \). The paths are LI if in \( G' \) the rank of the incidence matrix of any cut is exactly \( K \). We call a set of \( R \) LI paths the underlying flow \( F_u \). It has been shown in [14] that such an underlying flow \( F_u \) exists. In \( F_u \), we can consider the \( R \) output ports of layer \( l \), one from each path. This set of \( R \) ports is guaranteed to be regular, by the definition of the underlying flow. This set of output ports will be chosen as \( W \). The incidence matrix \( G(W, W') \) has full rank, again by definition of linearly independent paths. Therefore, the two properties of the lemma are maintained.

We note that in our construction we do not need to find the edges in \( F_u \). The concept of the underlying flow was introduced only for the proof of Lemma 4.5.

**Lemma 4.6:** For all regular sets \( W' \), we can find coding coefficients such that the coding vectors in \( W' \) are linearly independent simultaneously if the alphabet size \( q > 2(n_{layer})^2 \).

**Proof:** By Lemma 4.5, given a regular set \( W' \), there exists a regular set \( W \) containing \( R \) output ports of supernodes in layer \( l \) such that the incidence matrix \( G(W, W') \) has full rank. The coding vectors of the output ports in \( W \) are given by

\[
W = \{w_1, \ldots, w_R\}.
\] (33)

The coding vectors in the input ports in \( W' \) are given by

\[
W' = \{w'_1, \ldots, w'_R\}.
\] (34)
The vector $w_i$ is in the form

$$w_i = \sum_{j=1}^{R} g_{i,j} k_{j} w_j + \tilde{w}_i$$

(35)

where $k_j$ are the virtual coefficients, and $\tilde{w}_i$ is the contribution of output ports of layer $l$ that are not in $W$. The binary coefficient $g_{i,j}$ is the $(i, j)$th element of matrix $G(W, W')$. We need to find the conditions on the coefficient $k_j$ under which the ports in $W'$ have coding vectors which are linearly independent. Consider the equation

$$\varphi_1 w_i + \cdots + \varphi_R w'_R = 0. \quad (36)$$

Combining with (35), and rearranging,

$$\varphi_1 \left( \sum_{j=1}^{R} g_{1,j} k_{j} w_j \right) + \cdots + \varphi_R \left( \sum_{j=1}^{R} g_{R,j} k_{j} w_j \right) = -a_1 \tilde{w}_1 - \cdots - a_{\text{min}} \tilde{w}_{\text{min}}. \quad (37)$$

We can represent vector $\tilde{w}_i$ in the basis $W$ as

$$\tilde{w}_i = \gamma_{1,i} w_1 + \cdots + \gamma_{R,i} w_R. \quad (38)$$

Combining with (37) and rearranging terms,

$$\varphi_1 \gamma_{1,i} + \cdots + \varphi_R \gamma_{R,i} + \varphi_1 g_{1,1} k_1 + \cdots + \varphi_R g_{R,1} k_1 = 0$$

$$\varphi_1 \gamma_{1,i} + \cdots + \varphi_R \gamma_{R,i} + \varphi_1 g_{1,1} k_1 + \cdots + \varphi_R g_{R,1} k_1 = 0$$

$$\varphi_1 \gamma_{1,i} + \cdots + \varphi_R \gamma_{R,i} + \varphi_1 g_{1,1} k_1 + \cdots + \varphi_R g_{R,1} k_1 = 0.$$

Since $W$ is a basis, it follows that

$$\varphi_1 \gamma_{1,i} + \cdots + \varphi_R \gamma_{R,i} + \varphi_1 g_{1,1} k_1 + \cdots + \varphi_R g_{R,1} k_1 = 0$$

