

MIT Open Access Articles

Deterministic Network Model Revisited: An Algebraic Network Coding Approach

The MIT Faculty has made this article openly available. **Please share** how this access benefits you. Your story matters.

Citation: Erez, Elona et al. "Deterministic Network Model Revisited: An Algebraic Network Coding Approach" IEEE Transactions on Information Theory 60, 8 (June 2014): 4867-4879 © Copyright 2019 IEEE

As Published: <http://dx.doi.org/10.1109/tit.2014.2329840>

Publisher: Institute of Electrical and Electronics Engineers (IEEE)

Persistent URL: <https://hdl.handle.net/1721.1/122978>

Version: Original manuscript: author's manuscript prior to formal peer review

Terms of use: Creative Commons Attribution-Noncommercial-Share Alike



Deterministic Network Model Revisited: An Algebraic Network Coding Approach

MinJi Kim, Elona Erez, Edmund M. Yeh, Muriel Médard

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 of wireless medium and interference.

We show that the ADT network model can be described within the algebraic network coding framework introduced by Koetter and Médard. We prove that the ADT network problem can be captured by a single matrix, and show that the min-cut of an ADT network is the rank of this matrix; thus, eliminating the need to optimize over exponential number of cuts between two nodes to compute the min-cut of an ADT network. We extend the capacity characterization for ADT networks to a more general set of connections, including single unicast/multicast connection and non-multicast connections such as multiple multicast, disjoint multicast, and two-level multicast. We also provide sufficiency conditions for achievability in ADT networks for any general connection set. In addition, we show that random linear network coding, a randomized distributed algorithm for network code construction, achieves the capacity for the connections listed above. Furthermore, we extend the ADT networks to those with random erasures and cycles (thus, allowing bi-directional links).

In addition, we propose an efficient linear code construction for the deterministic wireless multicast relay network model. 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, 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 the 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, [1][2] 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.

References [1][2] 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. Note that min-cut of an ADT network may not equal to the graph theoretical cut value, as we shall discuss in Section V. 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 in the network coding for wireline networks [3][4]. 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 [1][5][6].

In this paper, we make a connection between the ADT network and network coding – in particular, algebraic network coding introduced by Koetter and Médard [4]. This paper is based on work from [7][8][9]. Other approaches to operations in high SNR networks have been proposed [10], however, we do not compare these different approaches but build upon the given model proposed by [1][2]. We show that the ADT network problems, including that of computing the min-cut and constructing a code, can be captured by the algebraic network coding framework.

In the context of network coding, [4] showed that the solvability of the communication problem [3] is equivalent to ensuring that a certain polynomial does *not* evaluate to zero – *i.e.* avoid the roots of this certain polynomial. Furthermore, [4] showed that there are only a *fixed finite number of roots* of the polynomial; thus, with large enough field size, decodability can be guaranteed even under randomized coding schemes as shown in [11]. As we increase the field size \mathbb{F}_q , the space of feasible network codes increases exponentially; while the number of roots remain fixed.

We show that the solvability of ADT network problem can be characterized in a similar manner. The important difference between the algebraic network coding in [4] and the ADT network is that the broadcast as well as the interference constraints are embedded in the ADT network. Note that the interference constraint, represented by the additive multiple access channels (MAC), can be easily incorporated into the algebraic framework in [4] by pre-encoding at the transmitting

Manuscript received on ..., and revised on The material in this paper was presented at the Information Theory and Applications Workshop (ITA), San Diego, CA, January 2010; and at the 2010 Allerton Conference on Communication, Control, and Computing, Urbana-Champaign, IL, September 2010.

M. Kim and M. Médard are with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: minjikim@mit.edu; medard@mit.edu).

E. Erez and E. M. Yeh are with the Department of Electrical Engineering, Yale University, New Haven, CT 06511 USA (e-mail: elona.erez@yale.edu; edmund.yeh@yale.edu).

nodes (*i.e.* MAC users). This is due to the fact that the MAC is modeled using finite field additive channel; thus, the operations performed by the MAC can be “canceled” by the transmitter appropriately pre-encoding the packets.

On the other hand, the broadcast constraint may seem more difficult to incorporate, as the same code affects the outputs of the broadcast channel simultaneously, and the dependencies propagate through the network. Thus, in essence, this paper shows that this broadcast constraint is not problematic.

To briefly describe the intuition, consider an ADT network without the broadcast constraint – *i.e.* the broadcast edges do not need to carry the same information. Using this “unconstrained” version of the ADT network, the algebraic framework in [4] can be applied directly; thus, there is only a finite fixed number of roots that need to be avoided. Furthermore, as the field size increases, the probability of randomly selecting a root approaches zero. Now, we “re-apply” the broadcast constraints to this unconstrained ADT network. The broadcast constraint fixes the codes of the broadcast edges to be the same; this is equivalent to intersecting the space of network coding solutions with an hyperplane, which enforces the output ports of the broadcast to carry the same code. As shown in Figure 1, this operations does not change the polynomial whose root we have to avoid, but changes the hyperspace we operate in. As a result, this operation does not affect the roots of the polynomial; thus, there are still only a fixed finite number of roots that need to be avoided, and with high enough field sizes, the probability of randomly selecting a root approaches zero. Note that intersecting the space of network coding solutions with an hyperplane may even “remove” some roots of the polynomial from consideration; therefore, we may effectively have fewer roots to avoid. By the same argument as [4][11], we can then show that the solvability of an ADT network problem is equivalent to ensuring that a certain polynomial does not evaluate to zero within the space defined by the polynomial and the broadcast constraint hyperplane. As a result, we can describe the ADT network within the algebraic network coding framework and extend the random linear network coding results to the ADT networks.

Using this insight, we prove that the ADT network problem can be captured by a single matrix, called the *system matrix*. We show that the min-cut of an ADT network is the rank of the system matrix; thus, eliminating the need to optimize over exponential number of cuts between two nodes to compute the min-cut of an ADT network. We extend the capacity characterization for ADT networks to a more general set of connections, including single unicast/multicast connection and non-multicast connections such as multiple multicast, disjoint multicast, and two-level multicast. We also provide sufficiency conditions for achievability in ADT networks for any general connection set. Furthermore, we extend the results on ADT networks to those with random erasures and cycles (thus, allowing bi-directional links).

We show that a direct consequence of this 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 for the connections listed above. However, random linear

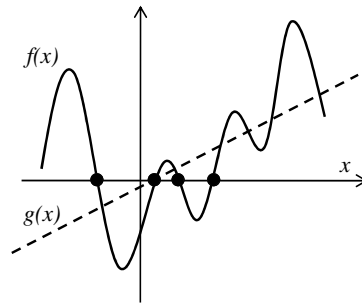


Fig. 1: A polynomial $f(x)$ and a hyperplane $g(x)$ with one variable $x \in \mathbb{R}$. The black dots represent the roots of $f(x)$. When considering the space where $f(x)$ intersects $g(x)$, the hyperplane $g(x)$ limits the space in which $f(x)$ operates in; however, does not change the roots of $f(x)$. Some of the roots of $f(x)$ may no longer be “feasible” given the additional constraint; thus, this operation may reduce the number of roots that we have to consider.

network coding does not guarantee decodability; it allows decodability at all destinations with high probability.

Therefore, we propose an efficient linear code construction for multicasting in ADT networks that *guarantees decodability*, if such 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 [12][13][14], 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. 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.

Our construction can be viewed as a non-straightforward generalization of the algorithm in [15] 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 into the ADT network model.

The paper is organized as follows. We present the network model in Section III, and an algebraic formulation of the ADT network in Section IV. Using this algebraic formulation, we provide a definition of the min-cut in ADT networks in Section V. In Sections VI, we restate the Min-cut Max-flow theorem using our algebraic formulation, and present new capacity characterizations for ADT networks to a more general set of traffic requirements in Section VII. The results in Section VII show the optimality of linear operations for non-multicast connections such as disjoint multicast and two-level multicast

connections. In Section VIII, we study ADT networks with link failures, and characterize the set of link failures such that the network solution is guaranteed to remain successful. Furthermore, in Sections IX, we extend the achievability results to ADT networks with delay. In Section X, we present our code construction algorithm for multicasting in ADT networks, and analyze its performance. Finally, we conclude in Section XI.

II. BACKGROUND

Avestimeher *et al.* introduced the ADT network model to better approximate wireless networks [1][2]. In the same work, they characterized the capacity of the ADT networks, and 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, [1] considered the single relay channel and the diamond network, and showed that the gap between the capacity of the ADT model and that of Gaussian network is within 1 bit and 2 bits, respectively. Reference [5] considered many-to-one and one-to-many Gaussian interference networks. The networks in [5] are special cases of 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 [5] provided an alternative proof to [16] on the existence of a scheme that can achieve a constant gap from the capacity for all values of channel parameters. In [6], the half-duplex butterfly network was considered. They showed that the deterministic model approximates the symmetric Gaussian butterfly network to within a constant.

As a result, 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 previous code constructions have been proposed for wireless relay networks. It is important to observe that in the code constructions for unicast communication, routing [13] or one-bit operations [12] are sufficient for achieving the capacity of the deterministic model. Amaidruz and Fragouli [17] 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 $O(|\mathcal{V}||\mathcal{E}|R^5)$, where \mathcal{V} is the set of nodes in the network, \mathcal{E} is the set of edges, and R is the rate of the code. In [13], 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 [13] extracts at each relay node a subset of the input vectors and sets the outputs to the same values as that subset. In [14], 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, [14] achieved a code complexity $O(\lambda N_{layer}^3 \log N_{layer})$, where λ

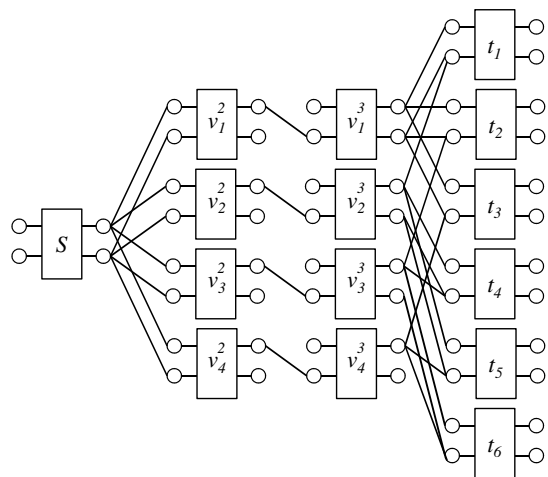


Fig. 2: Example network for a non-binary code.

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 [18]; thus, the fact that ADT networks are matroidal [14] is consistent with our result.

In the case for 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 [19][20][21] 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 $v_i^2, 1 \leq i \leq 4$ at the second layer, the transmitted symbol is a certain function of the symbols x_1, x_2 , given by $y_i = f_i(x_1, x_2)$. Node $v_i^3, 1 \leq i \leq 4$ 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 $f_j^1(y_j), f_j^2(y_j) + f_k^1(y_k)$ 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 $f_j^1(y_j) = y_j$ 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 $f_j^2(y_j)$ 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 [20] (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 higher field size, $\mathbb{F}_q, q \geq 2$, or use vector coding (or both).

References [22][23], independently from [7][8][9], proposed a polynomial time algorithm for multicasting in ADT networks. In particular, [23] extended the algebraic network coding result [4] to vector network coding, and showed that constructing a valid vector code is equivalent to certain algebraic conditions. This result [23] is supported by the result

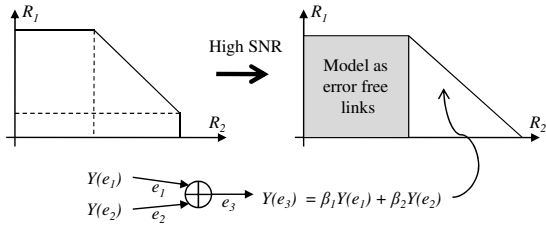


Fig. 3: Additive MAC with two users, and the corresponding rate region. The triangular region is modeled as a set of finite field additive MACs.

from [24]. Reference [24] introduced network codes, called *permute-and-add*, that only require bit-wise vector operations to take advantage of low-complexity operations in \mathbb{F}_2 . In addition, [24] 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 [4], suggests that an algebraic property of a scalar code may translate into another algebraic property of the corresponding vector code.

III. NETWORK MODEL

As in [1][2], we shall consider the high SNR regime, in which interference is the dominating factor. In high SNR, analog network coding, which allows/encourages strategic interference, is near optimal [10]. Analog network coding is a physical layer coding technique, introduced by [25], 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. Note that a network operating in high SNR regime is different from a network with high gain since a large gain amplifies the noise as well as the signal.

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 3. 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), which is equivalent to using noiseless finite-field additive MAC [26]. This result holds not only for binary field additive MAC, but also for higher field size additive MAC [26].

The ADT network model uses binary channels, and thus, binary additive MACs are used to model interference. Prior to [1][2], Effros *et al.* presented an additive MAC over a finite field \mathbb{F}_q [27]. The Min-cut Max-flow theorem holds for all of the cases above. It may seem that the ADT network model differs greatly from that of [27] owing to the difference in 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 higher field size \mathbb{F}_q and vice versa by normalizing to an appropriate time unit. Note that ADT network model uses binary additive MACs and point-to-point links. Therefore, our work in part shows an equivalence of higher field size codes and binary-vector codes in ADT networks.

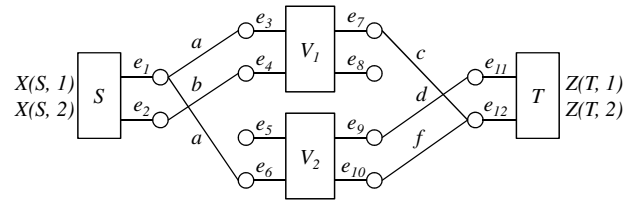


Fig. 4: Example network. We omit $I(S)$ and $O(T)$ in this diagram as they do not participate in the communication.

