The Encoding Complexity of Network Coding

Michael Langberg, *Member, IEEE*, Alexander Sprintson, *Member, IEEE*, and Jehoshua Bruck, *Fellow, IEEE*

***Abstract*—In the multicast network coding problem, a source s needs to deliver h packets to a set of k terminals over an underlying communication network G. The nodes of the multicast network can be broadly categorized into two groups. The first group incudes *encoding* nodes, i.e., nodes that generate new packets by combining data received from two or more incoming links. The second group includes *forwarding* nodes that can only duplicate and forward the incoming packets. Encoding nodes are, in general, more expensive due to the need to equip them with encoding capabilities. In addition, encoding nodes incur delay and increase the overall complexity of the network.**

Accordingly, in this paper, we study the design of multicast coding networks with a limited number of encoding nodes. We prove that in a directed acyclic coding network, the number of encoding nodes required to achieve the capacity of the network is bounded by h^3k^2. Namely, we present (efficiently constructible) network codes that achieve capacity in which the total number of encoding nodes is independent of the size of the network and is bounded by h^3k^2. We show that the number of encoding nodes may depend both on h and k by presenting acyclic coding networks that require $\Omega(h^2k)$ encoding nodes. In the general case of coding networks with cycles, we show that the number of encoding nodes is limited by the size of the minimum feedback link set, i.e., the minimum number of links that must be removed from the network in order to eliminate cycles. We prove that the number of encoding nodes is bounded by $(2B+1)h^3k^2$, where B is the minimum size of a feedback link set. Finally, we observe that determining or even crudely approximating the minimum number of required encoding nodes is an $\mathcal{NP}$-hard problem.

***Index Terms*—Coding networks, encoding links, encoding nodes, multicast, network coding.**

I. Introduction

THE goal of communication networks is to transfer information between source and destination nodes. Accordingly, the fundamental question that arises in network design is how to increase the amount of information transferred by the network. Recently, it has been shown that the ability of the network to transfer information can be significantly improved by employing the novel technique of *network coding* [1]–[3]. The idea is to allow the intermediate network nodes to combine data received over different incoming links. Nodes with coding capabilities are referred to as *encoding* nodes, in contrast to *forwarding* nodes that can only forward and duplicate incoming packets. The network coding approach extends traditional routing schemes, which include only forwarding nodes. The concept of network coding was introduced in a seminal paper by Ahlswede *et al.* [1] and immediately attracted significant attention from the research community. A large body of research focused on the multicast network coding problem where a source s needs to deliver h packets to a set T of k terminals over an underlying communication network G. It was shown in [1] that the capacity of the network, i.e., the maximum number of packets that can be sent between s and T, is bounded by the size of the minimum *cut*[1] that separates the source s and a terminal $t_i \in T$. Namely, a source s can transmit at rate h to a set T of terminals only if the size of the minimum cut separating s and any one of the terminals $t_i \in T$ is at least h. This combinatorial condition was shown to be sufficient by Li, Yeung, and Cai [2], and achievable by using linear network codes, i.e., codes in which each packet sent over the network is a linear combination of the original packets. In a subsequent work, Koetter and Médard [3] developed an algebraic framework for network coding and investigated linear network codes for directed graphs with cycles. This framework was used by Ho *et al.* [4] to show that linear network codes can be efficiently constructed by employing a randomized algorithm. Jaggi *et al.* [5] proposed a deterministic polynomial-time algorithm for finding feasible network codes for multicast networks.

Earlier work on network coding established a tight upper bound on the capacity of multicast networks and provided tools for constructing network codes that achieve capacity. However, optimization issues in network coding have received little attention from the research community. In general, the goal of network optimization is to minimize the amount of resources consumed by network connections. In this study, we focus on minimizing the total number of encoding nodes in multicast coding networks. More specifically, our goal is to find, for a given instance of the network coding problem, a feasible network code that requires as few encoding nodes as possible.

The problem of minimizing the number of encoding nodes is important for both theoretical and practical reasons. First, encoding nodes are, in general, more expensive than forwarding nodes, mostly because of the need to equip them with encoding capabilities. In addition, encoding nodes incur delay and increase the overall complexity of the network.

A. Contribution

The contribution of this paper can be summarized as follows. We prove the existence of efficiently constructible net-

Manuscript received March 15, 2005; revised December 20, 2005. This work was supported in part by the Caltech Lee Center for Advanced Networking and by the National Science Foundation under Grants ANI-0322475 and CCF-0346991. The material in this paper was presented in part at the IEEE International Symposium on Information theory, Adelaide, Australia, September 2005.

M. Langberg and J. Bruck are with the California Institute of Technology, Pasadena, CA 91125 USA (e-mail: mikel@cs.caltech.edu; bruck@paradise.caltech.edu).

A. Sprintson is with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843-3259 USA (e-mail: spalex@ece.tamu.edu).

Communicated by N. Cai, Guest Editor.

Digital Object Identifier 10.1109/TIT.2006.874434

[1]A cut (V_1, V_2) in graph $G(V, E)$ is a partition of V into two subsets V_1 and $V_2 = V \setminus V_1$. The size of the cut is determined by the number of links that leave a node in V_1 and enter a node in V_2. We say that a cut (V_1, V_2) separates nodes s and t if $s \in V_1$ and $t \in V_2$.

work codes in which the number of encoding nodes is independent of the size of the underlying graph G and depends only on the number of packets h and the number of terminals k. Our algorithm is simple and includes three basic steps: 1) Construct an auxiliary network by substituting each internal node by a "gadget" and removing all redundant links. The degree of any internal node in the auxiliary network is bounded by 3; 2) Find a feasible network code for the auxiliary network; 3) Reconstruct a network code for the original network. We show that such a procedure yields a network code that requires only h^3k^2 encoding nodes. We also show that in the worst case, the number of encoding nodes depends on both h and k. To that end, we present, for any values of h and k, a coding network that requires $\Omega(h^2k)$ encoding nodes.

We also consider the general case of coding networks with cycles. We show that in such networks, the number of encoding nodes required to enable transmission at rate h from a source s to k terminals depends on the size of the minimum feedback link set of the network, i.e., the minimum number of links that must be removed from the network in order to eliminate cycles. Specifically, we prove that the number of required encoding nodes is bounded by $(2B+1)h^3k^2$, where B is the minimum size of a feedback link set. We also present coding networks with cycles that require $\Omega(hB)$ encoding nodes.

Finally, we observe that determining, or even crudely approximating the minimum number of required encoding nodes is an $\mathcal{NP}$-hard problem.

B. Encoding Links

A more accurate estimation of the total amount of computation performed by a coding network can be obtained by counting *encoding links*, rather than encoding nodes. A link (v,u), $v \notin s$, is referred to as an encoding link if each packet sent on this link is a combination of two or more packets received on the incoming links of v. Indeed, as the output degrees of nodes in G may vary, encoding nodes might have different computation loads. In addition, only some of the outgoing links of a node may be encoding, while others may only forward incoming packets. Accordingly, we can consider the problem of finding a feasible network code that minimizes the total number of encoding links. It turns out that all upper and lower bounds on the minimum number of encoding nodes presented in this paper, as well as the inapproximability results, carry over to the problem of minimizing the number of encoding links. This follows from the fact that our results are derived by analyzing auxiliary networks in which the degree of any internal node is at most 3. In such networks, the number of encoding links is equal to the number of encoding nodes.

C. Related Work

The problem of minimizing the number of encoding nodes in coding networks is partially addressed in the works of Fragouli *et al.* [6], [7] and Tavory *et al.* [8]. The works of Fragouli *et al.* study the special case of transmitting two packets over an acyclic network (i.e., $h = 2$). They show that in this special case the number of encoding nodes is bounded by the number of terminals k. The proof techniques used in [6], [7] rely on a certain combinatorial decomposition of the underlying network and do not generalize to the case in which the number of packets h is larger than two.

The problem of minimizing the number of required encoding nodes is also studied by Tavory *et al.* [8]. They obtain partial results of nature similar to those of [6] and [7], mentioned above. Namely, they prove, for the case of $h = 2$, that the number of required encoding nodes is independent of the size of the underlying graph G. For general values of h, [8] conjectures that the number of encoding nodes required by an acyclic coding network is independent of its size and depends only on the number of packets h and the number of terminals k. In our study we prove this conjecture.

Finally, the issue of encoding versus forwarding nodes in the solution of network coding problems was also studied by Wu *et al.* [9]. They show the existence (and efficient construction) of network codes in which only nodes which are not directly connected to a terminal perform encoding. The results in [9] do not imply bounds on the number of required encoding nodes.

D. Organization

The rest of the paper is organized as follows. In Section II, we present a formal definition of the network coding problem and state our results in detail. In Section III, we define and motivate the notion of a *simple* network. In that section, we also present an algorithm for finding network codes that require a bounded number of encoding nodes. In Section IV, we establish the h^3t^2 upper bound on the number of encoding nodes in acyclic networks. In Section V, we focus on general (cyclic) networks. In Section VI, we present lower bounds for both acyclic and cyclic networks, and show that determining (or even approximating) the minimum number of encoding nodes is $\mathcal{NP}$-hard. Finally, in Section VII, we conclude with a few remarks and open problems.

II. Model

A communication network is modeled by a directed graph $G = (V, E)$ where V is the set of nodes in G and E is the set of links. We assume that each link $e \in E$ can transmit one packet per time unit. In order to model links whose capacity is higher than one unit, G may include multiple parallel links. A multicast network $\mathbb{N}(G, s, T, h)$ is a 4-tuple that includes a graph $G(V, E)$, a source node $s \in V$, a set of terminals $T \subseteq V$, and the number of packets h that must be transmitted from the source node s to every terminal $t \in T$. We assume that each packet is a symbol of some alphabet Σ.

Definition 1 (Network Code $\mathbb{F}(\mathbb{N})$): A network code for a multicast network $\mathbb{N}(G, s, T, h)$ is defined by encoding functions $\mathbb{F}(\mathbb{N}) = \{f_e \mid e \in E\}$. For links $e = (s, u)$ leaving the source, $f_e : \Sigma^h \to \Sigma$. For other links $e = (v, u)$, $f_e : \Sigma^{d_{\text{in}}(v)} \to \Sigma$. Here, $d_{\text{in}}(v)$ is the in-degree of node v.