(39)

or in matrix notation,

$$\begin{pmatrix} \gamma_{1,1} + g_{1,1} k_1 & \cdots & \gamma_{1,R} + g_{1,R} k_1 \\ \vdots & \ddots & \vdots \\ \gamma_{R,1} + g_{R,1} k_1 & \cdots & \gamma_{R,R} + g_{R,R} k_1 \end{pmatrix} \begin{pmatrix} \varphi_1 \\ \vdots \\ \varphi_R \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \quad (40)$$

Let $H$ denote the matrix on the left-hand side of (40). The zero vector is the only solution to (40) if and only if the matrix $H$ has full rank. The determinant of the matrix $H$ is a polynomial in the parameters $\{\gamma_{i,j}, k_j, g_{i,j} \}$, for $1 \leq i, j \leq R$. Denote the polynomial by $\Delta_{W,W'}(\beta_{i,j}, k_j, h_{i,j})$. When all $\gamma_{i,j} = 0$, the matrix $H$ is the same as the matrix $G(W, W')$, except that row $i$ of $H$ is equal to the $i$th row of $G(W, W')$ multiplied by $k_i$. Therefore, the polynomial $\Delta_{W,W'}$ is of the form:

$$\Delta_{W,W'}(\gamma_{i,j}, k_j, g_{i,j}) = \det(G(W, W')) \prod_{j=1}^{R} k_j + \delta(\gamma_{i,j}, k_j, g_{i,j}) \quad (41)$$

where $\det(G(W, W')) \neq 0$ is the determinant of (non-singular) matrix $G(W, W')$, and $\delta(\cdot)$ is a polynomial such that the sum of the degrees of all the parameters $k_j, 1 \leq j \leq R$, is smaller than $R$. It follows that for constant $\gamma_{i,j}, g_{i,j}, 1 \leq i, j \leq R$, $\Delta_{W,W'}$ is not the zero polynomial.

The polynomial $\Delta_{W,W'}$ corresponds to the pair of regular subsets $(W, W')$. We need to find the corresponding polynomials for all regular $W'$. Let $P_i$ denote the set of all these pairs $(W, W')$. In order for all such sets $W'$ to have independent coding vectors, we need to assign the coding coefficients such that the following polynomial does not vanish to zero.

$$P = \prod_{W' \exists (W, W') \in P_i} \Delta_{W,W'}. \quad (42)$$

By definition, $P$ is not the zero polynomial since it is a product of nonzero polynomials. Hence, there is a set of coefficients $k_j$ such that the polynomial does not vanish to zero.

Next, we discuss how such coefficients $k_j$’s can be found. Reference [11] proposes an algorithm to find a vector $a$ such that a given polynomial $P$ evaluated at $a$ is not equal to zero, i.e. $P(a) \neq 0$. We reproduce this algorithm from [11] in Algorithm 1 for completeness.

Algorithm 1 Algorithm to Find $a$ Such That $P(a) \neq 0 \quad (11)$

Input: A polynomial $P$ in variables $\zeta_1, \zeta_2, \ldots, \zeta_n$; integers $i = 1, t = 1$

Output: $a = (a_1, a_2, \ldots, a_n)$

Step 1: Find the maximal degree $d$ of $P$ in any variable $\zeta_i$, and let $i$ be the smallest number such that $2^i > d$;

Step 2: Find an element $a_i \in \mathbb{F}_{2^i}$ such that $P(\zeta_i) |_{\zeta_i = a_i} \neq 0$ and let $P \leftarrow P(\zeta_i) |_{\zeta_i = a_i}$;

Step 3: If $t = n$, then halt; otherwise, $t \leftarrow t + 1$ and go to Step 2.

In our scenario, the maximal degree of each variable in $\Delta_{W,W'}$ is 1 due to the structure of the matrix. It follows that the maximal degree of each variable in $P$ is at most $\left\lceil \log \left( \frac{m_{\text{Node}}}{} \right) \right\rceil$. Therefore, $d = \left\lceil \log \left( \frac{m_{\text{Node}}}{} \right) \right\rceil$ and we can always choose $i = \left\lfloor \log \left( \frac{m_{\text{Node}}}{} \right) \right\rfloor$. It follows that an alphabet larger than $2\left\lfloor \log \left( \frac{m_{\text{Node}}}{} \right) \right\rfloor \leq 2^{\frac{m_{\text{Node}}}{R}}$ will ensure that we can find coding coefficients such that all subsets $W'$ have independent coding vectors simultaneously.