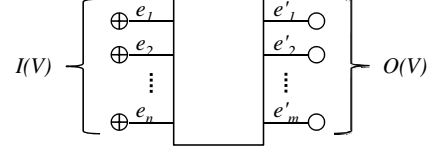


Fig. 5: A supernode V .

As noted in Section II, [24] presented a method of converting between binary-vector codes and higher field size codes. We can achieve a higher field size in ADT networks by combining 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}_4 . Thus, selecting a larger field size \mathbb{F}_q , $q > 2$ in ADT network results in fewer but higher capacity parallel channels. Reference [24] also provides a conversion from a code in a higher 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 [28]; thus, we need to operate in a higher field size. Therefore, we shall not restrict ourselves to \mathbb{F}_2 .

We now proceed to defining 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

$$\begin{aligned}
 Y(e_1) &= \alpha_{(1,e_1)}X(S,1) + \alpha_{(2,e_1)}X(S,2) \\
 Y(e_2) &= \alpha_{(1,e_2)}X(S,1) + \alpha_{(2,e_2)}X(S,2) \\
 Y(e_3) &= Y(e_6) = Y(e_1) \\
 Y(e_4) &= Y(e_2) \\
 Y(e_5) &= Y(e_8) = 0 \\
 Y(e_7) &= \beta_{(e_3,e_7)}Y(e_3) + \beta_{(e_4,e_7)}Y(e_4) \\
 Y(e_9) &= Y(e_{11}) = \beta_{(e_6,e_9)}Y(e_6) \\
 Y(e_{10}) &= \beta_{(e_6,e_{10})}Y(e_6) \\
 Y(e_{12}) &= Y(e_7) + Y(e_{10}) \\
 Z(T,1) &= \epsilon_{(e_{11},(T,1))}Y(e_{11}) + \epsilon_{(e_{12},(T,1))}Y(e_{12}) \\
 Z(T,2) &= \epsilon_{(e_{11},(T,2))}Y(e_{11}) + \epsilon_{(e_{12},(T,2))}Y(e_{12})
 \end{aligned}$$

Fig. 6: Equations relating the various processes of Figure 4.

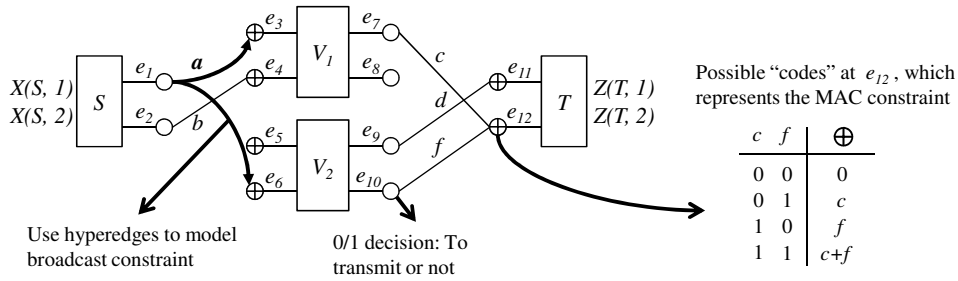


Fig. 7: A new interpretation of the example network from Figure 4. The broadcast channel is modeled using an *hyperedge*. As a result, an output port's decision to transmit or not naturally affects all the input ports adjacent to it. Furthermore, interference is modeled using a finite field additive MAC, which provides a set of possible *binary codes* at the input ports.

in Figure 4. 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 5. Let $\mathcal{S}, \mathcal{T} \subseteq \mathcal{V}$ be the set of source and destination supernodes. An edge (e_1, e_2) may exist from an output port $e_1 \in O(V_1)$ to an input port $e_2 \in I(V_2)$, for any $V_1, V_2 \in \mathcal{V}$. Let $\mathcal{E}(V_1, V_2)$ be the set of edges from $O(V_1)$ to $I(V_2)$. All edges are of unit capacity, where capacity is normalized with respect to the symbol size of \mathbb{F}_q .

Noise is embedded, or *hard-coded*, in the structure of the ADT network in the following way. Parallel links of $\mathcal{E}(V_1, V_2)$ deterministically model noise between V_1 and V_2 . Let $SNR_{(V_i, V_j)}$ be the signal-to-noise ratio from supernode V_i to supernode V_j . Then, $|\mathcal{E}(V_1, V_2)| = \lceil \frac{1}{2} \log SNR_{(V_i, V_j)} \rceil$. Thus, the number of edges between two supernodes V_i and V_j represents the channel quality (equivalently, the noise) between the two supernodes.

Given such a wireless network $G = (\mathcal{V}, \mathcal{E})$, let \mathcal{S} be the set of sources. A source supernode $S \in \mathcal{S}$ has *independent* random processes $\mathcal{X}(S) = [X(S, 1), X(S, 2), \dots, X(S, \mu(S))]$, $\mu(S) \leq |O(S)|$, 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 $\mathcal{X}(S, T) \subseteq \mathcal{X}(S)$, by the means 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 $R(c) = \sum_{X(S, i) \in \mathcal{X}(S, T)} H(X(S, i)) = |\mathcal{X}(S, T)|$ (symbols).

Information is transmitted through the network in the following manner. A supernode V sends information through $e \in O(V)$ at a rate at most one symbol per time unit. Let $Y(e)$ denote the random process at port e . In general, $Y(e)$, $e \in O(V)$, is a function of $Y(e')$, $e' \in I(V)$. In this paper, we consider only linear functions.

$$Y(e) = \sum_{e' \in I(V)} \beta_{(e', e)} Y(e'), \text{ for } e \in O(V). \quad (1)$$

For a source supernode S , and $e \in O(S)$,

$$Y(e) = \sum_{e' \in I(V)} \beta_{(e', e)} Y(e') + \sum_{X(S, i) \in \mathcal{X}(S)} \alpha_{(i, e)} X(S, i). \quad (2)$$

Finally, the destination T receives a collection of input processes $Y(e')$, $e' \in I(T)$. Supernode T generates a set of

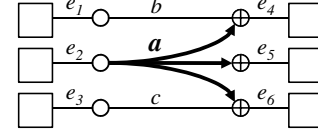


Fig. 8: An example of finite field additive MAC.

random processes $\mathcal{Z}(T) = [Z(T, 1), Z(T, 2), \dots, Z(T, \nu(T))]$ where

$$Z(T, i) = \sum_{e' \in I(T)} \epsilon_{(e', (T, i))} Y(e'). \quad (3)$$

A connection $c = (S, T, \mathcal{X}(S, T))$ is established successfully if $\mathcal{X}(S) = \mathcal{Z}(T)$. A supernode V is said to *broadcast* to a set $\mathcal{V}' \subseteq \mathcal{V}$ if $\mathcal{E}(V, V') \neq \emptyset$ for all $V' \in \mathcal{V}'$. In Figure 4, S broadcasts to supernodes V_1 and V_2 . Superposition occurs at the input port $e' \in I(V)$, i.e. $Y(e') = \sum_{(e, e') \in \mathcal{E}} Y(e)$ over a finite field \mathbb{F}_q . We say there is a $|\mathcal{V}'|$ -user MAC channel if $\mathcal{E}(V', V) \neq \emptyset$ for all $V' \in \mathcal{V}'$. In Figure 4, supernodes V_1 and V_2 are users, and T the receiver in a 2-user MAC.

For a given network G and a set of connections \mathcal{C} , we say that (G, \mathcal{C}) is *solvable* if it is possible to establish successfully all connections $c \in \mathcal{C}$. The broadcast and MAC constraints are given by the network; however, we are free to choose the variables $\alpha_{(i, e)}$, $\beta_{(e', e)}$, and $\epsilon_{(e', i)}$ from \mathbb{F}_q . Thus, the problem of checking whether a given (G, \mathcal{C}) is solvable is equivalent to finding a feasible assignment to $\alpha_{(i, e)}$, $\beta_{(e', e)}$, and $\epsilon_{(e', (T, i))}$.

Example 3.1: The equations in Figure 6 relate the various processes in the example network in Figure 4. Note that in Figure 4, we have set $Y(e_1) = a$, $Y(e_2) = b$, $Y(e_7) = c$, $Y(e_9) = d$, and $Y(e_{10}) = f$ for notational simplicity.

A. An Interpretation of the Network Model

The ADT network model uses multiple channels from an output port to model broadcast. In Figure 4, there are two edges from output port e_1 to input ports e_3 and e_6 ; however, due to the broadcast constraint, the two edges (e_1, e_3) and (e_1, e_6) carry the same information a . This introduces considerable complexity in constructing a network code as well as computing the min-cut of the network [1][2][12][14]. 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 by introducing the use of hyperedges, as shown in Figure 7 and Section IV. 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. This removes the difficulties of computing the min-cut of ADT networks (Section V), as it naturally captures the broadcast dependencies.

The finite field additive MAC model can be viewed as a set of codes that an input port may receive. As shown in Figure 7, input port e_{12} receives one of the four possible codes. The code that e_{12} receives depends on output ports e_7 's and e_{10} 's decision to transmit or not.

The difficulty in constructing a network code does not come from any single broadcast or MAC constraint. The difficulty in constructing a code is in satisfying multiple MAC and broadcast constraints simultaneously. For example, in Figure 8, the fact that e_4 may receive $a + b$ does not constrain the choice of a nor b . This is because we can choose any a and b such that $a + b \neq 0$, and ensure that both a and b are decoded as long as enough degrees of freedom are received by the destination node. The same argument applies to e_6 receiving $a + c$. However, the problem arises from the fact that a choice of value for a at e_4 interacts both with b and c . In such a case, we need to ensure that both $a + b \neq 0$ and $a + c \neq 0$; thus, our constraint is $(a + b)(a + c) \neq 0$. As the network grows in size, we will need to satisfy more constraints simultaneously. As we shall see in Section V, we eliminate this difficulty by allowing the use of a larger field, \mathbb{F}_q .

IV. ALGEBRAIC NETWORK CODING FORMULATION

We provide an algebraic formulation for the ADT network problem (G, \mathcal{C}) , and present an algebraic condition under which the system (G, \mathcal{C}) is solvable. We assume that G is acyclic in this section; however, we shall extend the results in this section to ADT networks with cycles in Section IX. For simplicity, we describe the multicast problem with a single source S and a set of destination supernodes \mathcal{T} , as in Figure 9. However, this formulation can be extended to multiple source S_1, S_2, \dots, S_K by adding a super-source S as in Figure 10.

We define a system matrix M to describe the relationship between the source's random processes $\mathcal{X}(S)$ and the destinations' processes $\mathcal{Z} = [\mathcal{Z}(T_1), \mathcal{Z}(T_2), \dots, \mathcal{Z}(T_{|\mathcal{T}|})]$. Thus, we want to characterize M where

$$\mathcal{Z} = \mathcal{X}(S) \cdot M. \quad (4)$$

The matrix M is composed of three matrices, A , F , and B .

A. Adjacency matrix F

Given G , we define the adjacency matrix F as follows:

$$F_{i,j} = \begin{cases} 1 & \text{if } (e_i, e_j) \in \mathcal{E}, \\ \beta_{(e_i, e_j)} & \text{if } e_i \in I(V), e_j \in O(V) \text{ for } V \in \mathcal{V}, \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

Matrix F is defined on the ports, rather than on the supernodes. This is because, in the ADT model, each port is the basic receiver/transmitter unit. Each entry $F_{i,j}$ represents the input-output relationships of the ports. A zero entry indicates that

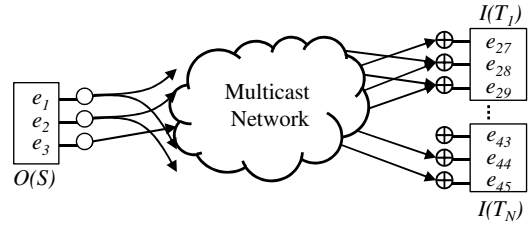


Fig. 9: Single multicast network with source S and receivers T_1, \dots, T_N .

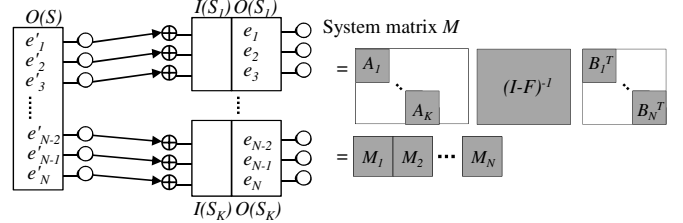


Fig. 10: A network with multiple sources S_1, S_2, \dots, S_K can be converted to a single source problem by adding a super-source S with $|O(S)| = \sum_{i=1}^K |O(S_i)|$. Each $e'_j \in O(S)$ has a “one-to-one connection” to a $e_j \in O(S_i)$, for $i \in [1, K]$. Matrix A_i represents the encoding matrix for source S_i , while B_j is the decoding matrix at destination T_j . The white area represents the zero elements, and the shaded area represents the coding coefficients.

the ports are not directly connected, while an entry of one represents that they are connected. The adjacency matrix F naturally captures the physical structure of the ADT network. Note that a row with multiple entries of 1 represents the broadcast hyperedge; while a column with multiple entries of 1 represents the MAC constraint. Note that the 0-1 entries of F represent the *fixed* network topology as well as the broadcast and MAC constraints. On the other hand, $\beta_{(e_i, e_j)}$ are free variables, representing the coding coefficients used at V to map the input port processes to the output port processes. This is the key difference between the work presented here and in [4] – F is partially fixed in the ADT network model due to network topology and broadcast/MAC constraints, while in [4], only the network topology affects F .