The function $f_{(v,u)}$ specifies the packet transmitted on link (v, u) for any possible combination of packets transmitted on the incoming links of v. For links leaving the source s, f_e takes as input the h packets available at s.

Definition 2 (Encoding and Forwarding Links and Nodes): Let $\mathbb{F}(\mathbb{N})$ be a network code, $e \in E$ a network link, and f_e the encoding function of e in $\mathbb{F}(\mathbb{N})$. Link e is referred to

as an *encoding link* if f_e depends on two variables or more. Otherwise, f_e is referred to as a *forwarding link*. We say that a node $v, v \neq s$, is an *encoding node* if at least one of its outgoing links (v,u) is encoding. If all outgoing links of a node v are forwarding, the node is referred to as a *forwarding node*.

In general, there may be links $e(v,u) \in E$ whose encoding functions f_e do not depend on the incoming packets of v (i.e., f_e is constant). Such links transmit no information and can be removed from the network. In addition, there may be links e for which the function f_e depends on a single variable, but $f_e(x) \neq x$. We refer to such links as forwarding nevertheless, and do not count them as encoding links. It is easy to verify that if $\mathbb{F}(\mathbb{N})$ includes links with corresponding functions f_e that depend on a single variable but are not the identity function, one can construct a new network code $\mathbb{F}'(\mathbb{N})$ without such functions such that the number of encoding nodes in $\mathbb{F}'(\mathbb{N})$ and $\mathbb{F}(\mathbb{N})$ are equal.

A network code $\{f_{(v,u)} \mid (v,u) \in E\}$ for a multicast network $\mathbb{N}(G,s,T,h)$ is said to be feasible if it allows communication at rate h between s and each terminal $t \in T$. We say that a network code $\mathbb{F}(\mathbb{N})$ for an acyclic multicast network $\mathbb{N}(G,s,T,h)$ allows communication at rate h if each terminal $t \in T$ can compute the original h packets available at the source from the packets received via its incoming links. To define the notion of rate for networks with cycles, we consider multiple rounds of transmission, at each round the source s sends h packets over the network. We say that the network code allows transmission at rate h if each terminal $t \in T$ can reconstruct the packets sent by the source, such that each packet is reconstructed after a fixed number of rounds.

We are ready now to define the multicast network coding problem.

Definition 3 (Multicast Network Coding Problem): Given a multicast network $\mathbb{N}(G,s,T,h)$, find a feasible network code $\mathbb{F}(\mathbb{N})$ that allows communication a rate h between source s and terminals T.

We say that $\mathbb{N}(G,s,T,h)$ is a feasible muticast network if there exists a feasible network code for $\mathbb{N}$. A multicast network $\mathbb{N}(G,s,T,h)$ is feasible if and only if each cut that separates the source node s and a terminal $t_i \in T$ contains at least h links [2].

A. Statement of Results

As mentioned in the Introduction, our goal is to find feasible network codes that require a minimum number of encoding nodes. For a multicast network $\mathbb{N}(G,s,T,h)$, we denote by $\mathrm{Opt}(\mathbb{N})$ the minimum number of encoding nodes in any feasible network code for $\mathbb{N}(G,s,T,h)$.

We show that computing $\mathrm{Opt}(\mathbb{N})$ is an $\mathcal{NP}$-hard problem. Furthermore, it is $\mathcal{NP}$-hard to approximate $\mathrm{Opt}(\mathbb{N})$ within any multiplicative factor or within an additive[2] factor significantly less than $|V|$. This result follows from the fact that it is $\mathcal{NP}$-hard to distinguish between networks $\mathbb{N}$ in which $\mathrm{Opt}(\mathbb{N}) = 0$ and networks in which $\mathrm{Opt}(\mathbb{N}) > 0$.

[2]An estimate to $\mathrm{Opt}(\mathbb{N})$ which is within an α-multiplicative factor of $\mathrm{Opt}(\mathbb{N})$ is referred to as an α-multiplicative approximation of $\mathrm{Opt}(\mathbb{N})$. An estimate to $\mathrm{Opt}(\mathbb{N})$ which is within an α-additive value of $\mathrm{Opt}(\mathbb{N})$ is referred to as an α-additive approximation of $\mathrm{Opt}(\mathbb{N})$.

Theorem 4: Let $\varepsilon > 0$ be any constant. Let $\mathbb{N}(G,s,T,h)$ be a multicast network in which the underlying graph has $|V|$ nodes. Approximating the value of $\mathrm{Opt}(\mathbb{N})$ within any multiplicative factor or within an additive factor of $|V|^{1-\varepsilon}$ is $\mathcal{NP}$-hard.

Although the problem of finding the exact or approximate value of $\mathrm{Opt}(\mathbb{N})$ is $\mathcal{NP}$-hard, we establish upper bounds on $\mathrm{Opt}(\mathbb{N})$ that hold for any multicast network $\mathbb{N}(G,s,T,h)$. The main contribution of this paper is an upper bound on $\mathrm{Opt}(\mathbb{N})$ for acyclic networks which is independent of the size of the network and depends only on h and $k = |T|$. Specifically, we show that $\mathrm{Opt}(\mathbb{N}) \leq h^3k^2$ for any acyclic multicast network $\mathbb{N}$ that delivers h packets to k terminals. Our bound is constructive, i.e., for any feasible network $\mathbb{N}(G,s,T,h)$ we present an efficient algorithm that constructs a network code with at most h^3k^2 encoding nodes. In what follows, an algorithm is said to be efficient if its running time is polynomial in the size of the underlying graph G.

Theorem 5 (Upper Bound, Acyclic Networks): Let G be an acyclic graph and let $\mathbb{N}(G,s,T,h)$ be a feasible multicast network. Then, one can efficiently find a feasible network code for $\mathbb{N}$ with at most h^3k^2 encoding nodes, i.e., $\mathrm{Opt}(\mathbb{N}) \leq h^3k^2$, where $k = |T|$.

Theorem 6 (Lower Bound, Acyclic Networks): Let r_1 and r_2 be arbitrary integers. Then, there exist multicast networks $\mathbb{N}(G,s,T,h)$ such that $h \geq r_1$, $|T| = k \geq r_2$, the underlying graph G is acyclic, and $\mathrm{Opt}(\mathbb{N}) \geq \Omega(h^2k)$.

Finally, we establish upper and lower bounds on the number of encoding nodes in the general setting of communication networks with cycles. We show that the value of $\mathrm{Opt}(\mathbb{N})$ in a cyclic network $\mathbb{N}$ depends on the size of the minimum feedback link set.

Definition 7 (Minimum Feedback Link Set [10]): Let $G(V,E)$ be a directed graph. A subset $\hat{E} \subseteq E$ is referred to as a feedback link set if the graph G' formed from G by removing all links in $\hat{E}$ is acyclic. A feedback link set of minimum size is referred to as the minimum feedback link set. Given a network $\mathbb{N}(G,s,T,h)$, we denote by B the minimum size of a feedback link set of its underlying graph G.

Theorem 8 (Upper Bound, Cyclic Networks): Let $\mathbb{N}(G,s,T,h)$ be a feasible multicast network. Then, one can efficiently find a feasible network code for $\mathbb{N}$ with at most $(2B+1)h^3k^2$ encoding nodes, i.e., $\mathrm{Opt}(\mathbb{N}) \leq (2B+1)h^3k^2$, where $k = |T|$.

Theorem 9 (Lower Bound, Cyclic Networks): Let r_1 and r_2 be arbitrary integers. Then a) there exist multicast networks $\mathbb{N}(G,s,T,h)$ such that $B \geq r_1$, $h \geq r_2$, and $\mathrm{Opt}(\mathbb{N}) \geq \Omega(Bh)$; b) there exist multicast networks $\mathbb{N}(G,s,T,h)$ such that $|V| \geq r_1$, $k = |T| = 2$, and $\mathrm{Opt}(\mathbb{N}) = \frac{|V|-5}{2}$ (here V is the set of nodes in G).

A couple of remarks are in place. First, note that Theorem 8 generalizes Theorem 5, as for acyclic networks the minimum feedback link set is of size 0. Second, note that Theorem 9 establishes two lower bounds. The first complements the upper bound of Theorem 8, while the second shows that in the case of

cyclic networks the value of $\mathrm{Opt}(\mathbb{N})$ is not necessarily independent of the size of the network and may depend linearly on the number of nodes in G.

III. "Simple" Multicast Networks

Let $\mathbb{N}(G, s, T, h)$ be a feasible multicast network. In order to establish a constructive upper bound on the minimal number of encoding nodes $\mathrm{Opt}(\mathbb{N})$ of $\mathbb{N}$ we consider a special family of feasible networks, referred to as *simple networks*. In what follows, we define simple multicast networks, and show that finding network codes with a bounded number of encoding nodes for this family suffices to prove Theorems 5 and 8. We start by defining feasible multicast networks which are minimal with respect to link removal.

Definition 10 (Minimal Multicast Network): A feasible multicast network $\mathbb{N}(G, s, T, h)$ is said to be minimal with respect to link removal if any network $\hat{\mathbb{N}}(\hat{G}, s, T, h)$ formed from $\mathbb{N}(G, s, T, h)$ by deleting a link e from G is no longer feasible.

Definition 11 (Simple Multicast Network): A multicast network $\mathbb{N}(G, s, T, h)$ is said to be simple if and only if a) $\mathbb{N}$ is feasible; b) $\mathbb{N}$ is minimal with respect to link removal; c) the total degree of each node in G is at most 3 (excluding the source and terminal nodes); and d) the terminal nodes T have no outgoing links.

We now present our reduction between general and simple networks.

A. Reduction to Simple Networks

Let $\mathbb{N}(G, s, T, h)$ be a feasible multicast network. We construct a simple multicast network $\hat{\mathbb{N}}(\hat{G}, s, \hat{T}, h)$ such that any feasible network code for $\hat{\mathbb{N}}$ yields a network code for $\mathbb{N}$ that requires the same or a smaller number of encoding nodes. Our construction is computationally efficient and includes the three following steps.

Step 1—Replacing Terminals: For each terminal $t_i \in T$ we add a new node $\hat{t}_i$ to G and connect t_i to $\hat{t}_i$ by h parallel links. We denote the new set $\{\hat{t}_1, \ldots, \hat{t}_k\}$ of terminals by $\hat{T}$, the resulting graph by G_1, and the resulting multicast network by $\mathbb{N}_1(G_1, s, \hat{T}, h)$.

