Network Coding: A Computational Perspective

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

***Abstract*—In this work, we study the computational perspective of network coding, focusing on two issues. First, we address the computational complexity of finding a network code for acyclic multicast networks. Second, we address the issue of reducing the amount of computation performed by network nodes. In particular, we consider the problem of finding a network code with the minimum possible number of *encoding nodes*, i.e., nodes that generate new packets by performing algebraic operations on packets received over incoming links.**

We present a deterministic algorithm that finds a feasible network code for a multicast network over an underlying graph $G(V, E)$ in time $O(|E|kh + |V|k^2h^2 + h^4k^3(k + h))$, where k is the number of destinations and h is the number of packets. Our algorithm improves the best known running time for network code construction. In addition, our algorithm guarantees that the number of encoding nodes in the obtained network code is upper-bounded by $O(h^3k^2)$.

Next, we address the problem of finding integral and fractional network codes with the minimum number of encoding nodes. We prove that in the majority of settings this problem is $\mathcal{NP}$-hard. However, we show that if $h = O(1)$, $k = O(1)$, and the underlying communication graph is acyclic, then there exists an algorithm that solves this problem in polynomial time.

***Index Terms*—Algorithms, computational perspective, encoding complexity, fractional network coding, integer network coding, multicast connections.**

I. Introduction

THE new paradigm of network coding promises to benefit many areas of communication and networking [1], [2]. The network coding approach generalizes traditional routing by allowing intermediate network nodes to generate new packets by performing algebraic operations on incoming data packets.

Establishing efficient multicast connections is a central problem in network coding. In the *multicast network coding problem* a source s needs to deliver h packets to a set T of k terminals over the underlying communication graph $G(V, E)$. It was shown in [2] and [3] that the capacity of the network, i.e., the maximum number of packets that can be sent from s to T per time unit, is equal to the minimum capacity of a cut that separates the source s from a terminal $t \in T$. Specifically, a source s can send h packets to all terminals T if and only if the total capacity of all links in any cut that separates s and $t \in T$ is at least h. Li *et al.* [3] proved that *linear network codes* are sufficient for achieving the capacity of the network. In a subsequent work, Koetter and Médard [4] 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.* [5] to show that linear network codes can be efficiently constructed through a randomized algorithm. Jaggi *et al.* [1] proposed a deterministic polynomial-time algorithm for finding feasible network codes in multicast networks.

In this paper, we study the computational perspective of multicast network coding. Our goal is to minimize both the time required for finding a feasible network code and the amount of computation performed by network nodes. In particular, we consider the problem of finding a network code that uses a bounded number of *encoding nodes*. Encoding nodes generate new packets by combining the packets received over incoming links, in contrast to *forwarding* nodes that can only forward and duplicate incoming packets.

We focus on both *integral* and *fractional* network codes. In fractional network codes, each packet can be split into a number of smaller packets, each of which is sent over different paths. In an integral network, codes packets cannot be split and have to be sent through the network in one piece. We also assume that all packets sent over any given link are generated by using the same algebraic operation (different links can use different operations).

A. Our Results

Our study makes the following contributions. First, we present an efficient algorithm for finding integral network codes. Given an acyclic multicast network with h packets and k terminals, our algorithm finds a network code that includes at most $O(h^3k^2)$ encoding nodes. The computational complexity of our algorithm is $O(|E| + |V|k^2 + k^4)$ for $h = 2$ and $O(|E|kh + |V|k^2h^2 + h^4k^3(k + h))$ for general h. To the best of our knowledge, this is the first efficient algorithm that identifies a feasible network code with a bounded number of encoding nodes. In addition, our algorithm improves the previously best known running time of $O(|E|kh + |V|k^2h^2(k + h))$ of the algorithm due to Jaggi *et al.* [1].[1] The improvement is most significant for sparse graphs with a large number of nodes, which is a typical case in communication networks.

The key idea of our algorithm is to transform a given multicast network into an auxiliary network in which the number

Manuscript received July 17, 2006; revised January 21, 2008. Current version published December 24, 2008. This work was supported in part by the Caltech Lee Center for Advanced Networking.

M. Langberg is with the Computer Science Division, The Open University of Israel, Raanana 43107, Israel (e-mail: mikel@openu.ac.il).

A. Sprintson is with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77845 USA (e-mail: spalex@tamu.edu).

J. Bruck is with the Department of Electrical Engineering, California Institute of Technology, Pasadena, California 91125 USA (e-mail: bruck@caltech.edu).

Communicated by M. Médard, Associate Editor for Communications.

Color versions of Figures 3 and 6–8 in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIT.2008.2008135

[1]To be more precise, the running time of our algorithm is $O(\min(|E|kh + |V|k^2h^2 + h^4k^3(k + h)), |E|k^2h + h^4k^3(k + h))$ and the running time of the algorithm of [1] is $O(\min(|E|kh + |V|k^2h^2(k + h), |E|kh(k + h)))$. This distinction is of significance when the size of the link set E is very small. Throughout this work we will refer to the running times of our algorithm and that of [1] without this distinction. In both cases, our algorithm suggests an improved running time.