In [1][2], the supernodes are allowed to perform any internal operations; while in [12][14], only permutation matrices (*i.e.* routing) are allowed. Note that [12][14] only consider a single unicast traffic, in which routing is known to be sufficient. References [1][2] showed that linear operations are sufficient to achieve the capacity in ADT networks for a single multicast traffic. We consider a general setup in which $\beta_{(e_i, e_j)} \in \mathbb{F}_q$ – thus, allowing any matrix operation, as in [1][2].

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 $I + F + F^2 + F^3 + \dots$ represents the connectivity of the network. It can be verified that F is nilpotent, which means that there exists a k such that F^k is a zero matrix. As a result, $I + F + F^2 + F^3 + \dots$ can be written as $(I - F)^{-1}$. Thus, $(I - F)^{-1}$ represents the impulse response of the network. Note that, $(I - F)^{-1}$ exists for all

$$\begin{pmatrix}
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \beta_{(e_3, e_7)} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \beta_{(e_4, e_7)} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \beta_{(e_6, e_9)} & \beta_{(e_6, e_{10})} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

Fig. 11: 12×12 adjacency matrix F for network in Figure 4.

acyclic network since $I - F$ is an upper-triangle matrix with all diagonal entries equal to 1; thus, $\det(I - F) = 1$.

Example 4.1: In Figure 11, we provide the 12×12 adjacency matrix F for the example network in Figures 4 and 7. Note that the first row (with two entries of 1) represents the broadcast hyperedge, e_1 connected to both e_3 and e_6 . The last column with two entries equal to 1 represents the MAC constraint, both e_7 and e_{10} transmitting to e_{12} . The highlighted elements in F represent the coding variables, $\beta_{(e', e)}$, of V_1 and V_2 . For some (e', e) , $\beta_{(e', e)} = 0$ since these ports of V_1 and V_2 are not used.

B. Encoding matrix A

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

$$A_{i,j} = \begin{cases} \alpha_{(i, e_j)} & \text{if } e_j \in O(S) \text{ and } X(S, i) \in \mathcal{X}(S), \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

Example 4.2: We provide the 2×12 encoding matrix A for the network in Figure 4.

$$A = \begin{pmatrix} \alpha_{1, e_1} & \alpha_{1, e_2} & 0 & \cdots & 0 \\ \alpha_{2, e_1} & \alpha_{2, e_2} & 0 & \cdots & 0 \end{pmatrix}.$$

C. Decoding matrix B

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

$$B_{i, (T_j, k)} = \begin{cases} \epsilon_{(e_i, (T_j, k))} & \text{if } e_i \in I(T_j), Z(T_j, k) \in \mathcal{Z}(T_j), \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

Example 4.3: We provide the 2×12 decoding matrix B for the example network in Figure 4.

$$B = \begin{pmatrix} 0 & \cdots & 0 & \epsilon_{(e_{11}, (T, 1))} & \epsilon_{(e_{12}, (T, 1))} \\ 0 & \cdots & 0 & \epsilon_{(e_{11}, (T, 2))} & \epsilon_{(e_{12}, (T, 2))} \end{pmatrix}.$$

D. System matrix M

Theorem 4.1: Given a network $G = (\mathcal{V}, \mathcal{E})$, let A , B , and F be the encoding, decoding, and adjacency matrices, respectively. Then, the system matrix M is given by

$$M = A(1 - F)^{-1} B^T. \quad (8)$$

System matrix $M = A(I - F)^{-1} B^T$

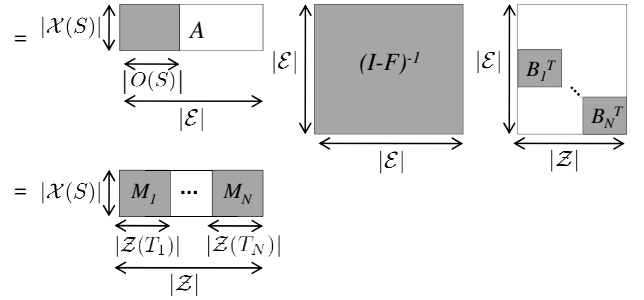


Fig. 12: The system matrix M and its components A , $(I - F)^{-1}$, and B for a single multicast connection with source S and destinations T_i , $i \in [1, N]$.

Proof: The proof of this theorem is similar to that of Theorem 3 in [4]. As previously mentioned, $(I - F)^{-1} = (I + F + F^2 + \dots)$ always exists for an acyclic network G . ■

Note that the algebraic framework shows a clear separation between the given physical constraints (fixed 0-1 entries of F showing the topology and the broadcast/MAC constraints), and the coding decisions. As mentioned previously, we can freely choose the coding variables $\alpha_{(i, e_j)}$, $\epsilon_{(e_i, (T_j, k))}$, and $\beta_{(e_i, e_j)}$. Thus, solvability of (G, \mathcal{C}) is equivalent to assigning values to $\alpha_{(i, e_j)}$, $\epsilon_{(e_i, (T_j, k))}$, and $\beta_{(e_i, e_j)}$ such that each receiver $T \in \mathcal{T}$ is able to decode the data it is intended to receive.

Example 4.4: We can combine the matrices F , A , and B from Examples 4.1, 4.2, and 4.3 respectively to obtain the system matrix $M = A(I - F)^{-1} B^T$ for the network in Figure 4. We show a schematic of the system matrix M in Figure 12.

V. DEFINITION OF MIN-CUT

Consider a source S and a destination T . Reference [1] proves the maximal achievable rate to be the minimum value of all S - T cuts, denoted $\text{mincut}(S, T)$, which we reproduce below in Definition 5.1.

Definition 5.1 (Min-cut [1][2]): A cut Ω between a source S and a destination T is a partition of the supernodes into two disjoint sets Ω and Ω^c such that $S \in \Omega$ and $T \in \Omega^c$. For any cut, G_Ω is the incidence matrix associated with the bipartite graph with ports in Ω and Ω^c . Then, the capacity of the given ADT network (equivalently, $\text{mincut}(S, T)$) is defined as

$$\text{mincut}(S, T) = \min_{\Omega} \text{rank}(G_\Omega).$$

This rate of $\text{mincut}(S, T)$ can be achieved using linear operations for a single unicast/multicast connection. ■

Note that, with the above definition, in order to compute $\text{mincut}(S, T)$, we need to optimize over all cuts between S and T . In addition, the proof of achievability in [1] is not constructive, as it assumes infinite block length and does not consider the details of internal supernode operations.

We introduce a new algebraic definition of the min-cut, and show that it is equivalent to that of Definition 5.1.

Theorem 5.1: The capacity of the given ADT, equivalently the minimum value of all $S - T$ cuts $\text{mincut}(S, T)$, is

$$\begin{aligned} \text{mincut}(S, T) &= \min_{\Omega} \text{rank}(G_{\Omega}) \\ &= \max_{\alpha_{(i,e)}, \beta_{(e',e)}, \epsilon_{(e',i)}} \text{rank}(M). \end{aligned}$$

Proof: By [1], we know that $\text{mincut}(S, T) = \min_{\Omega} \text{rank}(G_{\Omega})$. Therefore, we show that $\max_{\alpha, \beta, \epsilon} \text{rank}(M)$ is equivalent to the maximal achievable rate in an ADT network.

First, we show that $\text{mincut}(S, T) \geq \max_{\alpha, \beta, \epsilon} \text{rank}(M)$. In our algebraic formulation, $\mathcal{Z}(T) = \mathcal{X}(S)M$; thus, the rank of M represents the rate achieved. Let $R = \max_{\alpha, \beta, \epsilon} \text{rank}(M)$. Then, there exists an assignment of $\alpha_{(i,e)}$, $\beta_{(e',e)}$, and $\epsilon_{(e',i)}$ such that the network achieves a rate of R . By the definition of min-cut, it must be the case that $\text{mincut}(S, T) \geq R$.

Second, we show that $\text{mincut}(S, T) \leq \max_{\alpha, \beta, \epsilon} \text{rank}(M)$. Assume that $R = \text{mincut}(S, T)$. Then, by [1][2], there exists a linear configuration of the network such that we can achieve a rate of R such that the destination T is able to reproduce $\mathcal{X}(S, T)$. This configuration of the network provides a linear relationship of the source-destination processes (actually, the resulting system matrix is an identity matrix); thus, an assignment of the variables $\alpha_{(i,e)}$, $\beta_{(e',e)}$, and $\epsilon_{(e',i)}$ for our algebraic framework. We denote M' to be the system matrix corresponding to this assignment. Note that, by the definition, M' is an $R \times R$ matrix with a rank of R . Therefore, $\max_{\alpha, \beta, \epsilon} \text{rank}(M) \geq \text{rank}(M') = \text{mincut}(S, T)$. ■

The system matrix M (thus, the network and decodability at the destinations) depends not only on the structure of the ADT network, but also on the field size used, supernodes' internal operations, transmission rate, and connectivity. For example, the network topology may change with a choice of larger field size, since larger field sizes result in fewer parallel edges/channels. Another example, if we adjust the rate such that $|\mathcal{X}(S)| \leq \text{mincut}(S, T)$, then M is full rank. However, if $|\mathcal{X}(S)| > \text{mincut}(S, T)$, then M may have rank of $\text{mincut}(S, T)$ but not be full-rank. It is important to note that the cut value in the ADT network may not equal to the graph theoretical cut value (see Figure 2 in [12]).

VI. MIN-CUT MAX-FLOW THEOREM

In this section, we provide an algebraic interpretation of the Min-cut Max-flow theorem for a single unicast connection and a single multicast connection [1][2]. This result is a direct consequence of [4] when applied to the algebraic formulation for the ADT network. We also show that a distributed randomized coding scheme achieves the capacity for these connections.

Theorem 6.1 (Min-cut Max-flow Theorem): Given an acyclic network G with a single connection $c = (S, T, \mathcal{X}(S, T))$ of rate $R(c) = |\mathcal{X}(S, T)|$, the following are equivalent:

- 1) A unicast connection c is feasible.
- 2) $\text{mincut}(S, T) \geq R(c)$.
- 3) There exists an assignment of $\alpha_{(i,e_j)}$, $\epsilon_{(e_i,(T_j,k))}$, and $\beta_{(e_i,e_j)}$ such that the $R(c) \times R(c)$ system matrix M is invertible in \mathbb{F}_q (i.e. $\det(M) \neq 0$).

Proof: Statements 1) and 2) have been shown to be equivalent in ADT networks [1][12][14]. We now show that 1) and 3) are equivalent. Assume that there exists an assignment such that $\det(M) \neq 0$ in \mathbb{F}_q . Then, the system matrix M is invertible; thus, there exists M^{-1} such that $\mathcal{X}(S) = \mathcal{Z}M^{-1}$, and a connection of rate $R(c) = |\mathcal{X}(S, T)|$ is established. Conversely, if connection c is feasible, there exists a solution to the ADT network G that achieves a rate of $R(c)$. When using this ADT network solution, the destination T is able to reproduce $\mathcal{X}(S, T)$; thus the resulting system matrix is an identity matrix, $M = I$. Therefore, M is invertible. ■

Corollary 6.1 (Random Coding for Unicast): Consider an ADT network problem with a single connection $c = (S, T, \mathcal{X}(S, T))$ of rate $R(c) = |\mathcal{X}(S, T)| \leq \text{mincut}(S, T)$. Then, random linear network coding, where some or all code variables $\alpha_{(i,e_j)}$, $\epsilon_{(e_i,(T_j,k))}$, and $\beta_{(e_i,e_j)}$ are chosen independently and uniformly over all elements of \mathbb{F}_q , guarantees decodability at destination T with high probability at least $(1 - \frac{1}{q})^{\eta}$, where η is the number of links carrying random combinations of the source processes.

Proof: From Theorem 6.1, there exists an assignment of $\alpha_{(i,e_j)}$, $\epsilon_{(e_i,(T_j,k))}$, and $\beta_{(e_i,e_j)}$ such that $\det(M) \neq 0$, which gives a capacity-achieving network code for the given (G, \mathcal{C}) . Thus, this connection c is feasible for the given network. Reference [11] proves that random linear network coding is capacity-achieving and guarantees decodability with high probability $(1 - \frac{1}{q})^{\eta}$ for such a feasible unicast connection c . ■

Theorem 6.2 (Single Multicast Theorem): Given an acyclic network G and connections $\mathcal{C} = \{(S, T_1, \mathcal{X}(S)), (S, T_2, \mathcal{X}(S)), \dots, (S, T_N, \mathcal{X}(S))\}$, (G, \mathcal{C}) is solvable if and only if $\text{mincut}(S, T_i) \geq |\mathcal{X}(S)|$ for all i .

Proof: If (G, \mathcal{C}) is solvable, then $\text{mincut}(S, T_i) \geq |\mathcal{X}(S)|$. Therefore, we only have to show the converse. Assume $\text{mincut}(S, T_i) \geq |\mathcal{X}(S)|$ for all $i \in [1, N]$. The system matrix $M = \{M_i\}$ is a concatenation of $|\mathcal{X}(S)| \times |\mathcal{X}(S)|$ matrices where $\mathcal{Z}(T_i) = \mathcal{X}(S)M_i$, as shown in Figure 12. We can write $M = [M_1, M_2, \dots, M_N] = A(I - F)^{-1}B^T = A(I - F)^{-1}[B_1, B_2, \dots, B_N]$. Thus, $M_i = A(I - F)^{-1}B_i$. Note that A and B_i 's do not substantially contribute to the system matrix M_i since A and B_i only perform linear encoding and decoding at the source and destinations, respectively.