Step 2—Reducing Degrees: Let G_2 be the graph formed from G_1 by replacing each node $v \in G_1$, $v \neq s$, $v \notin \hat{T}$ whose degree is more than 3 by a subgraph Γ_v, constructed as follows. Let

$$\{(x_i, v) \mid i = 1, \ldots, d_{\mathrm{in}}(v)\} \text{ and } \{(v, y_i) \mid i = 1, \ldots, d_{\mathrm{out}}(v)\}$$

be the incoming and outgoing links of v, respectively, where $d_{\mathrm{in}}(v)$ and $d_{\mathrm{out}}(v)$ are the in- and out-degrees of v. For each incoming link (x_i, v) of v, we add to Γ_v a node $\hat{x}_i$ and a binary tree X_i with root at $\hat{x}_i$ and $d_{\mathrm{out}}(v)$ leaves $\hat{x}_i^1, \ldots, \hat{x}_i^{d_{\mathrm{out}}(v)}$. Similarly, for each outgoing link (v, y_i) of v, we add to Γ_v a node $\hat{y}_i$ and an inverted binary tree Y_i with root at $\hat{y}_i$ and $d_{\mathrm{in}}(v)$ leaves $\hat{y}_i^1, \ldots, \hat{y}_i^{d_{\mathrm{in}}(v)}$. Next, for each $1 \leq i \leq d_{\mathrm{in}}(v)$ and $1 \leq j \leq d_{\mathrm{out}}(v)$ we add a link $(\hat{x}_i^j, \hat{y}_j^i)$ to Γ_v. Finally, we connect Γ_v to the rest of the network by adding links $(x_i, \hat{x}_i)$ for $1 \leq i \leq d_{\mathrm{in}}(v)$ and $(y_i, \hat{y}_i)$ for $1 \leq i \leq d_{\mathrm{out}}(v)$. Figs. 1 and 2 demonstrate the construction of the subgraph Γ_v for a node v with $d_{\mathrm{in}}(v) = d_{\mathrm{out}}(v) = 3$. Note that for any two links (x_i, v) and (v, y_j) there is a path in Γ_v that connects $\hat{x}_i$ and $\hat{y}_i$. The resulting multicast network is denoted by $\mathbb{N}_2(G_2, s, \hat{T}, h)$.

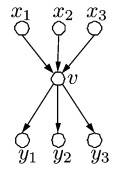

Fig. 1. A node $v \in G$.

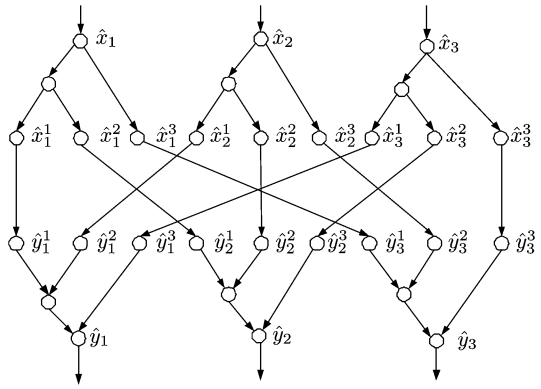

Fig. 2. The gadget Γ_v for v in Fig. 1.

Step 3—Removing Links: Let $\hat{G}$ be *any* subgraph of G_2 such that $\hat{\mathbb{N}}(\hat{G}, s, \hat{T}, h)$ is minimal with respect to link removal. The graph $\hat{G}$ can be efficiently computed by employing the following greedy approach. For each link $e \in G_2$, in an arbitrary order, we check whether removal of e from G_2 would result in a violation of the min-cut condition. The min-cut condition can be easily checked by finding h link-disjoint paths between s and each terminal $\hat{t}_i \in \hat{T}$ (via max-flow techniques, e.g., [11]). All links whose removal does not result in a violation of the min-cut condition are removed from G_2. The resulting network, denoted by $\hat{\mathbb{N}}(\hat{G}, s, \hat{T}, h)$, is the final outcome of our reduction.

We proceed to analyze the properties of $\hat{\mathbb{N}}$. First, we show that $\hat{\mathbb{N}}$ is a simple network. Note that each step of our construction maintains the min-cut condition. Hence, the size of the minimum cut in $\hat{\mathbb{N}}$ between s and any terminal $\hat{t}_i \in \hat{T}$ is at least h, which, in turn, implies, that $\hat{\mathbb{N}}$ is a feasible multicast network. Due to Step 1, the terminals $\hat{T}$ have no outgoing links. Finally, Steps 2 and 3 ensure that the total degree of any node in $\hat{G}$ excluding the source and terminals is at most 3 and that $\hat{\mathbb{N}}$ is minimal with respect to link removal.

Second, we observe that the minimum size of a feedback link set in $\hat{G}$ is smaller or equal to that of G. Indeed, it is easy to verify that if $\mathbb{B}$ is a feedback link set of G, then the links of $\hat{G}$ that correspond to $\mathbb{B}$ form a feedback link set.

Finally, we show that $\mathrm{Opt}(\mathbb{N}) \leq \mathrm{Opt}(\hat{\mathbb{N}})$. That is, any feasible network code for $\hat{\mathbb{N}}$ with ℓ encoding nodes can be used to

efficiently construct a network code for $\mathbb{N}$ that requires the same number of encoding nodes.

Reconstruction of a Feasible Network Code for $\mathbb{N}$*:* Let $\hat{\mathbb{F}}(\hat{\mathbb{N}})$ be a feasible network code for $\hat{\mathbb{N}}(\hat{G}, s, \hat{T}, h)$ with ℓ encoding nodes. A feasible network code $\mathbb{F}(\mathbb{N})$ for $\mathbb{N}(G, s, T, h)$ is constructed as follows. Let $e = (v, u)$ be a link in G. Let e' be the corresponding link between Γ_v and Γ_u in $\mathbb{N}_2$ (recall that $\mathbb{N}_2$ was constructed at Step 2 above). If e' does not appear in $\hat{\mathbb{N}}$ then no information is sent over the link e in $\mathbb{F}(\mathbb{N})$. If e' appears in $\hat{\mathbb{N}}$, the code f_e for $e = (v, u)$ is determined by the codes $f_{\hat{e}} \in \hat{\mathbb{F}}(\hat{\mathbb{N}})$ of links $\hat{e}$ that belong to Γ_v. Specifically, let $X = \{(x_1, \hat{x}_1), \ldots, (x_{d_{\text{in}}(v)}, \hat{x}_{d_{\text{in}}(v)})\}$ be the incoming links of Γ_v where $d_{\text{in}}(v)$ is the in-degree of v in G. The construction of $\mathbb{N}_2$ implies that the packet transmitted on the link e' is a function $f_{e'}$ of the packets transmitted on links X. We use this function as the encoding function f_e for link e in code $\mathbb{F}(\mathbb{N})$. The fact that the incoming links of v in G correspond to the links in X implies the feasibility of the resulting code $\mathbb{F}(\mathbb{N})$.

We note that a node v is an encoding node in $\mathbb{F}(\mathbb{N})$ if and only if at least one of the nodes in Γ_v performs encoding. On the other hand, each encoding node in $\hat{\mathbb{F}}(\hat{\mathbb{N}})$ corresponds to at most one encoding node in $\mathbb{F}(\mathbb{N})$. This implies that the number of encoding nodes in $\mathbb{F}(\mathbb{N})$ is at most ℓ and, in turn, $\text{Opt}(\mathbb{N}) \leq \text{Opt}(\hat{\mathbb{N}})$.

We summarize the above discussion by the following lemma:

Lemma 12: Let $\mathbb{N}(G, s, T, h)$ be a feasible multicast network. Then, one can efficiently construct a simple network $\hat{\mathbb{N}}(\hat{G}, \hat{s}, \hat{T}, h)$ for which a) $|\hat{T}| = |T|$, b) the size of the feedback link set of $\hat{\mathbb{N}}$ is less than or equal to that of $\mathbb{N}$, and c) any feasible network code for $\hat{\mathbb{N}}$ with ℓ encoding nodes can be used to efficiently construct a feasible network code for $\mathbb{N}$ with at most ℓ encoding nodes.

B. The Value of $\text{Opt}(\mathbb{N})$ in Simple Networks

In what follows, we show that for simple networks $\mathbb{N}(G, s, T, h)$ the value of $\text{Opt}(\mathbb{N})$ is equal to the number of nodes in G (excluding the terminals) of in-degree 2.

Definition 13: Let $\mathbb{N}(G, s, T, h)$ be a simple multicast network. We define $\alpha(\mathbb{N})$ to be the number of nodes in $G \setminus T$ of in-degree 2.

Lemma 14: Let $\mathbb{N}(G, s, T, h)$ be a simple multicast network. Let $\mathbb{F}(\mathbb{N})$ be any feasible network code for $\mathbb{N}$. Then, a node $v \in G$, $v \neq s$, $v \notin T$, is an encoding node in $\mathbb{F}(\mathbb{N})$ if and only if the in-degree of v is 2. Thus, $\text{Opt}(\mathbb{N}) = \alpha(\mathbb{N})$.

Proof: Let v be an encoding node in $\mathbb{F}(\mathbb{N})$. Then, the in-degree of v must be larger than one, otherwise all coding functions f_e of the outgoing links of v depend on a single variable, which contradicts the fact that v is an encoding node. Furthermore, since the total degree of v is at most three, and since it has at least one outgoing link, it follows that the in-degree of v is 2.

Now let v, $v \neq s$, $v \notin T$, be a node with in-degree 2, and let $e(v, u)$ be an outgoing link of v (node v must have an outgoing link, otherwise $\mathbb{N}$ is not minimal). If e is not an encoding link, then f_e is a function of a single variable, i.e., f_e depends on packets that arrive on one of the incoming links of v. This implies that the other incoming link can be omitted from G (while preserving the feasibility of $\mathbb{N}(G, s, T, h)$), which contradicts the minimality of $\mathbb{N}(G, s, T, h)$. □

Algorithm ($\mathbb{N}$)*:*

Input:

$\mathbb{N}$ - a multicast network,

1 Transform $\mathbb{N}$ into a simple network $\hat{\mathbb{N}}$ as described in Section III-A.
2 Find **any** feasible network code for $\hat{\mathbb{N}}$, e.g., by using algorithms appearing in [4], [5].
3 Reconstruct the corresponding network code for $\mathbb{N}$ as described in Section III-A.