Theorem 4.2: The invariant of the algorithm is maintained by the stage of coding for input ports.

Proof: We prove the theorem by induction. For the base case, consider the $R$ upper output ports of the source $S$. We assign the standard basis as coding vectors for these $R$ output ports. We then apply Lemmas 4.5 and 4.6 to the first layer.

For the inductive step, assume that the statement holds for $Q_i$, where $Q_i$ contains the output ports of the supernodes in layer $i$. Now, we show that the invariant is maintained for $Q_{i+1}$, which contains the input ports of the supernodes in layer $l + 1$. If $W'$ is a regular subset, then by Lemma 4.6, there is a coding assignment (according to the code construction presented) such that vectors in $W'$ are independent. Therefore, the invariant is maintained also for layer $l + 1$. 


Theorem 4.3: The invariant of the algorithm is maintained by the construction.

Proof: The theorem follows by induction. For the base case, the invariant is trivially maintained since the input ports of the source $S$ have as their coding vectors the standard basis.

For the general step of the algorithm, given that the invariant is maintained by the stage of coding for the input ports at layer $l$, then using Theorem 4.1, the invariant is maintained by the stage of coding at the output ports at layer $l$. Using Theorem 4.2, the invariant is maintained by the stage of coding at the input ports at layer $l + 1$.

We now analyze the complexity. For a given set $W'$, we need to verify whether there is a regular set $W$, such that the incidence matrix $G(W, W')$ of the bipartite graph $B(W, W')$ has full rank. This can be performed by a determinant computation with complexity $O(R^3)$. Therefore, the total worst case complexity of this stage for a single layer is $O(R^3 n_{l+1}^2)$.

For each variable of polynomial $P$, which corresponds to a certain iteration of Algorithm 1, at most $2^{\log n_{l+1}}$ assignments need to be verified. There are at most $n_{l+1}$ supernodes at each layer. Therefore, the maximal number of output ports, which is also the number of variables $k_j$, is $n n_{l+1}$. It follows that the worst case complexity of the coding for input ports for a single layer is $O((\frac{n n_{l+1}}{R}) (n n_{l+1} + R^3))$. Therefore, if we consider all $\lambda$ layers, the worst case complexity for the coding for the input ports is $O((\frac{n n_{l+1}}{R}) \lambda (n n_{l+1} + R^3))$.

Combining the complexity for both input and output ports, it follows that the total average complexity of the algorithm is $O((\frac{n n_{l+1}}{R}) n_{l+1} n^2 \lambda R)$.

V. DISCUSSIONS AND EXTENSIONS

In Section III, we applied the algebraic network coding framework from [11] to the ADT network problem. This allowed us to model the ADT network problem using a system matrix; thus, providing yet another way to recover the results from [4]–[6], which showed that linear operations are sufficient for multicast connections in ADT networks. A direct consequence of the connection between ADT network problems and algebraic network coding is that random linear network coding, a randomized distributed algorithm for network code construction, achieves the capacity with high probability.

Using the results in [11], we can further extend the capacity characterization of ADT networks to a more general set of connections. In particular, we can show that linear operations are sufficient to achieve capacity in ADT networks for some non-multicast connections, such as multiple multilists, disjoint multicast, and two-level multicast [11].

In addition, the algebraic formulation allows us to incorporate cycles into ADT networks and extend capacity characterizations of ADT networks to such networks. For example, to study cycles in ADT networks, Avestimehr et al. use the “unfolding” or “unraveling” technique introduced in [10]. It is important to note that this unfolded network’s structure is time-dependent — i.e., the code construction and the network structure changes with time as more cycles are incorporated into the unfolded acyclic network.