By Theorem 6.1, there exists an assignment of $\alpha_{(i,e_j)}$, $\epsilon_{(e_i,(T_j,k))}$, and $\beta_{(e_i,e_j)}$ such that each individual system submatrix M_i is invertible, i.e. $\det(M_i) \neq 0$. However, an assignment that makes $\det(M_i) \neq 0$ may lead to $\det(M_j) = 0$ for $i \neq j$. Thus, we need to show that it is possible to achieve *simultaneously* $\det(M_i) \neq 0$ for all i (equivalently $\prod_i \det(M_i) \neq 0$). By [11], we know that if the field size is larger than the number of receivers ($q > N$), then there exists an assignment of $\alpha_{(i,e_j)}$, $\epsilon_{(e_i,(T_j,k))}$, and $\beta_{(e_i,e_j)}$ such that $\det(M_i) \neq 0$ for all i . ■

Corollary 6.2 (Random Coding for Multicast): Consider an ADT network problem with a single multicast connection $\mathcal{C} = \{(S, T_1, \mathcal{X}(S)), (S, T_2, \mathcal{X}(S)), \dots, (S, T_N, \mathcal{X}(S))\}$ with $\text{mincut}(S, T_i) \geq |\mathcal{X}(S)|$ for all i . Then, random linear network coding, where some or all code variables $\alpha_{(i,e_j)}$, $\epsilon_{(e_i,(T_j,k))}$, and $\beta_{(e_i,e_j)}$ are chosen independently and

uniformly over all elements of \mathbb{F}_q , guarantees decodability at destination T_i for all i simultaneously with high probability at least $(1 - \frac{N}{q})^\eta$, where η is the number of links carrying random combinations of the source processes; thus, $\eta \leq |\mathcal{E}|$.

Proof: Given that the multicast connection is feasible (which is true by Theorem 6.2), reference [11] shows that random linear network coding achieves the capacity for multicast connections, and allows all destination supernodes to decode the source processes $\mathcal{X}(S)$ with high probability of at least $(1 - \frac{N}{q})^\eta$. ■

Theorem 6.1 and Theorem 6.2 provide an alternate proof of sufficiency of linear operations for unicast and multicast in ADT networks, which was first shown in [1].

VII. EXTENSIONS TO OTHER CONNECTIONS

In this section, we extend the ADT network results to a more general set of traffic requirements. We use the algebraic formulation and the results from [4] to characterize the feasibility conditions for a given problem (G, \mathcal{C}) .

A. Multiple Multicast

Theorem 7.1 (Multiple Multicast Theorem): Given a network G and a set of connections $\mathcal{C} = \{(S_i, T_j, \mathcal{X}(S_i)) \mid S_i \in \mathcal{S}, T_j \in \mathcal{T}\}$, (G, \mathcal{C}) is solvable if and only if Min-cut Max-flow bound is satisfied for any cut that separates the source supernodes \mathcal{S} and a destination T_j , for all $T_j \in \mathcal{T}$.

Proof: We first introduce a super-source S with $|O(S)| = \sum_{S_i \in \mathcal{S}} |O(S_i)|$, and connect each $e'_j \in O(S)$ to an input of S_i such that $e_j \in O(S_i)$ as shown in Figure 10. Then, we apply Theorem 6.2, which proves the statement. ■

Corollary 7.1 (Random Coding for Multiple Multicast): Consider an ADT network problem with multiple multicast connections $\mathcal{C} = \{(S_i, T_j, \mathcal{X}(S_i)) \mid S_i \in \mathcal{S}, T_j \in \mathcal{T}\}$ with $\text{mincut}(\mathcal{S}, T_j) \geq \sum_i |\mathcal{X}(S_i)|$ for all i . Then, random linear network coding, where some or all code variables $\alpha_{(e_i, e_j)}$, $\epsilon_{(e_i, (T_j, k))}$, and $\beta_{(e_i, e_j)}$ are chosen independently and uniformly over all elements of \mathbb{F}_q , guarantees decodability at destination T_i for all i simultaneously with high probability at least $(1 - \frac{N}{q})^\eta$, where η is the number of links carrying random combinations of the source processes; thus, $\eta \leq |\mathcal{E}|$.

The optimality of random coding in Corollary 7.1 comes from the fact that we allow coding across multicast connections $(S_i, T_j, \mathcal{X}(S_i))$'s – i.e., the source supernodes and the intermediate supernodes can randomly and uniformly select the coding coefficients. Thus, intermediate nodes within the network do not distinguish the flow from source S_i from that of S_j , and are allowed to encode them together randomly.

B. Disjoint Multicast

Theorem 7.2 (Disjoint Multicast Theorem): Given an acyclic network G with a set of connections $\mathcal{C} = \{(S, T_i, \mathcal{X}(S, T_i)) \mid i = 1, 2, \dots, K\}$ is called a *disjoint multicast* if $\mathcal{X}(S, T_i) \cap \mathcal{X}(S, T_j) = \emptyset$ for all $i \neq j$. Then, (G, \mathcal{C}) is solvable if and only if the min-cut between S and any subset of destinations $\mathcal{T}' \subseteq \mathcal{T}$ is at least $\sum_{T_i \in \mathcal{T}'} |\mathcal{X}(S, T_i)|$, i.e. $\text{mincut}(S, \mathcal{T}') \geq \sum_{T_i \in \mathcal{T}'} |\mathcal{X}(S, T_i)|$ for any $\mathcal{T}' \subseteq \mathcal{T}$.

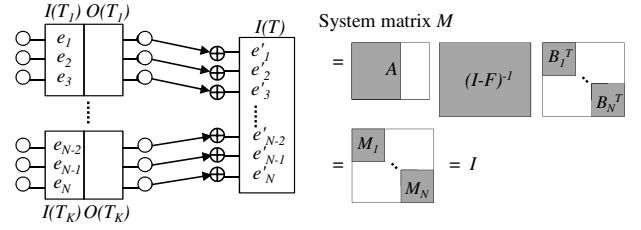


Fig. 13: Disjoint multicast problem can be converted into a single destination problem by adding a super-destination T . The system matrix M for the disjoint multicast problem is shown as well. Note that unlike the multicast problem in Figure 9 where $M = [M_1, M_2, \dots, M_N]$, the system matrix M is a diagonal concatenation of M_i 's.

Proof: Create a super-destination supernode T with $|I(T)| = \sum_{i=1}^K |I(T_i)|$, and an edge (e, e') from $e \in O(T_i)$, $i \in [1, K]$ to $e' \in I(T)$, as in Figure 13. This converts the problem of disjoint multicast to a single-source S , single-destination T problem with rate $\mathcal{X}(S, T) = \sum_{T' \in \mathcal{T}} |\mathcal{X}(S, T')|$. The $\text{mincut}(S, T) \geq |\mathcal{X}(S, T)|$; so, Theorem 6.1 applies. Thus, it is possible to achieve a communication of rate $\mathcal{X}(S, T)$ between S and T . Now, we have to guarantee that the receiver T_i is able to receive the exact subset of processes $\mathcal{X}(S, T_i)$. Since the system matrix to T is full rank, it is possible to carefully choose the encoding matrix A such that the system matrix M at super-destination supernode T is an identity matrix. This implies that for each edge from the output ports of T_i (for all i) to input ports of T is carrying a distinct symbol, disjoint from all the other symbols carried by those edges from output ports of T_j , for all $i \neq j$. Thus, by appropriately permuting the symbols at the source, S can deliver the desired processes to the intended T_i as shown in Figure 13. ■

Random linear network coding with a minor modification achieves the capacity for disjoint multicast. We note that only the source's encoding matrix A needs to be modified. The intermediate supernodes can randomly and uniformly select coding coefficients $\epsilon_{(e_i, (T_j, k))}$ and $\beta_{(e_i, e_j)}$ over all elements of \mathbb{F}_q . Once these coding coefficients at the intermediate supernodes are selected, S carefully chooses the encoding matrix A such that the system matrix corresponding to the receivers of the disjoint multicast is an identity matrix, as shown in Figure 13. To be more precise, when $\epsilon_{(e_i, (T_j, k))}$ and $\beta_{(e_i, e_j)}$ are randomly selected over elements of \mathbb{F}_q , with high probability, $(I - F)^{-1}B^T$ is full rank. Thus, there exists a matrix A such that $A(I - F)^{-1}B^T$ is an identity matrix I . Note that $A(I - F)^{-1}B^T$ does not need to be an identity matrix – it only needs to have a diagonal structure as shown in Figure 13; however, being an identity matrix is sufficient for proof of optimality.

We note another subtlety here. Theorem 7.2 holds precisely because we allow the intermediate nodes to code across all source processes, even they are destined for different receivers. This takes advantage of the fact that the single source can cleverly pre-code the data.

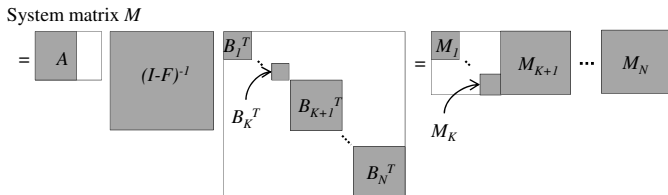


Fig. 14: The system matrix M for the two-level multicast problem. The structure of the system matrix M is a “concatenation” of the disjoint multicast problem (Figure 13) and the single multicast problem (Figure 9).

C. Two-level Multicast

Theorem 7.3 (Two-level Multicast Theorem): Given an acyclic network G with a set of connections $\mathcal{C} = \mathcal{C}_d \cup \mathcal{C}_m$ where $\mathcal{C}_d = \{(S, T_i, \mathcal{X}(S, T_i)) \mid \mathcal{X}(S, T_i) \cap \mathcal{X}(S, T_j) = \emptyset, i \neq j, i, j \in [1, K]\}$ is a disjoint multicast connection, and $\mathcal{C}_m = \{(S, T_i, \mathcal{X}(S)) \mid i \in [K+1, N]\}$ is a single source multicast connection. Then, (G, \mathcal{C}) is solvable if and only if the min-cut between S and any $\mathcal{T}' \subseteq \{T_1, \dots, T_K\}$ is at least $\sum_{T_i \in \mathcal{T}'} |\mathcal{X}(S, T_i)|$, and the min-cut between S and T_j is at least $|\mathcal{X}(S)|$ for $j \in [K+1, N]$.

Proof: We create a super-destination T for the disjoint multicast destinations as in the proof for Theorem 7.2. Then, we have a single multicast problem with receivers T and $\{T_i \mid i \in [K+1, N]\}$. Therefore, Theorem 6.2 applies. By choosing the appropriate matrix A , S can satisfy both the disjoint multicast and the single multicast requirements, as shown in Figure 14. ■

As in the disjoint multicast case, random linear network coding with a minor modification at the source achieves the capacity for two-level multicast. Note that, receivers $T_i, i \in [K+1, N]$ are of no concern – the source S can randomly choose coding coefficients $\alpha_{(i, e_j)}$ to achieve a full-rank system matrix M_i . Thus, S needs to carefully choose the encoding matrix A to satisfy the disjoint multicast constraint, which can be done as shown in Section VII-B.

Theorem 7.3 does not extend to a three-level multicast. Three-level multicast, in its simplest form, consists of connections $\{(S, T_i, \mathcal{X}(S, T_i)) \mid i \in [1, 3]\}$ where $\mathcal{X}(S, T_1) \subset \mathcal{X}(S, T_2) \subset \mathcal{X}(S, T_3)$.

D. General Connection Set

In the theorem below, we present sufficient conditions for solvability of a general connection set. This theorem does not provide necessary conditions, as shown in [29].

Theorem 7.4 (Generalized Min-cut Max-flow Theorem): Given an acyclic network G with a connection set \mathcal{C} , let $M = \{M_{i,j}\}$ where $M_{i,j}$ is the system matrix for source processes $\mathcal{X}(S_i)$ to destination processes $\mathcal{Z}(T_j)$. Then, (G, \mathcal{C}) is solvable if there exists an assignment of $\alpha_{(i, e_j)}, \epsilon_{(e_i, (T_j, k))}$, and $\beta_{(e_i, e_j)}$ such that

- 1) $M_{i,j} = 0$ for all $(S_i, T_j, \mathcal{X}(S_i, T_j)) \notin \mathcal{C}$,
- 2) Let $(S_{\sigma(i)}, T_j, \mathcal{X}(S_{\sigma(i)}, T_j)) \in \mathcal{C}$ for $i \in [1, K(j)]$. Thus, this is the set of connections with T_j as a receiver. Then, $[M_{\sigma(1), j}^T, M_{\sigma(2), j}^T, \dots, M_{\sigma(K_j), j}^T]$ is a $|\mathcal{Z}(T_j)| \times |\mathcal{Z}(T_j)|$ nonsingular system matrix.

Proof: Note that $[M_{\sigma(1), j}^T, M_{\sigma(2), j}^T, \dots, M_{\sigma(K_j), j}^T]$ is a system matrix for source processes $\mathcal{X}(S_{\sigma(i)})$, $i \in [1, K(j)]$, to destination processes $\mathcal{Z}(T_j)$.

Condition 2) states the Min-cut Max-flow condition; thus, is necessary to establish the connections. Condition 1) states that the destination supernode T_j should be able to distinguish the information it is intended to receive from the information that may have been mixed into the flow it receives. These two conditions are sufficient to establish all connections in \mathcal{C} . The proof is similar to that of Theorem 6 in [4]. ■

VIII. NETWORK WITH RANDOM ERASURES

We consider the algebraic ADT problem where links may fail randomly, and cause erasures. Wireless networks are stochastic in nature, and random erasures occur dynamically over time. However, the original ADT network models noise deterministically with parallel noise-free bit-pipes. As a result, the min-cut (Definition 5.1) and the network code [12][14][28], which depend on the hard-coded representation of noise, have to be recomputed every time the network changes.