Fig. 3. Algorithm for finding a network code with a bounded number of encoding nodes.

C. The Algorithm

Lemma 14 implies that for any given simple network $\mathbb{N}(G, s, T, h)$, any upper bound on $\alpha(\mathbb{N})$ is also an (efficiently constructible) upper bound on $\text{Opt}(\mathbb{N})$. Accordingly, in Sections IV and V, we prove that $\alpha(\mathbb{N})$ is bounded by h^3k^2 for acyclic networks and $h^3k^2(2B+1)$ for cyclic networks, where B is the size of the minimum feedback link set of G and $k = |T|$.

This leads to the following efficient procedure for finding a feasible network code with a bounded number of encoding nodes (implied by Lemma 12). The procedure works for a general (not necessarily simple) multicast network $\mathbb{N}(G, s, T, h)$. We begin by transforming $\mathbb{N}$ into a simple network $\hat{\mathbb{N}}$. Then, we find a feasible network code for $\hat{\mathbb{N}}$. Finally, we reconstruct the corresponding network code for the original network $\mathbb{N}$. The description of our procedure appears in Fig. 3.

IV. Upper Bound for Acyclic Networks

In this section, we present the proof of Theorem 5. The proof includes two steps. First, in Section IV-A, we analyze the special case in which the multicast network has only two terminals (i.e., $k = 2$). Then, in Section IV-B, we address the general case in which the number of terminals is arbitrary. In both cases, we establish an upper bound on $\alpha(\mathbb{N})$ for simple acyclic networks $\mathbb{N}$. By Lemmas 12 and 14, the upper bound on $\alpha(\mathbb{N})$ suffices to prove Theorem 5.

A. Networks With Two Terminals

Throughout this section we will use the following theorem implied by the results of [1] and [2] combined with Menger's theorem [12].

Theorem 15 ([12], [1], [2]): Let $\mathbb{N}(G, s, T, h)$ be a multicast network. Then, there exists a feasible network code $\mathbb{F}(\mathbb{N})$ if and only if for each terminal $t_i \in T$ there exist h link-disjoint paths that connect s and t_i.

We begin by introducing the concept of *residual graphs*. The residual graphs capture the minimality of the network at hand with respect to link removal. In particular, the residual graphs that correspond to minimal networks contain no cycles.

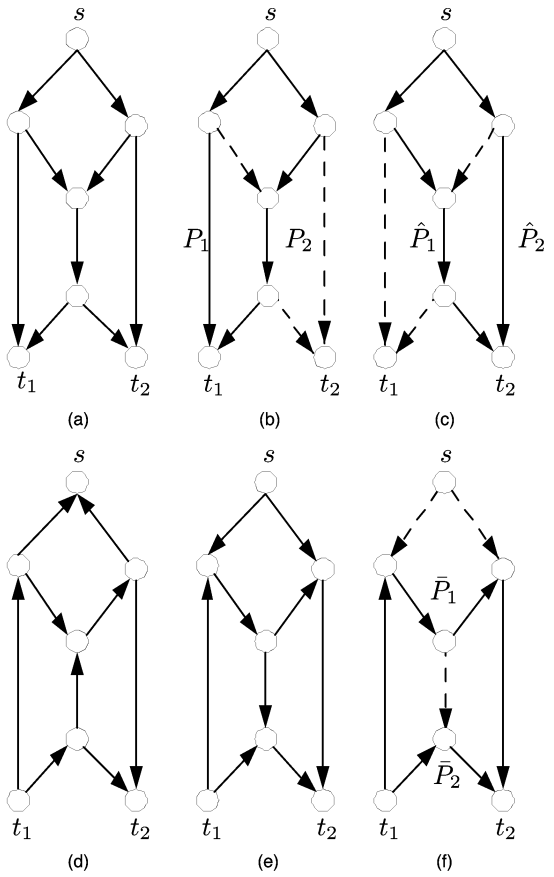

Fig. 4. (a) A simple multicast network $\mathbb{N}(G, s, T, h)$. (b) Red paths P_1 and P_2. (c) Blue paths $\hat{P}_1$ and $\hat{P}_2$. (d) Residual graph G_1. (e) Residual Graph G_2. (f) Auxiliary graph $\hat{G}$ and two green paths $\bar{P}_1$ and $\bar{P}_2$.

Let $\mathbb{N}(G, s, T, h)$ be a simple (and therefore feasible) multicast network with two terminals t_1 and t_2. By Theorem 15, there exist h link-disjoint paths from s to t_1 and h link-disjoint paths from s to t_2. Throughout this subsection, we fix a set of h link disjoint paths from s to t_1 and refer to them as *red* paths. Similarly, we fix a set of h link-disjoint paths from s to t_2 and refer to them as *blue* paths. We refer to links that belong to red paths as red links, and links that belong to blue paths as blue links. As a link in G can belong to a red path and to a blue path, it can be both red and blue.

We observe that a simple network $\mathbb{N}(G, s, T, h)$ has the following properties. First, every link in G belongs to either a blue or a red path. Second, at most one red (blue) path may pass through any node v in G which is not a terminal or the source. Third, all links entering t_1 are exclusively red, and all links entering t_2 are exclusively blue. Finally, the source node s has no incoming links. The first and final properties follow from the minimality of $\mathbb{N}(G, s, T, h)$. The second property follows from the fact that the degree of each node which is not a terminal or a source is at most 3. The third property follows from the fact that the terminals have no outgoing links and the fact that the red (blue) paths must terminate at t_1 (t_2).

A simple multicast network $\mathbb{N}(G, s, T, h)$ with corresponding red and blue paths is depicted in Fig. 4(a)–(c). We are ready now to define residual graphs G_1 and G_2.

Definition 16 (Residual Graph G_1): Let $\mathbb{N}(G, s, T, h)$ be a simple multicast network with two terminals (i.e., $k = 2$). The residual graph G_1 is formed from G by reversing all links that belong to red paths (including links that belong both to red and blue paths).

Definition 17 (Residual Graph G_2): Let $\mathbb{N}(G, s, T, h)$ be a simple multicast network with two terminals. The residual graph G_2 is formed from G by reversing all links that belong to red paths only (not including links that belong both to red and blue paths).

Examples of the residual graphs G_1 and G_2 are depicted in Fig. 4(d) and (e), respectively,

Lemma 18: Let $\mathbb{N}(G, s, T, h)$ be a simple multicast network with an acyclic graph G. Then, the residual graphs G_1 and G_2 are acyclic.

Proof: We denote the h red paths between s and t_1 by $P_1, \ldots, P_h$, and the h blue paths between s and t_2 by $\hat{P}_1, \ldots, \hat{P}_h$. Since $\mathbb{N}(G, s, T, h)$ is simple (and thus minimal with respect to link removal), every link in G is either red, blue, or both red and blue. Suppose that the residual graph G_1 contains a cycle C. Each link $e \in C$ has a corresponding link e' in G which is either identical to e or is the reverse of e. In what follows, the color of a link e in C is defined to be the color of the corresponding link e' in G.

First, we note that C does not include terminals t_1 and t_2 (because terminal t_1 has in-degree 0 in G_1 and terminal t_2 has out-degree 0 in G_1). Second, we observe that C contains at least one link e_1 whose color is exclusively blue. Otherwise, the links that belong to C would form a cycle in the original graph G consisting exclusively of red links, which contradicts our assumption that G is acyclic.

Third, we prove that C contains a link e_2 whose color is exclusively red. Otherwise, consider the case in which all links that belong to C are either exclusively blue or red and blue. At least one of them, e.g., e_1, is exclusively blue (which implies that the direction of e_1 in G is the same as in C). We observe that C must include links that are not exclusively blue, otherwise, there would be a cycle in G. Thus, there are links e in C which are both red and blue (which implies that they appear in opposite directions in G and C). Hence, there exist two links (u, v) and (v, w) in C, such that (u, v) is exclusively blue and (v, w) is both red and blue. This implies that in the original graph G node v has two input links, both of them belong to blue paths. Since v is not a terminal node, it must have two output links, one for each blue path. Thus, the degree of this node in G is at least 4, in contradiction to our assumption that the degree of each internal node in G is at most 3.

We have shown that the cycle C of G_1 includes a link e_1 which is exclusively blue and a link e_2 which is exclusively red. Let $e_2' \in G$ be the reverse link of $e_2 \in C$. We now show that e_2' can be removed from the original graph G, which contradicts the minimality of the network $\mathbb{N}(G, s, T, h)$. To that end, we show that there exist h link-disjoint paths $P_1', \ldots, P_h'$ from s to t_1 in G which do not include link e_2'.

Specifically, we use the techniques from the theory of network flows [11]. Let E' be a set of links that belong to red paths,

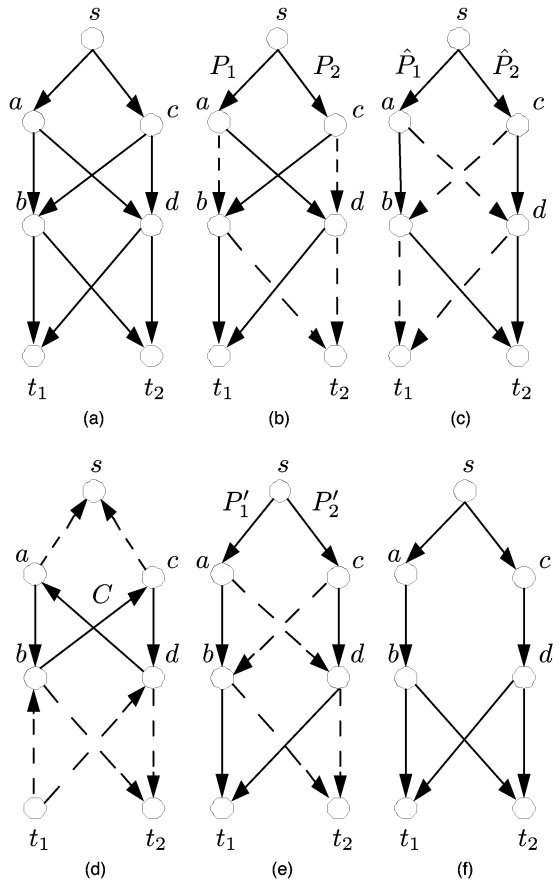

Fig. 5. (a) A non-minimal multicast network $\mathbb{N}(G, s, T, h)$. (b) Red paths P_1 and P_2. (c) Blue paths $\hat{P}_1$ and $\hat{P}_2$. (d) Residual graph G_1 which has a cycle C. (e) The new set of red paths (after augmentation along cycle C). (f) A minimal network formed from $\mathbb{N}(G, s, T, h)$ by removing links (a, d) and (c, b).