By using the algebraic framework, we can provide a time-invariant description of the ADT networks with cycles; and thus, provide a time-invariant code construction for these networks [11].

Furthermore, ADT network models wireless networks in the high SNR regime; however, even in the high SNR regime, losses may occur due to time-varying nature of wireless medium. Therefore, understanding the effect of random erasures in ADT networks is valuable. Using the algebraic formulation, we can show that there exists a static code for multicast connections even if the network is oblivious of the failures as long as the multicast connection is achievable despite the failure patterns. In addition, we can extend the ADT network model to time-varying networks with erasures.

These generalizations can be obtained by combining the algebraic formulation presented in this paper with the results in [11]. Therefore, we omit the details in this paper.

VI. CONCLUSIONS

ADT networks [4]–[6] have drawn considerable attention for their potential to approximate the capacity of wireless relay networks; We provided an algebraic network coding [11] framework for ADT networks. This connection between ADT network and algebraic network coding allows the recovery of the results in [4]–[6]. Furthermore, we used results on network coding to better understand ADT networks — showing that results from network coding can be directly applied to ADT networks to extend capacity characterization to a more general set of connections and to ADT networks with cycles and erasures.

Taking advantage of this insight, we proposed an efficient linear code construction for multicasting in ADT networks while guaranteeing decodability. The average complexity of the construction is $O((\frac{n n_{l+1}}{R}) n_{l+1} n^2 \lambda R)$. The required field size is at most $q = n (\frac{n_{l+1}}{R})$; thus, the block length required to represent a symbol is at most $k = \log_q (n (\frac{n_{l+1}}{R}))$. Our code construction only requires supernodes within the same layer to coordinate, and is therefore partially distributed. Furthermore, the algorithm does not require finding network flows or knowing the exact location of the sinks. When normalized by the number of sinks, our code construction has a complexity which is comparable to those of previous coding schemes for a single sink. A possible direction for future research is to use our construction to find new coding schemes for practical multilayer networks with receiver noise.

REFERENCES


Elona Erez received the B.Sc. (summa cum laude), M.Sc. (summa cum laude), and Ph.D. degrees, all in electrical engineering, from Tel Aviv University, Tel Aviv, Israel, 1999, 2002, and 2007, respectively.

During 2007-2008, she was a Postdoctoral Scholar at California Institute of Technology (Caltech), Pasadena. During 2008-2011, she was a Postdoctoral Associate in the Department of Electrical Engineering, Yale University, New Haven, CT. Her research interests are in the fields of information theory and data networks, with special interest in network coding.

Dr. Erez received the Weinstein Prize for an outstanding student in signal processing in 2003 and 2005 and for an outstanding publication in signal processing in 2002 and 2003, and the Colton Prize for an outstanding student in 2003-2006.

MinJi Kim (S’09) received B.S. degrees in Electrical Engineering and Computer Science and in Mathematics in 2006, the Master of Engineering degree in Electrical Engineering and Computer Science in 2007, and Ph.D. in Electrical Engineering and Computer Science in 2012, all from the Massachusetts Institute of Technology (MIT), Cambridge.

She is currently with Oracle Corporation, Redwood Shores, CA. Her research interests include distributed systems, security and reliability of networks, network protocols, network algorithms, wireless communication, and network coding.

Yun Xu (S’09) received the B.S. degree in Automation from Tsinghua University in 2004, as well as the M.S. and Ph.D. degrees in Electrical Engineering from Yale University, in 2009 and 2013, respectively. His research interests are in the areas of network coding, quantization, game theory, and microeconomics, particularly for pricing and mechanism design.