We show that the algebraic framework for the ADT network is robust against random erasures and failures. First, we show that for some set of link failures, the network code remains successful. This translates to whether the system matrix M preserves its full rank even after a subset of variables $\alpha_{(i, e_j)}, \epsilon_{(e_i, (D_j, k))}$, and $\beta_{(e_i, e_j)}$ associated with the failed links is set to zero. Second, we show that the specific instance of the system matrix M and its rank are not as important as the *average* rank(M) when computing the time average min-cut. Note that the original min-cut definition (Definition 5.1) requires an optimization over an exponential number of cuts for every time step to find the average min-cut. We shall use the results from [30] to show that random linear network coding achieves the time-average min-cut; thus, is capacity-achieving.

We assume that any link within the network may fail. Given an ADT network G and a set of link failures f , G_f represents the network G experiencing failures f . This can be achieved by deleting the failing links from G , which is equivalent to setting the coding variables in $B(f)$ to zero, where $B(f)$ is the set of coding variables associated with the failing links. We denote M be the system matrix for network G . Let M_f be the system matrix for the network G_f . We do not assume that the link failures are static; thus, we can consider a static link failure patterns, a distribution over link failures patterns, or a sequence f_1, f_2, f_3, \dots of link failures.

A. Robust against Random Erasures

Given an ADT network problem (G, \mathcal{C}) , let \mathcal{F} be the set of all link failures such that, for any $f \in \mathcal{F}$, the problem (G_f, \mathcal{C}) is solvable. The solvability of a given (G_f, \mathcal{C}) can be verified using resulting in Sections VI and VII. We are interested in static solutions, where the network is oblivious of f . In other words, we are interested in finding the set of link failures such that the network code is still successful in delivering the source

processes to the destinations. For a multicast connection, we show the following surprising result.

Theorem 8.1 (Static Solution for Random Erasures):

Given an ADT network problem (G, \mathcal{C}) with a multicast connection $\mathcal{C} = \{(S, T_1, \mathcal{X}(S)), (S, T_2, \mathcal{X}(S)), \dots, (S, T_N, \mathcal{X}(S))\}$, there exists a *static* solution to the problem (G_f, \mathcal{C}) for all $f \in \mathcal{F}$. In other words, there exists a *fixed* network code that achieves the multicast rate despite any failures $f \in \mathcal{F}$.

Proof: By Theorem 6.2, we know that for any given $f \in \mathcal{F}$, the problem (G_f, \mathcal{C}) is solvable; thus, there exists a code $\det(M_f) \neq 0$. Now, we need to show that there exists a code such that $\det(M_f) \neq 0$ for all $f \in \mathcal{F}$ simultaneously. This is equivalent to finding a non-zero solution to the following polynomial: $\prod_{f \in \mathcal{F}} \det(M_f) \neq 0$. Reference [11] showed that if the field size is large enough ($q > |\mathcal{F}||\mathcal{T}| = |\mathcal{F}|N$), then there exists an assignment of $\alpha_{(i,e_j)}, \epsilon_{(e_i,(D_j,k))}$, and $\beta_{(e_i,e_j)}$ such that $\det(M_f) \neq 0$ for all $f \in \mathcal{F}$. ■

Corollary 8.1 (Random Coding against Random Erasures):

Consider an ADT network problem with a multicast connection $\mathcal{C} = \{(S, T_1, \mathcal{X}(S)), (S, T_2, \mathcal{X}(S)), \dots, (S, T_N, \mathcal{X}(S))\}$, which is solvable under link failures f , for all $f \in \mathcal{F}$. Then, random linear network coding, where some or all code variables $\alpha_{(i,e_j)}, \epsilon_{(e_i,(D_j,k))}$, and $\beta_{(e_i,e_j)}$ are chosen independently and uniformly over all elements of \mathbb{F}_q guarantees decodability at destination supernodes T_i for all i simultaneously and remains successful regardless of the failure pattern $f \in \mathcal{F}$ with high probability at least $(1 - \frac{N|\mathcal{F}|}{q})^\eta$, where η is the number of links carrying random combinations of the source processes.

Proof: Given a multicast connection that is feasible under any link failures $f \in \mathcal{F}$, [11] showed that random linear network coding achieves the capacity for multicast connections, and is robust against any link failures $f \in \mathcal{F}$ with high probability $(1 - \frac{N|\mathcal{F}|}{q})^\eta$. ■

We note that it is unclear whether this can be extended to the non-multicast connections, as noted in [4]. Reference [4] shows a simple example network in which no static solution is available for a set of feasible failure patterns.

B. Time-average Min-cut

In this section, we study the time-average behavior of the ADT network, given random erasures. We use techniques from [30], which studies reliable communication over lossy networks with network coding.

Consider an ADT network G . In order to study the time-average steady state behavior, we introduce erasure distributions. Let \mathcal{F}' be a set of link failure patterns in G . A set of link failures $f \in \mathcal{F}'$ may occur with probability p_f .

Theorem 8.2 (Min-cut for Time-varying Network):

Assume an ADT network G in which link failure pattern $f \in \mathcal{F}'$ occurs with probability p_f . Then, the average min-cut between two supernodes S and T in G , $\text{mincut}_{\mathcal{F}'}(S, T)$ is

$$\text{mincut}_{\mathcal{F}'}(S, T) = \sum_{f \in \mathcal{F}'} p_f \left(\max_{\alpha_{(i,e)}, \beta_{(e',e)}, \epsilon_{(e',i)}} \text{rank}(M_f) \right).$$

Proof: By Theorem 5.1, we know that at any given time instance with failure pattern f , the min-cut between S and T is given by $\max_{\alpha_{(i,e)}, \beta_{(e',e)}, \epsilon_{(e',i)}} \text{rank}(M_f)$. Then, the above statement follows naturally by taking a time average of the min-cut values between S and T . ■

The key difference between Theorems 8.1 and 8.2 is that in Theorem 8.1, any $f \in \mathcal{F}$ may change the network topology as well as min-cut but $\text{mincut}(S, T) \geq |\mathcal{X}(S)|$ holds for all $f \in \mathcal{F}$ – i.e. (G_f, \mathcal{C}) is assumed to be solvable. However, in Theorem 8.2, we make no assumption about the connection as we are evaluating the average value of the min-cut.

Unlike the case of static ADT networks, with random erasures, it is necessary to maintain a queue at each supernode in the ADT network. This is because, if a link fails when a supernode has data to transmit on it, then it will have to wait until the link recovers. In addition, a transmitting supernode needs to be able to learn whether a packet has been received by the next hop supernode, and whether it was innovative – this can be achieved using channel estimation, feedback and/or redundancy. In the original ADT network, the issue of feedback was removed by assuming that the links are noiseless bit-pipes. We present the following corollaries under these assumptions.

Corollary 8.2 (Multicast in Time-varying Network):

Consider an ADT network G and a multicast connection $\mathcal{C} = \{(S, T_1, \mathcal{X}(S)), \dots, (S, T_N, \mathcal{X}(S))\}$. Assume that failures occur where failure pattern $f \in \mathcal{F}'$ occurs with probability p_f . Then, the multicast connection is feasible if and only if $\text{mincut}_{\mathcal{F}'}(S, T_i) \geq |\mathcal{X}(S)|$ for all i .

Proof: Reference [30] shows that the multicast connection is feasible if and only $\text{mincut}_{\mathcal{F}'}(S, T_i) \geq |\mathcal{X}(S)|$ for all i . The proof in [30] relies on the fact that every supernode behaves like a stable $M/M/1$ queuing system in steady-state, and thus, the queues (or the number of innovative packets to be sent to the next hop supernode) has a finite mean if the network is run for sufficiently long period of time. ■

Corollary 8.3 (Random Coding for Time-varying Network):

Consider (G, \mathcal{C}) where \mathcal{C} is a multicast connection. Assume failure pattern $f \in \mathcal{F}'$ occurs with probability p_f . Then, random linear coding, where some or all code variables $\alpha_{(i,e_j)}, \beta_{(e_i,e_j)}, \epsilon_{(e_i,(D_j,k))}$ are chosen over all elements of \mathbb{F}_q guarantees decodability at destination nodes T_i for all i simultaneously with arbitrary small error probability.

Proof: This is a direct consequence of Corollary 8.2 and results in [11][30]. ■

IX. NETWORK WITH CYCLES

ADT networks are acyclic, with links directed from the source supernodes to the destination supernodes. However, wireless networks intrinsically have cycles as wireless links are bi-directional by nature. In this section, we extend the ADT network model to networks with cycles. In order to incorporate cycles, we need to introduce the notion of time – since, without the notion of time, the network with cycles may not be causal. To do so, we introduce delay on the links. As in [4], we model each link to have the same delay, and express the network random processes in the delay variable D .

$$\begin{pmatrix} 1 & 0 & D & 0 & 0 & D & D^2\beta_{(e_3,e_7)} & 0 & D^2\beta_{(e_6,e_9)} & D^2\beta_{(e_6,e_{10})} & D^3\beta_{(e_6,e_9)} & D^3\beta_{(e_3,e_7)} + D^3\beta_{(e_6,e_{10})} \\ 0 & 1 & 0 & D & 0 & 0 & D^2\beta_{(e_4,e_7)} & 0 & 0 & 0 & 0 & D^3\beta_{(e_4,e_7)} \\ 0 & 0 & 1 & 0 & 0 & 0 & D\beta_{(e_3,e_7)} & 0 & 0 & 0 & 0 & D^2\beta_{(e_3,e_7)} \\ 0 & 0 & 0 & 1 & 0 & 0 & D\beta_{(e_4,e_7)} & 0 & 0 & 0 & 0 & D^2\beta_{(e_4,e_7)} \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & D\beta_{(e_6,e_9)} & D\beta_{(e_6,e_{10})} & D^2\beta_{(e_6,e_9)} & D^2\beta_{(e_6,e_{10})} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & D \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & D & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & D \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Fig. 15: 12×12 matrix $(I - DF)^{-1}$ for network in Figure 4. The matrix F can be found in Figure 11.

We define $X_t(S, i)$ and $Z_t(T, j)$ to be the i -th and j -th binary random process generated at source S and received at destination T at time t , for $t = 1, 2, \dots$. We define $Y_t(e)$ to be the process on edge e at time $t = 1, 2, \dots$, respectively. We express the source processes as a power series in D , $\mathcal{X}(S, D) = [X(S, 1, D), X(S, 2, D), \dots, X(S, \mu(S), D)]$ where $X(S, i, D) = \sum_{t=0}^{\infty} X_t(S, i)D^t$. Similarly, we express the destination random processes $\mathcal{Z}(T, D) = [Z(T, 1, D), \dots, Z(T, \nu(Z), D)]$ where $Z(T, i, D) = \sum_{t=0}^{\infty} Z_t(T, i)D^t$. In addition, we express the edge random processes as $Y_t(e, D) = \sum_{t=0}^{\infty} Y_t(e)D^t$. Then, we can rewrite Equations (1) and (2) as

$$Y_{t+1}(e) = \sum_{e' \in I(V)} \beta_{(e', e)} Y_t(e') + \sum_{X_t(S, i) \in \mathcal{X}(S)} \alpha_{(i, e)} X_t(S, i).$$

Furthermore, the output processes $Z_t(T, i)$ can be rewritten as

$$Z_{t+1}(T, i) = \sum_{e' \in I(T)} \epsilon_{e', (T, i)} Y_t(e').$$

Using this formulation, we can extend the results from [4] to ADT networks with cycles. We show that a system matrix $M(D)$ captures the input-output relationships of the ADT networks with delay and/or cycles.

Theorem 9.1: Given a network $G = (\mathcal{V}, \mathcal{E})$, let $A(D)$, $B(D)$, and F be the encoding, decoding, and adjacency matrices, as defined here:

$$A_{i,j} = \begin{cases} \alpha_{(i, e_j)}(D) & \text{if } e_j \in O(S) \text{ and } X(S, i) \in \mathcal{X}(S), \\ 0 & \text{otherwise.} \end{cases}$$

$$B_{i, (T_j, k)} = \begin{cases} \epsilon_{(e_i, (T_j, k))}(D) & \text{if } e_i \in I(T_j), Z(T_j, k) \in \mathcal{Z}(T_j), \\ 0 & \text{otherwise.} \end{cases}$$

and F as in Equation (5). The variables $\alpha_{(i, e_j)}(D)$ and $\epsilon_{(e_i, (T_j, k))}(D)$ can either be constants or rational functions in D . Then, the system matrix of the ADT network with delay (and thus, may include cycles) is given by

$$M(D) = A(D) \cdot (I - DF)^{-1} \cdot B(D)^T. \quad (9)$$

Proof: The proof for this is similar to that of Theorem 4.1. \blacksquare

Similar to Section IV, $(I - DF)^{-1}$ represents the impulse response of the network with delay. This is because the series $I + DF + D^2F^2 + D^3F^3 + \dots$ represents the connectivity of the network while taking delay into account. For example, F^k has a non-zero entry if there exists a path of length k between two port. Now, since we want to represent the time associated

with traversing from port e_i to e_j , we use $D^k F^k$, where D^k signifies that the path is of length k . Thus, $(I - DF)^{-1} = I + DF + D^2F^2 + D^3F^3 + \dots$ is the impulse response of the network with delay. An example of $(I - DF)^{-1}$ for the example network in Figure 7 is shown in Figure 15.