i.e., $E' = \{e \mid e \in P_i, i = 1, \ldots, h\}$. The set E' represents a flow between s and t_1 of value h. We observe that G_1 is a residual graph with respect to this flow and C is an *augmenting cycle* in the residual graph G_1. We now augment E' along C and denote the resulting flow by E''. Flow E'' includes the following links: a) each link $e = (v, u)$ in E' whose reverse link (u, v) does not belong to C, b) each link $e = (v, u)$ that belongs to C and whose reverse link (u, v) does not belong to E', i.e.,

$$E'' = \{(v,u) \mid (v,u) \in E' \text{ and } (u,v) \notin C\} \cup \{(v,u) \mid (v,u) \in C \text{ and } (u,v) \notin E'\}. \quad (1)$$

Note that E'' does not include link e'_2. Also, it is easy to verify that any cut that separates s and t_1 in G has at least h links that belong to E''. Indeed, for any cut (V_1, V_2), the number of links of C that cross (V_1, V_2) in the forward direction is identical to the number of links of C that cross (V_1, V_2) in the reverse direction. This implies [12] that E'' can be decomposed to h link-disjoint paths $P'_1, \ldots, P'_h$ between s and t_1 in G. We note that these paths do not include link e'_2. This implies that e'_2 can be removed from the G, which contradicts the minimality of $\mathbb{N}(G, s, T, h)$.

Our proof is illustrated in Fig. 5. Fig. 5(a)–(c) depicts a multicast network $\mathbb{N}(G, s, T, h)$ with two terminals t_1 and t_2, and the initial sets of red and blue paths, $\{P_1, P_2\}$ and $\{\hat{P}_1, \hat{P}_2\}$. The corresponding residual graph G_1 is depicted in Fig. 5(d). Note that G_1 has a cycle $\{a, b, c, d\}$ that includes two links (b, c) and (d, a) which are exclusively red. Fig. 5(e) depicts a new set of red paths $\{P'_1, P'_2\}$ in G obtained by augmenting the flow formed by red paths $\{P_1, P_2\}$ along cycle C. Since links (c, b) and (a, d) no longer belong to red paths, they can be removed from the graph. A minimal network formed from $\mathbb{N}(G, s, T, h)$ by removing links (a, d) and (c, b) is depicted on Fig. 5(f).

Using a similar argument, we can show that the same property holds for the residual graph G_2. Specifically, we consider the graph $\hat{G}_2$ formed from G_2 by reversing all its links, and replacing "red" with "blue" in the arguments above. □

Up to this point, after fixing the red and blue paths of G, we have defined two residual graphs G_1 and G_2. We now define an additional (and final) graph $\hat{G}$ and a set of *green* paths.

Definition 19 (Auxiliary Graph $\hat{G}$): Let $\mathbb{N}(G, s, T, h)$ be a simple multicast network with two terminals (i.e., $k = 2$). Let $\{P_1, \ldots, P_h\}$ be a set of red paths and let $\{\hat{P}_1, \ldots, \hat{P}_h\}$ be a set of blue paths. Let $\hat{G}$ be the graph formed from G by a) deleting links that belong to both a red path and a blue path in $\mathbb{N}$; and b) reversing links that belong to a red path and do not belong to a blue path.

As in the case of G_1 and G_2, the definition of $\hat{G}$ depends on the sets of red and blue path chosen above.

In the following lemma we prove that $\hat{G}$ consists of h link-disjoint paths between t_1 and t_2. We refer to these paths as *green* paths. An example of an auxiliary graph $\hat{G}$ with two green paths is depicted in Fig. 4(f).

Lemma 20: Let $\mathbb{N}(G, s, T, h)$ be a simple multicast network. Then, the corresponding graph $\hat{G}$ consists of h link-disjoint paths between t_1 and t_2 ($\hat{G}$ may also include isolated vertices of total degree 0).

Proof: First we prove that each node $v \in \hat{G}$, excluding the terminals t_1, t_2 and the source node s, has exactly one incoming link and exactly one outgoing link. For each such node v in the original graph G, one of the following holds.

Case 1: Node v has one incoming link e and one outgoing link e'. In this case, either a blue path or a red path or both red and blue paths (together) pass though v. In the first two cases, both links appear in $\hat{G}$, either in the original direction or in the reverse direction, hence the node v has one incoming link and one outgoing link in the auxiliary graph $\hat{G}$. In the last case, both links e and e' do not appear in $\hat{G}$.

Case 2: Node v has two incoming links e_1, e_2 and one outgoing link e'. In this case, a red and a blue path enter node v through separate links and exit v (together) through e'. Thus, e' does not appear in $\hat{G}$ and one of the incoming links (e_1 or e_2) is reversed. Thus, the node v has one incoming link and one outgoing link in the auxiliary graph $\hat{G}$.

Case 3: Node v has one incoming link e and two outgoing links e'_1 and e'_2. In this case, a red and a blue path enter node v through link e and exit v through separate links e'_1 and e'_2. Thus, e does not appear in $\hat{G}$ and one of the outgoing links (e_1 or e_2) is reversed. Thus, the node v has one incoming link and one outgoing link in the auxiliary graph $\hat{G}$.

Next, we show that the in-degree and the out-degree of the source node s in the auxiliary graph $\hat{G}$ are equal. Since $\mathbb{N}(G,s,T,h)$ is minimal, s has no incoming links and each outgoing link of s belongs to a blue path or a red path. Let h' be the number of outgoing links of s in G that belong to red paths and do not belong to blue paths. Note that h' is the in-degree of s in $\hat{G}$. Since the number of blue paths is equal to the number of red paths, the number of outgoing links of s in G that belong to blue paths and do not belong to red paths is also h'. Thus, the out-degree of s in $\hat{G}$ is also h'.

Finally, we observe that in the original graph G, the in-degree of each terminal t_1, t_2 is exactly h. This follows from the fact that any link entering t_1 (t_2) is exclusively red (blue).

In summary, the auxiliary graph $\hat{G}$ has the following properties. First, the out-degree of t_1 and in-degree of t_2 are equal to h. Second, the in-degree of s is equal to its out-degree. Third, each other node $v \in \hat{G}$ has in-degree 1 and out-degree 1. Finally, $\hat{G}$ does not contain a cycle, otherwise, there would be a cycle in G_1 (note that $\hat{G} \subseteq G_1$), which contradicts the minimality of $\mathbb{N}(G,s,T,h)$. This implies that there exist h link-disjoint paths in $\hat{G}$ that connect t_1 and t_2. These paths are referred to as *green* paths. It remains to show that each link in $\hat{G}$ belongs to a green path. Suppose, by way of contradiction, that there exists a link $e = (v,u) \in \hat{G}$ that does not belong to a green path. By the preceding analysis, one can extend this link into a path from v to t_2. This path will consist of links that do not belong to the h green paths described above, implying that t_2 has in-degree larger than h, a contradiction. □

We observe that since $\hat{G}$ is a subgraph of the residual graphs G_1 and G_2, the green paths belong to G_1 and G_2 as well.

The proof of the following lemma follows directly from Lemma 20 and the definitions of G_1 and G_2.

Lemma 21: Let $\mathbb{N}(G,s,T,h)$ be a simple multicast network. For each link $e = (v,u) \in G$ exactly one of the following conditions hold: a) e belongs to a red and to a blue path in G, b) e belongs to a red path in G and the reverse link (u,v) of e belongs to a green path in $\hat{G}$, c) e belongs to a blue path in G and to a green path in $\hat{G}$.

Proof: We consider three cases: 1) Link e is both a red and a blue link. In this case, e satisfies property a). Since e does not appear in $\hat{G}$, properties b) and c) do not hold for e. 2) Link e is an exclusively red link (not blue). Clearly, properties a) and c) stated in the lemma do not hold for e. Moreover, the reverse of e belongs to $\hat{G}$. Hence (by Lemma 20) e belongs to a green path and satisfies property b). 3) Link e is an exclusively blue link (not red). In this case, e does not satisfy properties a) and b). Since e belongs to $\hat{G}$ it belongs to a green path and hence satisfies property c). □

We are ready to establish the upper bound on $\alpha(\mathbb{N})$ for simple multicast networks $\mathbb{N}$ with two terminals. We begin with the following lemma.

Lemma 22: Let $\mathbb{N}(G,s,T,h)$ be a simple multicast network with two terminals. Fix a set $\mathbb{S}_r$ of link-disjoint red paths in G from s to t_1 and a set $\mathbb{S}_b$ of link-disjoint blue paths in G from s to t_2. Let $\hat{G}$ be the auxiliary graph, defined above, and let $\mathbb{S}_g$ be the set of green paths from t_1 to t_2 in $\hat{G}$. Let $P_r \in \mathbb{S}_r$ be a red path, $P_b \in \mathbb{S}_b$ be a blue path, and $P_g \in \mathbb{S}_g$ be a green path. Then, there exists at most one node $v \in G$ that belongs to all three paths P_r, P_b, and P_g.

Proof: Let G_1, G_2 be residual graphs of G. We observe that path P_g belongs to both G_1 and G_2. We also observe that for each red path $P_r \in G$ there exists a path $P'_r \in G_1$ formed by reversing the links of P_r. Suppose, by way of contradiction, that there exist two nodes that belong to P_r, P_b, and P_g. We denote these nodes by v_1 and v_2, such that v_1 is a predecessor of v_2 in P_g. First, we note that v_1 must be a predecessor of node v_2 in path P_b. Indeed, if this does not hold, then there exists a cycle in the residual graph G_2 formed by links that belong to paths P_b and P_g, which, by Lemma 18, contradicts the minimality of $\mathbb{N}(G,s,T,h)$. Second, the node v_2 must be a predecessor of node v_1 in P_r, otherwise, there would be a cycle in the residual graph G_1 that includes links that belong to paths P_g and P'_r, again contradicting the minimality of $\mathbb{N}(G,s,T,h)$. We conclude that node v_1 is a predecessor of node v_2 in P_b, while v_2 is a predecessor of node v_1 in P_r. This, however, implies that there exists a cycle in the original graph G, which contradicts the assumption that G is acyclic. □

We summarize our results by the following theorem.