Edmund M. Yeh (SM’12) received his B.S. in Electrical Engineering with Distinction from Stanford University in 1994, his M.Phil in Engineering from the University of Cambridge in 1995, and his Ph.D. in Electrical Engineering and Computer Science under Professor Robert Gallager from MIT in 2001. Since July 2011, he has been Associate Professor of Electrical and Computer Engineering at Northeastern University. Prior to coming to Northeastern, he was Assistant and Associate Professor of Electrical Engineering, Computer Science, and Statistics at Yale University. Professor Yeh has held visiting positions at MIT, Princeton, University of California at Berkeley, Swiss Federal Institute of Technology Lausanne (EPFL), and Technical University of Munich. He has been on the technical staff at the Mathematical Sciences Research Center, Bell Laboratories, Lucent Technologies, Signal Processing Research Department, AT&T Bell Laboratories, and Space and Communications Group, Hughes Electronics Corporation.

Professor Yeh is the recipient of the Alexander von Humboldt Research Fellowship, the Army Research Office Young Investigator Award, the Winston Churchill Scholarship, the National Science Foundation and Office of Naval Research Graduate Fellowships, the Barry M. Goldwater Scholarship, the Frederick Emmons Terman Engineering Scholarship Award, and the President’s Award for Academic Excellence (Stanford University). He is a member of Phi Beta Kappa and Tau Beta Pi. He received the Best Paper Award at the IEEE International Conference on Ubiquitous and Future Networks (ICUFN), Phuket, Thailand, July 2012.

Professor Yeh serves as the Secretary of the Board of Governors of the IEEE Information Theory Society. He is an inaugural Associate Editor for IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING. He was an Associate Editor of IEEE TRANSACTIONS ON MOBILE COMPUTING. He served as Guest Editor-in-Chief of the Special Issue on Wireless Networks for Internet Mathematics, and a Guest Editor of the Special Series on Smart Grid Communications for IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS. He is a Steering Committee Member for IEEE International Conference on Smart Grid Communications (SmartGridComm). He has also served on the technical program committees of many leading conferences and workshops, including IEEE Infocom, Globecom, ICC, SmartGridComm, and ACM MobiHoc.
Muriel Médard (F’08) is a Professor in the Department of EECS at MIT. She was previously an Assistant Professor in the ECE Department at UIUC and Staff Member at MIT Lincoln Laboratory. She received B.S. degrees in EECS, in mathematics, and in humanities, as well as M.S. and Sc.D. degrees in EE, all from MIT. She serves as Editor-in-Chief of IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS (JSAC). She has served as an Associate Editor for the OPTICAL COMMUNICATIONS AND NETWORKING SERIES of IEEE JSAC, IEEE/OSA JOURNAL OF LIGHTWAVE TECHNOLOGY (JLT), IEEE TRANSACTIONS ON INFORMATION THEORY, and the OSA Journal of Optical Networking. She has served as a Guest Editor for IEEE/OSA JLT (two terms), IEEE TRANSACTIONS ON INFORMATION THEORY (TWICE), THE IEEE JSAC (twice), and IEEE TRANSACTIONS ON INFORMATION Forensics and Security. She is a member of the Board of Governors of the IEEE Information Theory Society and serves as the senior Past President. She has served as TPC co-chair of ISIT, WiOpt, and CONEXT. She was awarded the 2009 IEEE Communication Society and Information Theory Society Joint Paper Award, the 2009 IEEE William R. Bennett Prize in the Field of Communications, and the 2002 IEEE Leon K. Kirchmayer Prize Paper Award, as well as several conference Best Paper awards. She was co-recipient of the 2004 MIT Harold E. Edgerton Faculty Achievement Award. In 2007, she was named a Gilbreth Lecturer by the National Academy of Engineering. Her research interests are in the areas of network coding and reliable communications, particularly for optical and wireless networks.
AUTHOR QUERIES

AQ:1 = Please provide the subject area for associate editor.
AQ:2 = Fig. 6 is not cited in body text. Please indicate where it should be cited.