Using the system matrix $M(D)$ from Theorem 9.1, we can extend Theorem 6.1, Theorem 6.2, Theorem 7.1, Theorem 7.2, and Theorem 7.3 to ADT networks with cycles/delay. However, there is a minor technical change. We now operate in a different field – instead of having coding coefficients from the finite field \mathbb{F}_q , the coding coefficients $\alpha_{(i, e_j)}(D)$ and $\epsilon_{((e_i, (T_j, k)))}(D)$ are now from $\mathbb{F}_q(D)$, the field of rational functions of D . We shall not discuss the proofs in detail; however, this is a direct application of the results in [4].

X. CODE CONSTRUCTION FOR MULTICAST CONNECTION

As presented in Sections VI-IX, random linear network coding, which is completely distributed, requires randomization. As a result, random linear network coding achieves the capacity for the ADT network model *with high probability*. Similarly, [24] introduced a distributed binary-vector network code, called *permute-and-add*, in which each node randomly and uniformly selects a permutation matrix from all such matrices for its coding operation. In [24], they show that permute-and-add is still optimal for multicast connections – *i.e.* achieves the capacity with high probability. Thus, network codes in \mathbb{F}_q , $q \geq 2$, can be converted to vector code in binary field, $(\mathbb{F}_2)^{\log_2(q)}$, without loss in performance. As a result, permute-and-add can be applied to the ADT network model. However, it is important to note that a randomized, distributed approach does not *guarantee decodability* (with probability 1).

In this section, we propose an efficient code construction for multicast connection in ADT network, which *guarantees all destination supernodes to decode if the connection is feasible*. Furthermore, we only require that there be some *local* coordination among neighboring supernodes (more precisely, among supernodes within a layer), as we shall discuss in Section X-C2.

Given an acyclic ADT network $G = (\mathcal{V}, \mathcal{E})$ and a multicast connection $\mathcal{C} = \{(S, T_i, \mathcal{X}(S)) | T_i \in \mathcal{T}\}$, we consider the problem of efficiently finding an assignment for $\alpha_{(i, e_j)}$, $\epsilon_{(e_i, (D_j, k))}$, and $\beta_{(e_i, e_j)}$ such that the system matrix M is invertible in \mathbb{F}_q . Note that we assume that (G, \mathcal{C}) is solvable – *i.e.* $\text{mincut}(S, T_i) \geq R$ for all i , where $R = |\mathcal{X}(S)|$ is the multicast rate. We shall consider $R = \min_{T_i \in \mathcal{T}} \text{mincut}(S, T_i)$.

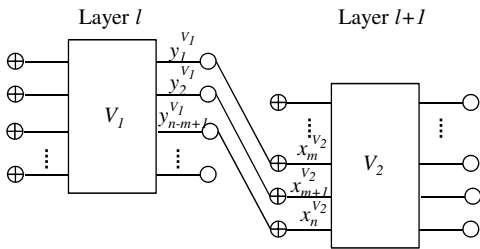


Fig. 16: Supernodes V_1 and V_2 .

For the given multicast rate R , we define the set

$$\mathcal{V}(R) \triangleq \{V \in \mathcal{V} \mid \text{mincut}(S, V) \geq R\}. \quad (10)$$

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 \mathcal{T} . We also note that the code designer may be oblivious of the exact location of the nodes in \mathcal{T} or $\mathcal{V}(R)$.

We assume that each supernode $V \in \mathcal{V}$ contains n input ports and n output ports, where

$$n = \left\lceil \frac{1}{2} \log \max_{(V_i, V_j) \in \mathcal{E}} \text{SNR}_{(V_i, V_j)} \right\rceil. \quad (11)$$

Therefore, we assume that $|I(V)| = |O(V)| = n$. For ease of notation, we distinguish the input ports of supernode V by

$$I(V) \triangleq \{x_1^V, \dots, x_n^V\}, \quad (12)$$

and the output ports by

$$O(V) \triangleq \{y_1^V, \dots, y_n^V\}. \quad (13)$$

The network is assumed to have λ layers, where all links are from layer l to layer $l+1$ for $l = 1, \dots, \lambda-1$. The source is at layer 1. We assume that there are at most N_{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 $l = 1, \dots, \lambda-1$. 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 $P(x_i^V)$ to be the set of output ports that have links incoming into the input port x_i^V of a supernode V in layer l . By assumption, $0 \leq |P(x_i^V)| \leq N_{\text{layer}} - i.e.$ there are at most N_{layer} edges incoming to x_i^V from output ports in layer $l-1$. We denote $C(y_j^V)$ to be the set of input ports that have links outgoing to the output port y_j^V of a supernode V in layer l . Note that $0 \leq |C(y_j^V)| \leq N_{\text{layer}}$ since y_j^V 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 $l+1$.

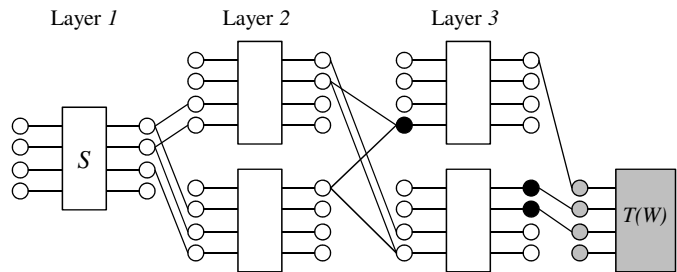


Fig. 17: A set W of input/output ports (in black) where $R = 3$. The supernode and the ports shaded in grey represent the virtual sink $T(W)$. Note that W is *not* regular.

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 contains a subset of V 's output ports, or a subset of V 's input ports, but not both, or no ports of V . 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 17 for an illustration.

Definition 10.1 (Regular set): The set W is said to be regular if $\text{mincut}(S, T(W)) = R$.

In Figure 17, 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. Each port has a coding vector 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

$$\mathbf{x}^{I(V)} = \{x_1^V, \dots, x_n^V\}, \quad (14)$$

where $x_i^V \in \mathbb{F}_q^R$ is the coding vector associated with the input port x_i^V . Similarly, we denote the coding vectors of the output

ports of supernode V by

$$\mathbf{y}^{O(V)} = \{\mathbf{y}_1^V, \dots, \mathbf{y}_n^V\}, \quad (15)$$

where $\mathbf{y}_j^V \in \mathbb{F}_q^R$ is the coding vector associated with the output port y_j^V . The coding vector \mathbf{y}_j^V is

$$\mathbf{y}_j^V = \sum_{i=1}^n \beta_{(x_i^V, y_j^V)} \mathbf{x}_i^V, \quad 1 \leq j \leq n, \quad (16)$$

where $\beta_{(x_i^V, y_j^V)} \in \mathbb{F}_q$ are the supernode internal coding coefficients, as described in Section IV.

Although we introduce new notation for the coding vectors for convenience, these vectors can still be captured by the system matrix $M = A(I - F)^{-1}B^T$. Consider an input port x_i^V . Assume that x_i^V is the p th port in the topological order. Then, the coding vector \mathbf{x}_i^V at this input port is p th column of the matrix $A(I - F)^{-1}$. This is because what input port x_i^V receives depends on two things: first, the sources encoding operations, represented by A ; second, the network connectivity and coding operations performed within the network, represented by $(I - F)^{-1}$. The same argument applies to the output ports y_j^V . Thus, the columns of $A(I - F)^{-1}$ is equal to the coding vectors at the corresponding input/output ports in the network.

We refer to this coding operation in Equation (16) as “forward coding”. Once the coding vector \mathbf{y}_j^V 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_i^V , let $\mathbf{y}_1, \dots, \mathbf{y}_p$ be the coding vectors of the output ports in the set $P(x_i^V)$. Then, the coding vector of the input port x_i^V is given by

$$\mathbf{x}_i^V = \sum_{j=1}^p k_j \mathbf{y}_j, \quad (17)$$

where $k_j \in \mathbb{F}_q$ are the virtual coding coefficients. Note that *only a single 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 10.2 (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 $\mathcal{C}_t = (\hat{S}, \hat{S}^c)$ 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_j^V is in \hat{S} if the coding coefficient $\beta_{(x_i^V, y_j^V)}$ of its supernode V have already been determined. An input port x_i^V is in \hat{S} if *all* of the virtual coefficients k_j of output ports in $P(x_i^V)$ have been determined. The input ports of the source are in \hat{S} .

A cut of ports \mathcal{C}_t is not necessarily a cut of supernodes. In a cut of ports, ports of the same supernode can be in two different parts of the cut \mathcal{C}_t . We do, however, restrict ourselves to a specific type of cuts of ports – all the input ports of a specific supernode are on the same side of the cut, and all the output ports are on the same side of the cut.

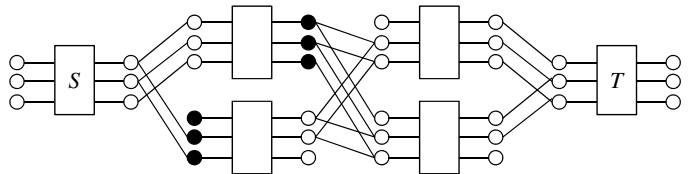


Fig. 18: An example of $\mathcal{Q}_{\mathcal{C}_t}$ (in black).

Definition 10.3 (Boundary set): A subset of ports $\mathcal{Q}_{\mathcal{C}_t}$ is a boundary set if

$$\mathcal{Q}_{\mathcal{C}_t} = \{\text{Input ports in } \hat{S} \text{ with outputs in } \hat{S}^c\} \cup \{\text{Output ports in } \hat{S} \text{ with edges outgoing to } \hat{S}^c\}.$$

Since the topological order proceeds from top to bottom at each layer, if the boundary set contains the input ports of a certain supernode V at layer l , then it will also contain the input ports of the supernodes that are above supernode V at layer l . Similarly, if the boundary set contains the output ports of a certain supernode V then it will also contain the output ports of the supernodes that are above it at layer l . Figure 18 shows an example of a boundary set, $\mathcal{Q}_{\mathcal{C}_t}$.

Lemma 10.1: $R \leq n \leq |\mathcal{Q}_{\mathcal{C}_t}| \leq nN_{\text{layer}}$.

Proof: Consider a cut of supernodes Ω where source supernode $\Omega = \{S\}$ and $\Omega^c = \mathcal{V} \setminus \{S\}$. Since S has n output ports, we conclude that $R \leq n$ since R is upper bounded by the min-cut. By definition, the output ports of a supernode V are restricted to be on the same side of the cut \mathcal{C}_t . The same is true for the input ports of a supernode. It follows that there are at least n ports in $\mathcal{Q}_{\mathcal{C}_t}$ – i.e. $n \leq |\mathcal{Q}_{\mathcal{C}_t}|$. Since the network is assumed to be layered, and the maximal number of supernodes in a layer is N_{layer} , it follows from the definition of $\mathcal{Q}_{\mathcal{C}_t}$ that $|\mathcal{Q}_{\mathcal{C}_t}| \leq nN_{\text{layer}}$. ■

The code construction considers each subset of R ports in $\mathcal{Q}_{\mathcal{C}_t}$. Some of the subsets we shall consider are regular sets. Define \mathcal{L}_t by

$$\mathcal{L}_t = \{\text{Regular subsets of } \mathcal{Q}_{\mathcal{C}_t} \text{ of size } R\}. \quad (18)$$

The number of subsets in \mathcal{L}_t is upper bounded by

$$|\mathcal{L}_t| \leq \binom{|\mathcal{Q}_{\mathcal{C}_t}|}{R} \leq \binom{nN_{\text{layer}}}{R}. \quad (19)$$

Code Invariant: Ensure that at each stage of the code construction, each subset in \mathcal{L}_t is associated with linearly independent coding vectors.

Lemma 10.2: Maintaining the invariant of the algorithm is sufficient to ensure the decodability of the code at rate R .

Proof: For a (non-virtual) sink $T_j \in \mathcal{T}$, let $W \subseteq O(T_j)$ be the R upper output ports of T_j . By the definition of R , we have $\text{mincut}(S, T_j) \geq R$. We connect the ports in W to a virtual sink $T(W)$ with R edges, where the i th output port of T_j is connected to the i th input port of $T(W)$. It follows that $R = \text{mincut}(S, T(W))$. Therefore, the set W is regular.

The invariant ensures that the set W will be associated with linearly independent coding vectors. It follows that T_j will be able to decode the data, as required by the code. The same argument can be applied to all nodes in $\mathcal{V}(R)$. The

linearly independent vectors of the regular sets can be used to reconstruct the data of the source by matrix inversion¹. ■

As the algorithm proceeds, ports may leave or enter $\mathcal{Q}_{\mathcal{C}_t}$. The list \mathcal{L}_t is then updated accordingly, as we shall discuss in the following sections.

C. Algorithm Description

The algorithm starts from the upper R input ports of the source S with the standard basis as their coding vectors. The lower input ports of the source have the zero vector as their coding vectors. The source bits are mapped into the source symbols in the field \mathbb{F}_q , where $q = 2^k$ for some k . The transmission is over block length $k = \log |\mathbb{F}_q|$. The vector of source symbols is $\mathbf{u} = (u_1, \dots, u_R)^T$, where $u_i \in \mathbb{F}_q$. The symbol received by a port with coding vector $\mathbf{x} \in \mathbb{F}_q^R$ is $\mathbf{x}^T \mathbf{u}$.

Trivially, the invariant of the algorithm is maintained for the source S . At each step of the algorithm, we proceed to the next port in the topological order, determine and update the followings.

- 1) The coding coefficients for the new port (and thus the coding vectors).
- 2) The updated list \mathcal{L}_t according to the new cut \mathcal{C}_t .