Theorem 23: Let $\mathbb{N}(G,s,T,h)$ be a simple acyclic multicast network with two terminals. Then, $\alpha(\mathbb{N}) \leq h^3$.

Proof: Fix a set $\mathbb{S}_r$ of link-disjoint red paths in G between s and t_1 and a set $\mathbb{S}_b$ of link-disjoint blue paths in G between s and t_2. Let G_1, G_2, $\hat{G}$ be auxiliary graphs and let $\mathbb{S}_g$ be the set of green paths between t_1 and t_2 in $\hat{G}$. We denote by V' the set of nodes of G that are not terminals and whose in-degree is 2. Note that $|V'| = \alpha(\mathbb{N})$. Note also that each node $v \in V'$ has two incoming links: one belongs to a blue path (only) and one to a red path (only); and a single outgoing link that belongs to a red and to a blue path (the above follows directly from the fact that $\mathbb{N}$ is a simple network). Thus, by Lemma 21, each such node belongs to a red path $P_r \in \mathbb{S}_r$, a blue path $P_b \in \mathbb{S}_b$, and a green path $P_g \in \mathbb{S}_g$. Lemma 22 implies that at most one node belongs to the same three paths of different colors. Since there are exactly h red paths, h blue paths, and h green paths, this implies that the number of nodes in V' is at most h^3. □

B. Theorem 5: Networks With $k > 2$

In this subsection, we establish an upper bound on the size of $\alpha(\mathbb{N})$ for simple acyclic multicast networks with an arbitrary number of terminals. This suffices to prove Theorem 5, stated in the Introduction.

Theorem 24: Let $\mathbb{N}(G,s,T,h)$ be a simple acyclic multicast network. Then, $\alpha(\mathbb{N}) \leq h^3k^2$, where $k = |T|$.

Proof: We begin with the following observation. Let $t_i, t_j \in T$ be two terminals. By Theorem 15, there are h link-disjoint paths between s and t_i and h link-disjoint paths between s and t_j. Let $G'_{(i,j)}$ be a subgraph of G induced by links that belong to these disjoint paths. We note that the multicast network $\mathbb{N}'(G'_{(i,j)}, s, \{t_i, t_j\}, h)$ is feasible, but not necessarily minimal. Let $\mathbb{N}_{(i,j)}(G_{(i,j)}, s, \{t_i, t_j\}, h)$ be a minimal multicast network obtained from $G'_{(i,j)}$ by repeatedly removing redundant links. Note that $\mathbb{N}_{(i,j)}$ is a simple network.

We denote by $\mathbb{P}^i_{(i,j)}$ and $\mathbb{P}^j_{(i,j)}$ sets of h link-disjoint paths in $G_{(i,j)}$, from s to t_i and from s to t_j, respectively. Consider the set of nodes $V_{(i,j)}$ in $G_{(i,j)}$ that have in-degree 2, not including the source and terminals. Note that $\alpha(\mathbb{N}_{(i,j)}) = |V_{(i,j)}|$ and, by Theorem 23, $\alpha(\mathbb{N}_{(i,j)}) \leq h^3$.

Assume, by way of contradiction, that $\alpha(\mathbb{N})$ exceeds h^3k^2. Then, there exists at least one node v with in-degree 2, $v \neq s$, $v \notin T$ that does not belong to $\bigcup_{t_i,t_j \in T} V_{(i,j)}$. Let e' and e'' be the incoming links of v and let e be the outgoing link of v.

We show that $\mathbb{N}(G,s,T,h)$ remains feasible after the deletion of either e' or e'', which contradicts the minimality of $\mathbb{N}(G,s,T,h)$. Note that for each terminal t_i, there are several possible sets $\{\mathbb{P}^i_{(i,j)} \mid t_j \in T\}$ of h link-disjoint paths between s and t_i. If for each terminal $t_i \in T$ it holds that there exists $\mathbb{P}^i_{(i,j)}$ that does not include e', then link e' can be omitted from $\mathbb{N}(G,s,T,h)$ without violating its feasibility. Otherwise, there exists a terminal t_i such that link e' is included by all sets $\{\mathbb{P}^i_{(i,j)} \mid t_j \in T\}$. However, in this case it holds that all sets $\{\mathbb{P}^j_{(i,j)} \mid t_j \in T\}$ do not include e'', which contradicts the minimality of $\mathbb{N}$. Indeed, if e'' belongs to some set $\mathbb{P}^j_{(i,j)}$, then v necessarily has degree 2 in $G_{(i,j)}$ and thus belongs to $V_{(i,j)}$. □

V. Upper Bound for General (Cyclic) Networks

In this section, we establish an upper bound of $(2B+1)h^3k^2$ on the number of encoding nodes for multicast networks $\mathbb{N}(G,s,T,h)$ with cycles (Theorem 8), where B is the minimum size of a feedback link set of G. The proof of the upper bound for cyclic networks is very similar to that of Theorem 5. Specifically, we show that in a cyclic network, the value of $\alpha(\mathbb{N})$ is bounded by $(2B+1)h^3k^2$. This fact, coupled with Lemmas 12 and 14, is sufficient to prove the correctness of Theorem 8. As we bound $\alpha(\mathbb{N})$, our bounds on the number of encoding nodes hold for both time-invariant (e.g., [2]) and time-varying network codes.

In what follows we roughly outline the proof of Theorem 8 emphasizing only the changes required to extend the proof of Theorem 5 to the case in which the given multicast network $\mathbb{N}(G,s,T,h)$ includes cycles.

A. Residual Graphs and Link-Disjoint Paths

Let $\mathbb{N}(G,s,T,h)$ be a simple multicast network with two terminals $T = \{t_1,t_2\}$ (i.e., $k = 2$). As in Section IV, $\mathbb{N}(G,s,T,h)$ is feasible, thus by Theorem 15 there exist h link-disjoint paths $P_1,\ldots,P_h$ that connect s and terminal t_1 (referred to as red paths) and h link-disjoint paths $\hat{P}_1,\ldots,\hat{P}_h$ between s and t_2 (referred to as blue paths). We assume, without loss of generality, that paths $P_1,\ldots,P_h$ and $\hat{P}_1,\ldots,\hat{P}_h$ do not include cycles. Indeed, if the size of a minimum cut between s and a terminal t is h, then there exist h link-disjoint paths between s and t that do not include cycles [11]. Thus, Lemmas 18, 20, and 21 hold in the cyclic case as well. The proof of Lemma 18 needs minor and straightforward modifications, while Lemmas 20, and 21 carry over without modification. We now prove an analog to Lemma 22, and state the resulting analogs to Theorems 23 and 24.

Lemma 25: Let $\mathbb{N}(G,s,T,h)$ be a simple multicast network with two terminals and let B be the size of the minimum feedback link set in G. Fix a set $\mathbb{S}_r$ of link-disjoint red paths in G between s and t_1 and a set $\mathbb{S}_b$ of link-disjoint blue paths in G between s and t_2. Let $\hat{G}$ be the auxiliary graph, as defined in Section IV-A, and let $\mathbb{S}_g$ be the set of green paths between t_1 and t_2 in $\hat{G}$. Let $P_r \in \mathbb{S}_r$ be a red path, $P_b \in \mathbb{S}_b$ be a blue path, and $P_g \in \mathbb{S}_g$ be a green path. Then, there exist at most $(2B+1)$ nodes in G that belong to all three paths P_r, P_b, and P_g.

Proof: Let G_1, G_2 be residual graphs of G, as defined in Section IV-A. We observe that path P_g belongs to G_1 and G_2. We also observe that for each red path $P_r \in G$ there exists a path $P'_r \in G_1$ formed by reversing the links of P_r. Suppose, by way of contradiction, that there exist $2B+2$ nodes that belong to P_r, P_b, and P_g. We denote these nodes by $v_1,\ldots,v_{2B+2}$ such that for $i = 1$ to $2B+1$, v_i is a predecessor of v_{i+1} in P_g. First, we note that v_i must be a predecessor of node v_{i+1} in path P_b. Indeed, if this does not hold, then there exists a cycle in the residual graph G_2 formed by links that belong to paths P_b and P_g, which, by Lemma 18, contradicts the minimality of $\mathbb{N}(G,s,T,h)$. Second, v_{i+1} must be a predecessor of v_i in P_r, otherwise, there would be a cycle in the residual graph G_1 that includes links belonging to paths P_g and P'_r, which contradicts the minimality of $\mathbb{N}(G,s,T,h)$.

We conclude that for each i, v_i is a predecessor of v_{i+1} in P_b, while v_{i+1} is a predecessor of v_i in P_r. This implies that there exist at least $2B+1$ cycles in the subgraph G' of G induced by links that belong to P_r and P_b. We now show that the minimum feedback link set of G' is at least $B+1$, which contradicts our assumption that the minimum feedback link set of G is B. Consider a link e in G'. If e is exclusively red or blue, then e appears in only one of the $2B+1$ cycles in G'. Otherwise, if e is both red and blue, it can appear in at most two cycles in G'. Thus, the minimum feedback link set of G' is of size at least $B+1$. □

B. Upper Bound

The following theorems establish the upper bound on the number of encoding nodes in networks with cycles.

Theorem 26: Let $\mathbb{N}(G,s,T,h)$ be a simple multicast network with two terminals. Then, $\alpha(\mathbb{N}) \leq (2B+1)h^3$.

Proof: The proof follows the same lines as that of Theorem 23, using Lemma 25 instead of Lemma 22. □

Theorem 27: Let $\mathbb{N}(G,s,T,h)$ be a simple multicast network. Then, $\alpha(\mathbb{N}) \leq (2B+1)h^3k^2$, where $k = |T|$.

Proof: The proof follows the same lines as that of Theorem 24, using Lemma 25 instead of Lemma 22. □

Let $\mathbb{N}(G,s,T,h)$ be a general (not necessarily simple) multicast network. We observe that the bound on $\mathrm{Opt}(\mathbb{N})$ stated in Theorem 8 may be strengthened by replacing the parameter B with the size of the minimum feedback link set in any simple network $\hat{\mathbb{N}}$ obtained form $\mathbb{N}$ by link removal.

In the following definition, $B(\mathbb{N})$ denotes the minimum size of a feedback link set of the underlying graph G of $\mathbb{N}(G,s,T,h)$.

Definition 28: Let $\mathbb{N}(G,s,T,h)$ be a feasible multicast network. We define $\mathrm{Opt}_{FES}(\mathbb{N})$ to be $\min_{\hat{\mathbb{N}}} B(\hat{\mathbb{N}})$ taken over all feasible subnetworks $\hat{\mathbb{N}}(\hat{G},s,T,h)$ obtained from $\mathbb{N}(G,s,T,h)$ by removing links from the underlying graph G (i.e., $\hat{G}\subseteq G$).

Theorem 29 (Strong Upper Bound for Cyclic Networks): Let $\mathbb{N}(G,s,T,h)$ be a feasible multicast network. Then there exists a feasible network code for $\mathbb{N}$ in which the number of encoding nodes is at most $(2\mathrm{Opt}_{FES}(\mathbb{N})+1)h^3k^2$, where $k=|T|$.

Theorem 29 implies that given a multicast network $\mathbb{N}(G,s,T,h)$ it is desirable to find a simple network $\hat{\mathbb{N}}(\hat{G},s,T,h)$, such that $\hat{G}\subseteq G$ and the size of the minimum feedback link set of $\hat{G}$ is $\mathrm{Opt}_{FES}(\mathbb{N})$. However, in Section VI-B we show that this task is $\mathcal{NP}$-hard. Moreover, we show that even finding a network $\hat{\mathbb{N}}$ with a feedback link set that is *somewhat close* to being of size $\mathrm{Opt}_{FES}(\mathbb{N})$ is $\mathcal{NP}$-hard.

VI. Negative Results

In this section we prove Theorems 4, 6, 9, and the $\mathcal{NP}$-hardness of computing or approximating $\mathrm{Opt}_{FES}(\mathbb{N})$ (as defined in Section V).

A. Theorem 4: Exact and Approximate Computation of $\mathrm{Opt}(\mathbb{N})$

Proof (of Theorem 4): We present a reduction between the link-disjoint path (LDP) problem in directed graphs and the problem of computing the minimum number of encoding nodes required by a multicast network $\mathbb{N}$. Given a directed graph $G=(V,E)$ and two pairs of nodes in V, (s_1,t_1), and (s_2,t_2) the LDP problem is the problem of finding two link-disjoint paths in G connecting s_1 to t_1 and s_2 to t_2. This problem is known to be $\mathcal{NP}$-complete [13]. This reduction will imply the hardness results stated in Theorem 4.[3]

In Fig. 6(a), we show how an instance $\{G',s_1,t_1,s_2,t_2\}$ of problem LDP is reduced to an instance $\mathbb{N}(G,s,T,h)$ of the multicast network coding problem (where $T=\{\hat{t}_1,\hat{t}_2\}$). Suppose that there exist two link-disjoint paths in G' connecting s_1 to t_1 and s_2 to t_2. This implies that there exist two link-disjoint (Steiner) trees in G with root s and terminals $\hat{t}_1$ and $\hat{t}_2$, such that the first tree includes links $(s,v_1),(v_1,\hat{t}_2),(v_1,s_1),(t_1,\hat{t}_1)$, and a path between s_1 and t_1, while the second tree includes links $(s,v_2),(v_2,\hat{t}_1),(v_2,s_2),(t_2,\hat{t}_2)$, and a path between s_2 and t_2. We conclude that $\mathrm{Opt}(\mathbb{N})=0$.

Assume that $\mathbb{N}$ has a network code with no encoding nodes. This implies the existence of two link-disjoint (Steiner) trees with s as their root and terminals t_1 and t_2. Our construction implies that one of the trees includes a path P_1 between s_1 and t_1, while the other includes a path P_2 between s_2 and t_2. Note that paths P_1 and P_2 must be link-disjoint, which completes the proof of our assertion.

To complete the proof of Theorem 4, let n be the number of nodes in the underlying graph of $\mathbb{N}$ and let $\varepsilon>0$ be any constant. Due to the gap location of our reduction, it is clear that $\mathrm{Opt}(\mathbb{N})$ cannot be efficiently approximated by any multiplicative factor unless $\mathcal{P}=\mathcal{NP}$.

[3]Independently, Cheriyan and Salavatipour [14] proposed a similar reduction when addressing the computational hardness of the Steiner tree packing problem. Both reductions assume integral routing, i.e., each path or tree has integer capacity. A more general case of fractional routing is out of the scope of this paper.

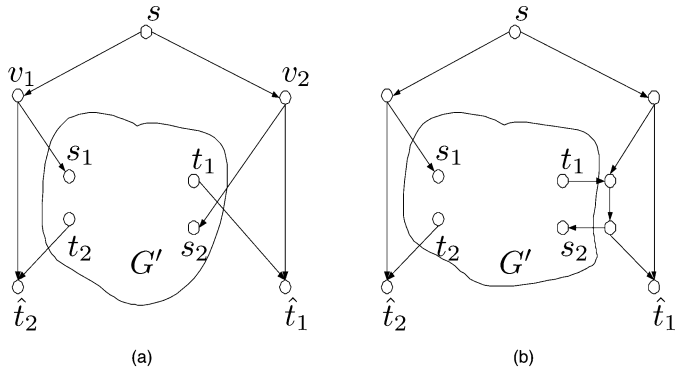

Fig. 6. (a) Reduction from problem LDP to the problem of computing the minimum number of encoding nodes. (b) Reduction from problem NDP to the problem of computing $\mathrm{Opt}_{FES}(\mathbb{N})$. Network $\mathbb{N}(G,s,T,h)$ includes two terminals $\hat{t}_1$ and $\hat{t}_2$ and has $h=2$.

We now sketch the proof for the additive approximation gap. We use a slightly different reduction which guarantees an additive gap of $|V|^{1-\varepsilon}$. Namely, we define an new multicast network

$$\mathbb{N}^{\mathrm{new}}=(G^{\mathrm{new}},s^{\mathrm{new}},T^{\mathrm{new}},h).$$

The new underlying graph G^{new} contains $n^{1/\varepsilon}$ copies of the previous graph G used above. A single source node s^{new} is connected by two links to node s in each copy of G, and two new nodes t_1^{new} and t_2^{new} are added with two links connecting each terminal $\hat{t}_i$ in each copy of G to t_i^{new}. We also set $T^{\mathrm{new}}=\{t_1^{\mathrm{new}},t_2^{\mathrm{new}}\}$ and $h^{\mathrm{new}}=2n^{1/\varepsilon}$. It is not hard to verify that $\mathrm{Opt}(\mathbb{N}^{\mathrm{new}})=0$ if and only if there exist two link-disjoint paths, one between s_1 and t_1 and the second between s_2 and t_2 in G'. Moreover, if $\mathrm{Opt}(\mathbb{N}^{\mathrm{new}})\neq 0$ then it must be the case that $\mathrm{Opt}(\mathbb{N}^{\mathrm{new}})>n^{1/\varepsilon}$ (one encoding node per each copy of G), which suffices to prove our assertion. □

B. Finding Feasible Subgraphs With Small Feedback Link Set

Theorem 30: Let $\varepsilon>0$ be any constant. Let $\mathbb{N}(G,s,T,h)$ be a multicast network in which the underlying graph has m links. Approximating the value of $\mathrm{Opt}_{FES}(\mathbb{N})$ within any multiplicative factor or within an additive factor of $m^{1-\varepsilon}$ is $\mathcal{NP}$-hard.

Proof: Our proof is very similar in nature to that presented in Section VI-A. Namely, we start by presenting a reduction between the node-disjoint path (NDP) problem in directed graphs and the problem of computing $\mathrm{Opt}_{FES}(\mathbb{N})$. Given a directed graph $G=(V,E)$ and two pairs of nodes in V, (s_1,t_1), and (s_2,t_2) the NDP problem is the problem of finding two node-disjoint paths in G connecting s_1 to t_1 and s_2 to t_2. This problem is known to be $\mathcal{NP}$-hard (this can be seen by a simple reduction from problem LDP).

In Fig. 6(b), we show how an instance $\{G',s_1,t_1,s_2,t_2\}$ of problem NDP is reduced to an instance $\mathbb{N}(G,s,T,h)$ of the network coding problem (where $T=\{\hat{t}_1,\hat{t}_2\}$). In this reduction, it is not hard to verify that there exist node-disjoint paths between s_1 and t_1 and between s_2 and t_2 if and only if $\mathrm{Opt}_{FES}(\mathbb{N})=0$ (i.e., there exists a feasible acyclic network $\hat{\mathbb{N}}(\hat{G},s,T,h)$ in which $\hat{G}\subseteq G$). This shows that it is $\mathcal{NP}$-hard to distinguish

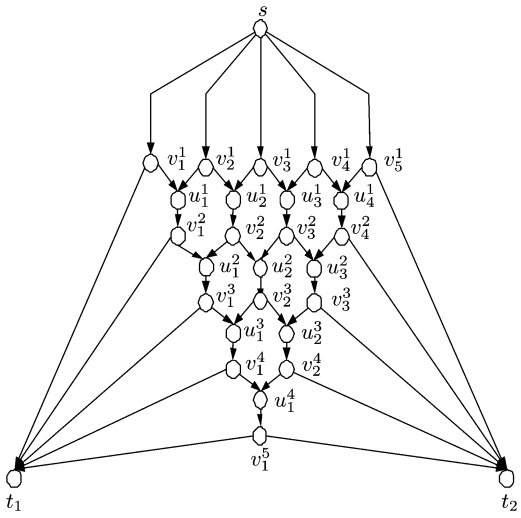

Fig. 7. Lower bound on the number of encoding nodes for $h = 5$.

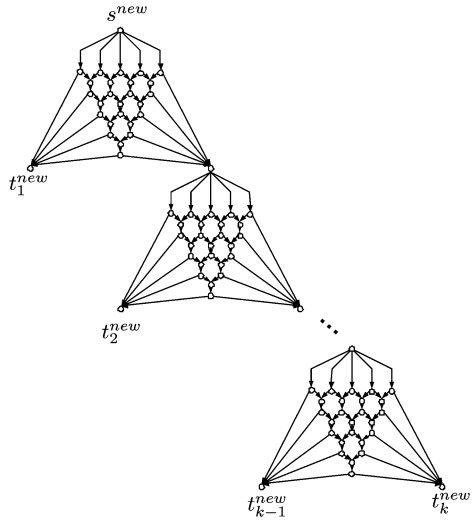

Fig. 8. Lower bound on the number of encoding nodes for $k > 2$.

between networks $\mathbb{N}$ in which $\mathrm{Opt}_{FES}(\mathbb{N}) = 0$ and networks in which $\mathrm{Opt}_{FES}(\mathbb{N}) \geq 1$.