To do so, we shall treat the input and the output ports separately, as discussed in the subsequent sections. For each layer, the coding coefficients for the input ports have to be determined before the coding coefficients for the output ports. We start by considering coding for the output ports, assuming that the coding vectors at the input ports are given.

1) **Coding for Output Ports:** Assume that at time t , the topological order has reached supernode V . For the output ports, we assume that in the topological order, y_j^V precedes y_k^V if $j \leq k$. Consider a certain subset $W \in \mathcal{L}_t$. Some of the ports in this subset can be input ports and some of them output ports, as is the case in Figure 18. This can occur if the topological order has already reached the output ports of several supernodes in the layer, while other output ports at the same layer have not yet been reached. Suppose that the ports in a subset of the list are given by $W = \{w_1, \dots, w_R\}$ and their coding vectors are given by

$$\mathbf{W} = \{\mathbf{w}_1, \dots, \mathbf{w}_R\}. \quad (20)$$

If the set W contains $p \geq 1$ input ports from $I(V)$, then the subset W has to be updated. After the coding of the output ports of supernode V , 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 $\{w_1, \dots, w_p\} = \{x_1^V, \dots, x_p^V\}$. We choose a set of size p from $O(V)$ and denote the set by $\{w'_1, \dots, w'_p\}$. There are $\binom{n}{p}$ such possible sets.

We now update list \mathcal{L}_t to \mathcal{L}_{t+1} . In \mathcal{L}_{t+1} , we replace $W = \{w_1, \dots, w_R\}$ with $W' = \{w'_1, \dots, w'_p, w_{p+1}, \dots, w_R\}$ for all $\binom{n}{p}$ choices of $\{w'_1, \dots, w'_p\}$. 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 10.3: Consider a subset $W \in \mathcal{L}_t$, 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 $\beta_{(x_i^V, y_j^V)} \in \mathbb{F}_q$ for $1 \leq i \leq n, 1 \leq j \leq n$ such that the coding vectors of a subset $W' \in \mathcal{L}_{t+1}$ are linearly independent.

Proof: The coding vectors of W' are

$$\mathbf{W}' = \left\{ \sum_{i=1}^p \beta_{x_i^V, y_1^V} \mathbf{x}_i^V + \mathbf{v}_1, \dots, \sum_{i=1}^p \beta_{x_i^V, y_p^V} \mathbf{x}_i^V + \mathbf{v}_p, \right. \\ \left. \mathbf{w}_{p+1}, \dots, \mathbf{w}_R \right\},$$

where $\mathbf{v}_i, i = 1, \dots, p$, are the contributions of the coding vectors of the input ports in $I(V) \setminus W$. The \mathbf{v}_i 's are assumed to be fixed. Since $W \in \mathcal{L}_t$, invoking the inductive hypothesis, the set of vectors $\mathbf{W} = \{\mathbf{x}_1^V, \dots, \mathbf{x}_p^V, \mathbf{w}_{p+1}, \dots, \mathbf{w}_R\}$ is a basis. We need to determine under which conditions the subset \mathbf{W}' is also a basis.

Consider the equation

$$\varphi_1 \left(\sum_i \beta_{x_i^V, y_1^V} \mathbf{x}_i^V + \mathbf{v}_1 \right) + \dots + \varphi_p \left(\sum_i \beta_{x_i^V, y_p^V} \mathbf{x}_i^V + \mathbf{v}_p \right) \\ + \varphi_{p+1} \mathbf{w}_{p+1} + \dots + \varphi_R \mathbf{w}_R = 0. \quad (21)$$

In order for \mathbf{W}' to be a basis, we find a sufficient condition for $\varphi_1 = \varphi_2 = \dots = \varphi_R = 0$ to be the only solution to (21). First, we express \mathbf{v}_i in the basis \mathbf{W} as

$$\mathbf{v}_i = \gamma_{1,i} \mathbf{x}_1^V + \dots + \gamma_{p,i} \mathbf{x}_p^V + \gamma_{p+1,i} \mathbf{w}_{p+1} + \dots + \gamma_{R,i} \mathbf{w}_R,$$

where γ are not all zeros. Substituting and rearranging the terms of (21) yields

$$\left[\varphi_1 (\beta_{x_1^V, y_1^V} + \gamma_{1,1}) + \dots + \varphi_p (\beta_{x_1^V, y_p^V} + \gamma_{1,p}) \right] \mathbf{x}_1^V + \dots \\ + \left[\varphi_1 (\beta_{x_p^V, y_1^V} + \gamma_{p,1}) + \dots + \varphi_p (\beta_{x_p^V, y_p^V} + \gamma_{p,p}) \right] \mathbf{x}_p^V \\ + [\varphi_1 \gamma_{p+1,1} + \dots + \varphi_p \gamma_{p+1,p} + \varphi_{p+1}] \mathbf{w}_{p+1} + \dots \\ + [\varphi_1 \gamma_{R,1} + \dots + \varphi_p \gamma_{R,p} + \varphi_R] \mathbf{w}_R = 0. \quad (22)$$

Since \mathbf{W} is a basis, it follows that

$$\varphi_1 (\beta_{x_1^V, y_1^V} + \gamma_{1,1}) + \dots + \varphi_p (\beta_{x_1^V, y_p^V} + \gamma_{1,p}) = 0 \\ \vdots \\ \varphi_1 (\beta_{x_p^V, y_1^V} + \gamma_{p,1}) + \dots + \varphi_p (\beta_{x_p^V, y_p^V} + \gamma_{p,p}) = 0 \quad (23)$$

This can be written in matrix form as

$$\begin{pmatrix} \beta_{x_1^V, y_1^V} + \gamma_{1,1} & \dots & \beta_{x_1^V, y_p^V} + \gamma_{1,p} \\ \vdots & \ddots & \vdots \\ \beta_{x_p^V, y_1^V} + \gamma_{p,1} & \dots & \beta_{x_p^V, y_p^V} + \gamma_{p,p} \end{pmatrix} \begin{pmatrix} \varphi_1 \\ \vdots \\ \varphi_p \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}. \quad (24)$$

We note that $\varphi_1 = \dots = \varphi_p = 0$ is the only solution of (21) if and only if the matrix in Equation (24) is non-singular. For a $p \times p$ matrix over a field \mathbb{F}_q , the total number of matrices

¹The coding vectors at the edges incoming into each sink can be made known to the sink by the source transmitting the identity matrix. Thus, the sink is informed of the matrix which it must invert for decoding. This idea is similar to the one used for network coding [31].

is q^{p^2} . Using a combinatorial argument, the number of non-singular matrices is

$$(q^p - 1)(q^p - q)(q^p - q^2) \cdots (q^p - q^{p-1}) \quad (25)$$

$$= q^{p^2} \left(1 - \frac{1}{q^p}\right) \left(1 - \frac{1}{q^{p-1}}\right) \cdots \left(1 - \frac{1}{q}\right) \geq q^{p^2} \left(1 - \frac{1}{q}\right)^p.$$

Equation (25) 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 (24). If these conditions are maintained, then (21) becomes

$$\varphi_{p+1} \mathbf{w}_{p+1} + \cdots + \varphi_R \mathbf{w}_R = 0. \quad (26)$$

The only solution to this relation is $\varphi_{p+1} = \cdots = \varphi_R = 0$ since the vectors $\mathbf{w}_{p+1}, \cdots, \mathbf{w}_R$ are in the basis W and are therefore linearly independent.

We conclude that if the matrix in (24) is non-singular, then the vectors in \mathbf{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 $\beta_{x_i^y, y_j^y}$, for $1 \leq i \leq n, 1 \leq j \leq n$, such that the matrix is non-singular. ■

Lemma 10.4: If alphabet size $q > n \binom{n N_{\text{layer}}}{R}$, then there exists a set of coding coefficients $\beta_{x_i^y, y_j^y} \in \mathbb{F}_q$ for $1 \leq i \leq n, 1 \leq j \leq n$, such that all the subsets in \mathcal{L}_{t+1} have linearly independent coding vectors simultaneously.

Proof: The subset $W \in \mathcal{L}_t$ contains $p \geq 1$ input ports from $I(V)$. From (25), it follows that for a specific subset W' in \mathcal{L}_{t+1} , the number of non-singular matrices is at least

$$q^{p^2} \left(1 - \frac{1}{q}\right)^p \geq q^{p^2} \left(1 - \frac{p}{q}\right), \quad (27)$$

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

$$q^{p^2} - q^{p^2} \left(1 - \frac{p}{q}\right) = pq^{p^2-1} \leq nq^{n^2-1}. \quad (28)$$

In \mathcal{L}_{t+1} , there are at most $\binom{n N_{\text{layer}}}{R}$ newly added subsets. For each subset, there are at most nq^{n^2-1} choices of a set of coding coefficients $\beta_{x_i^y, y_j^y}$, for $1 \leq i \leq n, 1 \leq j \leq n$, such that the coding vector associated with the subset is linearly dependent. Therefore by the union bound there are at most $\binom{n N_{\text{layer}}}{R} nq^{n^2-1}$ choices of sets of coding coefficients such that at least one of the subsets in \mathcal{L}_{t+1} can have dependent coding vectors. The total number of choices of coding coefficients is q^{n^2} . Therefore, if $q > n \binom{n N_{\text{layer}}}{R}$, then we will have at least a single set of coding coefficients such that all the subsets in \mathcal{L}_{t+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 10.1: The invariant of the algorithm is maintained for the output ports.

Proof: By assumption, the invariant is maintained for the set $\mathcal{Q}_{\mathcal{L}_t(I)}$, which contains the input ports of the supernodes in layer l . We need to show that the invariant is maintained for the set $\mathcal{Q}_{\mathcal{L}_t(O)}$, which contains the output ports of the supernodes in layer l , where $t(I) < t(O)$. This follows by induction from Lemma 10.4. ■

The average complexity of this stage is computed using arguments similar to those in [15] for the network code construction. According to Lemma 10.4, we can choose the alphabet size at this stage to be $q = 2n \binom{n N_{\text{layer}}}{R}$. It follows from the proof of Lemma 10.4 that the probability of failure when the coding coefficients are chosen randomly is upper bounded by

$$P_f \leq \frac{\binom{n N_{\text{layer}}}{R} n q^{n^2-1}}{q^{n^2}} = \frac{\binom{n N_{\text{layer}}}{R} n}{q} = \frac{1}{2}. \quad (29)$$

Therefore, the expected number of trials until the vectors in \mathbf{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 $N_{\text{layer}} n^2$. In our case, the equivalent to the number of sinks $|\mathcal{T}|$ in [15] is the size of \mathcal{L}_t , which is at most $\binom{n N_{\text{layer}}}{R}$. Therefore, similarly to the complexity in [15] for network coding, the average complexity for a layer is $O\left(\binom{n N_{\text{layer}}}{R} N_{\text{layer}} n^2 R\right)$. If the total number of layers is λ , then the total average complexity of finding the coding coefficients of the output ports is $O\left(\binom{n N_{\text{layer}}}{R} N_{\text{layer}} n^2 R \lambda\right)$.

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 X-C1. We need to update the coding coefficients of the input ports of layer $l + 1$. The list \mathcal{L}_t contains ports from layer l only. From the list \mathcal{L}_t , we choose an arbitrary subset

$$W = \{w_1, \cdots, w_R\} \in \mathcal{L}_t. \quad (30)$$

The set W' is a subset of R input ports at layer $l + 1$. Consider the bipartite network that consists of W and the input ports of layer $l + 1$, with edges from ports in W to ports in layer $l + 1$. Let $\mathcal{B}(W, W')$ denote the bipartite subgraph with vertices (W, W') and edges from ports in W to ports in W' . Let $G_{(W, W')}$ denote the incidence matrix for $\mathcal{B}(W, W')$. 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 \mathcal{L}_t 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 19, 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 10.1 since the upper and the lower ports in W' always receive the same symbol.

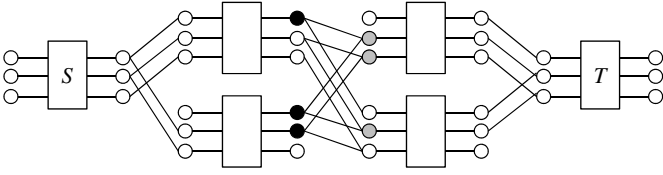


Fig. 19: Example of a non-regular W' . Ports in black are in W , and ports in grey are in W' .

Lemma 10.5: 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 [13][14][17]². Specifically, in [17], a path is defined as a disjoint set of edges (e_1, \dots, e_μ) where e_1 starts from the source, e_μ 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 [17], 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 [17] 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 definition of the underlying flow. This set of output ports will be chosen as W . The rank($G_{(W,W')}$) is 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 10.5.

Lemma 10.6: For a regular set W' , we can find coding coefficients such that the coding vectors \mathbf{W}' are linearly independent simultaneously if the alphabet size $q > 2^{\binom{n_{\text{layer}}}{R}}$.

Proof: By Lemma 10.5, 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 $G_{(W,W')}$ is full rank. The coding vectors of the output ports in W are given by

$$\mathbf{W} = \{\mathbf{w}_1, \dots, \mathbf{w}_R\}. \quad (31)$$

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

$$\mathbf{W}' = \{\mathbf{w}'_1, \dots, \mathbf{w}'_R\}. \quad (32)$$

The vector \mathbf{w}_i is in the form

$$\mathbf{w}_i = \sum_{j=1}^R g_{i,j} k_j \mathbf{w}_j + \tilde{\mathbf{w}}_i \quad (33)$$

where k_j are the virtual coefficients, and $\tilde{\mathbf{w}}_i$ 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 $G_{(W,W')}$.

²These works consider the problem of unicast communication. We can use their result for the proof of Lemma 10.5 since we are interested here in a single set W' with a corresponding virtual sink $T(W')$.

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

$$\varphi_1 \mathbf{w}'_1 + \dots + \varphi_R \mathbf{w}'_R = 0. \quad (34)$$

Combining with (33), and rearranging,

$$\begin{aligned} \varphi_1 \left(\sum_{j=1}^R g_{1,j} k_j \mathbf{w}_j \right) + \dots + \varphi_R \left(\sum_{j=1}^R g_{R,j} k_j \mathbf{w}_j \right) \\ = -\alpha_1 \tilde{\mathbf{w}}_1 - \dots - \alpha_{h_{\min}} \tilde{\mathbf{w}}_{h_{\min}}. \end{aligned} \quad (35)$$

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

$$\tilde{\mathbf{w}}_i = \gamma_{1,i} \mathbf{w}_1 + \dots + \gamma_{R,i} \mathbf{w}_R. \quad (36)$$

Combining with (35) and rearranging terms,

$$\begin{aligned} (\varphi_1 \gamma_{1,1} + \dots + \varphi_R \gamma_{1,R} + \varphi_1 g_{1,1} k_1 + \dots + \varphi_R g_{R,1} k_1) \mathbf{w}_1 \\ + \dots + (\varphi_1 \gamma_{R,1} + \dots + \varphi_R \gamma_{R,R} \\ + \varphi_1 g_{1,R} k_R + \dots + \varphi_R g_{R,R} k_R) \mathbf{w}_R = 0. \end{aligned} \quad (37)$$

Since W is a basis, it follows that

$$\begin{aligned} \varphi_1 \gamma_{1,1} + \dots + \varphi_R \gamma_{1,R} + \varphi_1 g_{1,1} k_1 + \dots + \varphi_R g_{R,1} k_1 &= 0 \\ &\vdots \\ \varphi_1 \gamma_{R,1} + \dots + \varphi_R \gamma_{R,R} \\ + \varphi_1 g_{1,R} k_R + \dots + \varphi_R g_{R,R} k_R &= 0, \end{aligned}$$

or in matrix notation,

$$\begin{pmatrix} \gamma_{1,1} + g_{1,1} k_1 & \dots & \gamma_{1,R} + g_{R,1} k_1 \\ \vdots & \ddots & \vdots \\ \gamma_{R,1} + g_{1,R} k_R & \dots & \gamma_{R,R} + g_{R,R} k_R \end{pmatrix} \begin{pmatrix} \varphi_1 \\ \vdots \\ \varphi_R \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \quad (38)$$

Let H denote the matrix on the left-hand side of (38). The zero vector is the only solution to (38) if and only if the matrix H is full rank. The determinant of the matrix H is a polynomial in the parameters $\{\gamma_{i,j}, k_j, g_{i,j}, \text{ for } 1 \leq i, j \leq R\}$. Denote the polynomial by $\Delta_{W,W'}(\beta_{i,j}, k_j, h_{i,j})$. When all $\gamma_{i,j} = 0$, 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:

$$\Delta_{W,W'}(\gamma_{i,j}, k_j, g_{i,j}) = \det(G_{(W,W')}) \prod_{j=1}^R k_j + \delta(\gamma_{i,j}, k_j, g_{i,j}) \quad (39)$$

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 \mathcal{P}_t 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.

$$P = \prod_{W': \exists (W, W') \in \mathcal{P}_t} \Delta_{W,W'}. \quad (40)$$

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 [4] proposes an algorithm to find a vector \underline{a} such that a given polynomial P evaluated at \underline{a} is not equal to zero, *i.e.* $P(\underline{a}) \neq 0$. We reproduce this algorithm from [4] in Algorithm 1 for completeness.

- Input** : A polynomial P in variables $\xi_1, \xi_2, \dots, \xi_n$;
integers $i = 1, t = 1$
- Output**: $\underline{a} = (a_1, a_2, \dots, a_n)$
- Step 1** : Find the maximal degree d of P in any variable ξ_i , and let i be the smallest number such that $2^i > d$;
- Step 2** : Find an element a_t in \mathbb{F}_{2^i} such that $P(\xi)|_{\xi_t=a_t} \neq 0$ and let $P \leftarrow P(\xi)|_{\xi_t=a_t}$;
- Step 3** : If $t = n$, then halt; otherwise, $t \leftarrow t + 1$ and go to **Step 2**;

Algorithm 1: Algorithm to find \underline{a} such that $P(\underline{a}) \neq 0$ [4].

In our scenario, the maximal degree of each variable in $\Delta_{W,W'}$ is 1 because of the structure of the matrix. It follows that the maximal degree of each variable in P is at most $\binom{nN_{layer}}{R}$. Therefore, $d = \binom{nN_{layer}}{R}$ and we can always choose $i = \lceil \log \binom{nN_{layer}}{R} \rceil$. It follows that an alphabet larger than $2^{\lceil \log \binom{nN_{layer}}{R} \rceil} \leq 2^{\binom{nN_{layer}}{R}}$ will ensure that there exists coding coefficients such that \mathbf{W}' are linearly independent. ■

Theorem 10.2: The invariant of the algorithm is maintained 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 10.5 and 10.6 to the first layer.

For the inductive step, assume that the statement holds for \mathcal{Q}_t , where \mathcal{Q}_t contains the output ports of the supernodes in layer l . Now, we show that the invariant is maintained for \mathcal{Q}_{t+1} , which contains the input ports of the supernodes in layer $l + 1$. If W' is a regular subset, then by Lemma 10.6, there is a coding assignment (according to the code construction presented) such that vectors in \mathbf{W}' are independent. Therefore, the invariant is maintained also for layer $l + 1$. ■

We now analyze the complexity. For each pair of sets (W, W') , we need to verify whether the incidence matrix $G_{(W,W')}$ of the bipartite graph $\mathcal{B}(W, W')$ is full rank. This can be performed by determinant computation in complexity $O(R^3)$. Therefore, the total complexity of this stage for a single layer is $O(R^3 \binom{nN_{layer}}{R})$.

For each variable of P , we require an iteration of the Algorithm 1. Each iteration of Algorithm 1 takes at most $2^{\lceil \log \binom{nN_{layer}}{R} \rceil}$ assignments need to be verified. There are at most N_{layer} supernodes at each layer. Therefore, the maximal number of output ports, which is also the number of variables k_j , is nN_{layer} . It follows that the complexity of the coding for input ports for a single layer is $O(\binom{nN_{layer}}{R}(nN_{layer} + R^3))$. Therefore, if we consider all λ layers, the complexity for the

coding for the input ports is $O(\binom{nN_{layer}}{R}\lambda(nN_{layer} + R^3))$. Combining the complexity for both input and output ports, it follows that the total complexity of the algorithm is $O(\binom{nN_{layer}}{R}N_{layer}n^2\lambda R)$.

XI. CONCLUSIONS

ADT networks [1][2] have drawn considerable attention for their potential to approximate the capacity of wireless relay networks. In this paper, we showed that the ADT network can be described well within the algebraic network coding framework [4]. This connection between ADT network and algebraic network coding allows the use of results on network coding to understand better the ADT networks.

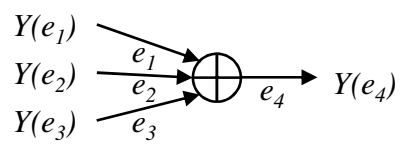
In this paper, we derived an algebraic definition of min-cut for the ADT networks, and provided an algebraic interpretation of the Min-cut Max-flow theorem for a single unicast/multicast connection in ADT networks. Furthermore, by taking advantage of the algebraic structure, we have shown feasibility conditions for a variety of set of connections \mathcal{C} , such as multiple multicast, disjoint multicast, and two-level multicast. We also showed optimality of linear operations for the connections listed above in the ADT networks, and showed that random linear network coding achieves the capacity. Furthermore, we extended the capacity characterization to networks with cycles and random erasures/failures. We proved the optimality of linear operations (as well as random linear network coding) for multicast connections in ADT networks with cycles. By incorporating random erasures into the ADT network model, we showed that random linear network coding is robust against failures 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 $O(\binom{nN_{layer}}{R}N_{layer}n^2\lambda R)$. The required field size is at most $q = n \binom{nN_{layer}}{R}$; thus, the block length required to represent a symbol is at most $k = \log(n \binom{nN_{layer}}{R})$. Our code construction 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 multiuser networks with receiver noise.

REFERENCES

- [1] A. S. Avestimehr, S. N. Diggavi, and D. N. C. Tse, "A deterministic approach to wireless relay networks," in *Proceedings of Allerton Conference on Communication, Control, and Computing*, September 2007.
- [2] —, "Wireless network information flow," in *Proceedings of Allerton Conference on Communication, Control, and Computing*, September 2007.
- [3] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, pp. 1204–1216, 2000.
- [4] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Transaction on Networking*, vol. 11, pp. 782–795, 2003.
- [5] G. Bresler, A. Parekh, and D. Tse, "The approximate capacity of the many-to-one and one-to-many Gaussian interference channels," *submitted to Transactions on Information Theory*, Sept. 2008.

- [6] A. S. Avestimehr and T. Ho, "Approximate capacity of the symmetric half-duplex Gaussian butterfly network," *ITW 2009, Volos, Greece, June 10-12, 2009*.
- [7] M. Kim and M. Médard, "Algebraic network coding approach to deterministic wireless relay networks," in *48th Allerton Conference on Communication, Control and Computing*, 2010.
- [8] E. Erez, Y. Xu, and E. M. Yeh, "Coding for the deterministic network model," in *ITA*, January 2010.
- [9] —, "Coding for the deterministic network model," in *48th Allerton Conference on Communication, Control and Computing*, 2010.
- [10] I. Maric, A. Goldsmith, and M. Médard, "Analog network coding in the high-SNR regime," in *Proceedings of ITA Workshop*, 2010.
- [11] T. Ho, M. Médard, R. Koetter, M. Effros, J. Shi, and D. R. Karger, "A random linear coding approach to multicast," *IEEE Transaction on Information Theory*, vol. 52, pp. 4413–4430, 2006.
- [12] A. Amaxudruz and C. Fragouli, "Combinatorial algorithms for wireless information flow," in *Proceedings of ACM-SIAM Symposium on Discrete Algorithms*, 2009.
- [13] S. M. Sadegh Tabatabaei Yazdi and S. A. Savari, "A combinatorial study of linear deterministic relay networks," 2009. [Online]. Available: <http://arxiv.org/abs/0904.2401v1.pdf>
- [14] M. X. Goemans, S. Iwata, and R. Zenklusen, "An algorithmic framework for wireless information flow," in *Proceedings of Allerton Conference on Communication, Control, and Computing*, September 2009.
- [15] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Transactions on Information Theory*, vol. 51, no. 6, pp. 1973–1982, June 2005.
- [16] R. Etkin, D. Tse, and H. Wang, "Gaussian interference channel capacity to within one bit," *IEEE Transactions on Information Theory*, Dec. 2008.
- [17] A. Amaxudruz and C. Fragouli, "Combinatorial algorithms for wireless information flow," in *SODA '09: Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2009, pp. 555–564.
- [18] R. Dougherty, C. Freiling, and K. Zeger, "Networks, matroids, and non-shannon information inequalities," *IEEE Transactions on Information Theory*, vol. 53, no. 6, June 2007.
- [19] M. Feder, D. Ron, and A. Tavory, "Bounds on linear codes for network multicast," *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 10, no. 33, 2003.
- [20] A. Rasala-Lehman and E. Lehman, "Complexity classification of network information flow problems," *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 142–150, New Orleans, Louisiana, January 2004.
- [21] C. Fragouli, E. Soljanin, and A. Shokrollahi, "Network coding as a coloring problem," *IEEE Annual Conference on Information Sciences and Systems (CISS 2004)*, Princeton, NJ, March 2004.
- [22] J. Ebrahimi and C. Fragouli, "Multicasting algorithms for deterministic networks," in *Proceedings of IEEE Information Theory Workshop (ITW)*, January 2010.
- [23] —, "Vector network coding," in *EPFL Technical Report*, February 2010.
- [24] S. Jaggi, Y. Cassuto, and M. Effros, "Low complexity encoding for network codes," in *Proceedings of International Symposium on Information Theory*, 2006.
- [25] S. Katti, I. Maric, A. Goldsmith, D. Katabi, and M. Médard, "Joint relaying and network coding in wireless networks," in *IEEE International Symposium on Information Theory*, 2007, pp. 1101–1105.
- [26] S. Ray, M. Médard, and J. Abounadi, "Noise-free multiple access networks over finite fields," in *Proceedings of Allerton Conference on Communication, Control, and Computing*, October 2003.
- [27] M. Effros, M. Médard, T. Ho, S. Ray, D. Karger, and R. Koetter, "Linear network codes: A unified framework for source channel, and network coding," in *Proceedings of the DIMACS workshop on network information theory (Invited paper)*, 2003.
- [28] E. Erez, Y. Xu, , and E. M. Yeh, "Coding for the deterministic network model," in *Proceedings of the Information Theory and Applications Workshop (ITA), Invited Paper*, January 2010.
- [29] R. Dougherty, C. Freiling, and K. Zeger, "Insufficiency of linear coding in network information flow," *IEEE Transaction on Information Theory*, vol. 51, pp. 2745–2759, 2005.
- [30] D. Lun, M. Médard, R. Koetter, and M. Effros, "On coding for reliable communication over packet networks," *Physical Communication*, vol. 1, no. 1, pp. 3–20, March 2008.
- [31] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," *41st Allerton Conference on Communication, Control and Computing*, Monticello, IL, October 2003.



$$Y(e_4) = \beta_1 Y(e_1) + \beta_2 Y(e_2) + \beta_3 Y(e_3)$$