To obtain our inapproximability results, we follow the line of proof given in Section VI-A, and use multiple copies of the reduction above. The detailed proof is omitted. □

C. Theorem 6: Global Lower Bound on $\mathrm{Opt}(\mathbb{N})$

In this subsection, we establish a lower bound on the number of encoding nodes required by an acyclic multicast network (Theorem 6). We start by presenting a lower bound for multicast networks with two terminals ($k = 2$). We then generalize our bound for arbitrary values of k. We start by proving the following lemma.

Lemma 31: Let h be any integer larger than 2. There exist multicast networks $\mathbb{N}(G, s, T, h)$ with $k = |T| = 2$ that require at least $\frac{(h-1)h}{2} = \Omega(h^2)$ encoding nodes.

Proof: For any value of $h \geq 2$, we present a multicast network $\mathbb{N}$ that requires $\frac{(h-1)h}{2}$ encoding nodes. An example of the network $\mathbb{N}$ for $h = 5$ is depicted in Fig. 7. The network includes the following nodes: The source node s and two terminals t_1 and t_2; intermediate v-nodes $v_1^1, \ldots, v_h^1$, $v_1^2, \ldots, v_{h-1}^2, \ldots, v_1^{h-1}$, v_2^{h-1}, and v_1^h; and intermediate u-nodes $u_1^1, \ldots, u_{h-1}^1$, $u_1^2, \ldots, u_{h-2}^2, \ldots, u_1^{h-2}$, u_2^{h-2}, and u_1^{h-1}. The links of the network are defined as follows: s is connected by a link to each of the nodes $v_1^1, \ldots, v_h^1$; each node v_i^1, $i = 1, \ldots, h$ is connected by a link to t_1; each node v_{h-i+1}^i, $i = 1, \ldots, h$ is connected by a link to t_2; each node v_i^j, $j = 1, \ldots, h-1$, $i = 1, \ldots h-j$ is connected by a link to u_i^j; each node v_i^j, $j = 1, \ldots, h-1$, $i = 2, \ldots h-j+1$ is connected by a link to u_{i-1}^j; each node u_i^j is connected to v_i^{j+1}. In the multicast network $\mathbb{N}$, the source node transmits h packets to terminals t_1 and t_2.

We now analyze the properties of $\mathbb{N}$. It is not hard to verify that $\mathbb{N}$ is acyclic, and satisfies the degree constraints specified in Definition 11. We observe that $\mathbb{N}$ is feasible and minimal with respect to link removal. Indeed, the underlying graph G of $\mathbb{N}$ contains h link-disjoint paths from s to each terminal and removal of any link in G will result in a violation of the min-cut condition. This implies that $\mathbb{N}$ is simple and that $\mathrm{Opt}(\mathbb{N}) = \alpha(\mathbb{N})$ (recall that $\alpha(\mathbb{N})$ is the number of nodes in G of in-degree 2, excluding the terminals). It is not hard to verify that there are $\frac{(h-1)h}{2} = \Omega(h^2)$ such nodes (marked as u_i^j). □

To prove Theorem 6, we extend Lemma 31 to capture the general case in which $k \geq 2$. This is done by concatenating many multicast networks $\mathbb{N}$ presented above.

Proof: (of Theorem 6): We build a network $\mathbb{N}^{\mathrm{new}}(G^{\mathrm{new}}, s^{\mathrm{new}}, T^{\mathrm{new}}, h)$ by using the network $\mathbb{N}$ defined in the proof of Lemma 31 as a basic building block. Let G be the underlying network of $\mathbb{N}$. Network G^{new} has one source node s^{new}, and k terminals t_1^{new} to t_k^{new}, and is defined by $k-1$ copies of the graph G (which we denote $G_1, \ldots, G_{k-1}$) as follows. The source s^{new} coincides with the source s of G_1. For $i = 1, \ldots, k-2$, the node t_2 of copy i coincides with the source node s of copy $i+1$. For $i = 1, \ldots, k-1$, node t_1 of G_i coincides with the terminal t_i^{new}. Finally, node t_2 of G_{k-1} coincides with the terminal t_k^{new}.

The construction of $\mathbb{N}^{\mathrm{new}}$ is demonstrated in Fig. 8. As in Lemma 31, it can be shown that the removal of any link from G^{new} will result in a violation of the min-cut condition, implying that each and every link of G^{new} must be used in a network code that transfers all h packets from s^{new} to the terminals of G^{new}. This implies that all u-nodes u_i^j in each copy of G must be encoding nodes, which, in turn, completes our proof. □

D. Theorem 9: Lower Bound for Cyclic Networks

In this subsection, we study multicast networks $\mathbb{N}$ in which the underlying graph is cyclic, and sketch the proof of Theorem 9.

Proof (of Theorem 9): The multicast network $\mathbb{N}(G, s, T, h)$ we suggest is depicted in Fig. 9 (for the case $h = 5$) and is defined as follows. The underlying graph G has a source s, and two terminals t_1 and t_2. There are $h-1$ links leaving s toward the right, and a single link leaving s toward the left. Call the links going right *regular* links and the remaining link leaving s *special* link. The terminal t_1 is connected to s via $h-1$ paths that start at regular links, and a

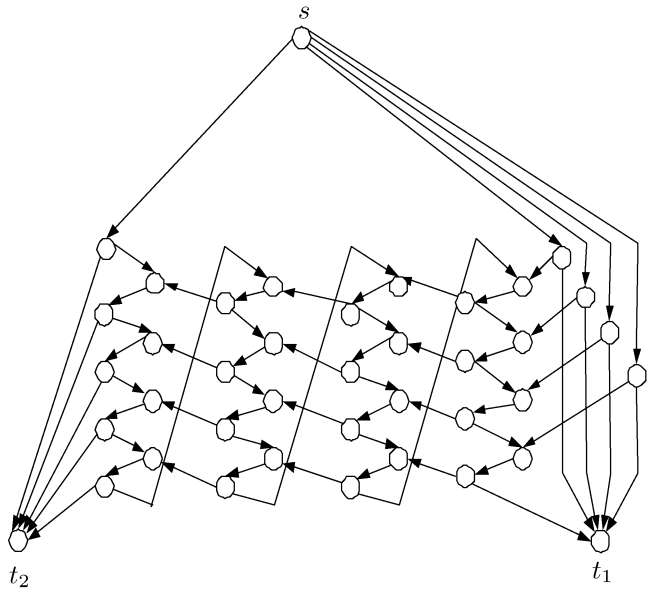

Fig. 9. Lower bound for $\mathrm{Opt}(\mathbb{N})$ in cyclic networks.

single path that starts in the special link. The same holds for t_2. The graph is constructed such that the path from s to t_1 starting at the special link, intersects the paths from s to t_2 starting at regular links (in a systematic manner, as depicted in Fig. 9). Let B be the size of the minimum feedback link set in G. It is not hard to verify that a) the graph G is minimal with respect to link removal (every link appears in a minimum cut of size h), and b) the number of encoding nodes in $\mathbb{N}(G, s, T, h)$ is $(B+1)(h-1) = \Omega(Bh)$. This proves the first assertion. For the second assertion, set $h = 2$ and notice that the number of nodes in G is $2(B+1)(h-1)+h+3 = 2(B+1)+5$, while the number of encoding nodes is $(B+1)(h-1) = B+1$. □

VII. Conclusion

We consider the design of network codes which enable the source to transmit at rate h to k terminals and include a bounded number of encoding nodes. For acyclic networks, we present an efficient and simple procedure which finds a network code in which the number of encoding nodes is independent of the size of the network and is bounded by h^3k^2. We show that our bound on the number of required encoding nodes may depend both on h and k as we present networks in which any feasible network code has at least $\Omega(h^2k)$ encoding nodes. The gap of hk between our lower and upper bounds remains an open problem.

For general (cyclic) networks we present results of similar nature. Namely, we present an upper bound of $(2B+1)h^3k^2$, which depends on the size of the minimum feedback link set B of the network. We also present a lower bound of $\Omega(\max(|V|/2, Bh, h^2k))$, where $|V|$ is the total number of nodes in the network.

Acknowledgment

We would like to thank Matthew Cook for useful discussions.

References

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

[2] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf. Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003.

[3] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, Oct. 2003.

[4] T. Ho, R. Koetter, M. Médard, D. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *Proc. IEEE Int. Symp. Information Theory*, Yokohama, Japan, Jun./Jul. 2003, p. 442.

[5] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Inf, Theory*, to be published.

[6] C. Fragouli, E. Soljanin, and A. Shokrollahi, "Network coding as a coloring problem," in *Proc. Conf. Information Science and Systems*, Princeton, NJ, Mar. 2004.

[7] C. Fragouli and E. Soljanin, "Information flow decomposition for network coding," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 829–848, Mar. 2006.

[8] A. Tavory, M. Feder, and D. Ron, "Bounds on linear codes for network multicast," in *Proc. Electronic Colloquium on Computational Complexity (ECCC)*, vol. 10, 2003.

[9] Y. Wu, K. Jain, and S. Y. Kung, "A unification of Menger's and Edmonds' graph theorems and Aslswede et al's network coding theorem," in *Proc. Allerton Conf. Communications, Control, and Computing*, Monticello, IL, Oct. 2004.

[10] M. R. Garey and D. S. Johnson, *Computers and Intractability*. San Francisco, CA: Freeman, 1979.

[11] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Networks Flows*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[12] K. Menger, "Zur allgemeinen Kurventheorie," *Fund. Math.*, vol. 10, pp. 95–115, 1927.

[13] S. Fortune, J. Hopcroft, and J. Wyllie, "The directed subgraph homeomorphism problem," *Theor. Comp. Sci.*, vol. 10, no. 2, pp. 111–121, 1980.

[14] J. Cheriyan and M. R. Salavatipour, "Hardness and approximation results for packing Steiner trees," in *Proc. Europ. Symp. Algorithms (ESA 2004)*, Sep. 2004.