Type of network code	Restrictions	Acyclic	General(cyclic)
Integral	k and h are constant	$\|V\|^{O(h^6k^4)}$ (Theorem 11)	$\mathcal{NP}$-hard (Theorem 9)
	no restrictions	$\mathcal{NP}$-hard (Theorem 10)	
Fractional	k and h are constant	$\|V\|^{O(h^3k^2)}$ (Theorem 16)	Not resolved in this work
	no restrictions	$\mathcal{NP}$-hard (Theorem 17)	$\mathcal{NP}$-hard (Theorem 17)

Fig. 1. Our results for the problem of finding a network code with the minimum possible number of encoding nodes.

of links is upper-bounded by $O(h^3k^2)$. This transformation is accomplished through an efficient procedure that exploits the structural properties of minimal multicast networks (as defined in [6]). The auxiliary network we construct is equivalent to the original network, i.e., a feasible network code for the auxiliary network yields a feasible network code for the original network. A feasible network code for the auxiliary network is constructed by invoking the algorithm due to Jaggi *et al.* [1] as a subroutine.

Second, we study the problem of finding integral and fractional network codes with the minimum possible number of encoding nodes. We prove that in the majority of settings this problem is $\mathcal{NP}$-hard. However, we show that if $h = O(1)$, $k = O(1)$, and the underlying communication graph is acyclic, then the problem can be solved in polynomial time. Our results are summarized in Fig. 1.

B. Related Work

The fastest deterministic algorithm for finding feasible network codes for multicast networks prior to our work was due to Jaggi *et al.* [1]. Randomized algorithms for this problem have been presented in [1] and [5]. For acyclic graphs, the currently best known randomized algorithm has expected running time of $O(|E|kh+kh^{2.376})$ when the packet size depends on the size of the network G and $O(|E|kh+|V|k^2h^3)$ when the packet size is independent of the size of G [1]. Recently, [7] observed a connection between a problem of finding a feasible network code and the problem of maximizing the rank of mixed matrices, i.e., the matrices whose entries can be both numeric values and symbolic variables. In particular, [7] proposed an algorithm that addresses a more general class of problems and has a running time of $\min(O(k|E|^3\log|E|), O(|E|kh + |V|^3k^3h^3\log(|V|h)))$. However, none of the previous algorithms provides a nontrivial upper bound on the number of encoding nodes in the network.

In a previous work of ours [6] we established both a lower bound of $\Omega(h^2k)$ and an upper bound of h^3k^2 on the minimum number of encoding nodes in integral network codes. In addition, we showed that there exists a polynomial-time algorithm that finds a network code with a bounded number of encoding nodes. We also proved that finding the minimum number of encoding nodes in integral network codes with cycles is an $\mathcal{NP}$-hard problem. The algorithm presented in [6] is based on a simple greedy approach and has high computational complexity. Specifically, its running time is at least quadratic in the size of the network, while the running time of the algorithm presented in this paper is linear in the size of the network.

A recent work by Bhattad *et al.* [8] considered several optimization problems related to fractional network codes in multicast networks. This work models the flow of information in a coding network by a linear program with $O(|E|2^k)$ variables. For small values of k this program enables to optimize several objective functions that are strongly related to the number of encoding nodes in coding networks. We use the framework of [8] in parts of this work.

The rest of the paper is organized as follows. In Section II, we formulate the network model and present the definition of integral network codes. In Section III, we present an algorithm for finding integral network codes with a bounded number of encoding nodes. In Section IV, we define fractional network codes and analyze the computational complexity of minimizing the number of encoding nodes in both fractional and integral network codes.

II. Model

In this section, we define integral network codes. A more general setting of fractional network codes is discussed in Section IV.

The communication network is represented by a directed graph $G = (V, E)$, where V is the set of nodes and E the set of links in G. The capacity c_e of a link $e \in E$ is defined to be the number of packets that can be sent over e in one time unit. We assume that link capacities c_e are integer numbers. An instance $\mathbb{N}(G, s, T, h)$ of the multicast network coding problem is a 4-tuple that includes the graph $G(V, E)$, a source node $s \in V$, a set $T \subset V$ of terminals, and the number of packets h that must be transmitted from the source node s to every terminal $t \in T$. We assume, without loss of generality, that i) the source s has no incoming links; ii) the source node s has exactly h outgoing links of capacity one that transmit h original packets; iii) each terminal $t \in T$ has no outgoing links. Indeed, if i) and ii) do not hold, we can add a new source $\hat{s}$ connected to the original source s by h parallel links. Similarly, if iii) does not hold for some $t \in T$, we can add a new terminal $\hat{t}$ and add h parallel links between t and $\hat{t}$. We also assume that each packet is an element of a finite field Σ. We denote the size $|T|$ of the terminal set by k. Each node $v \in G$, $v \neq s$, $v \notin T$ is referred to as an *internal* node.

Definition 1 (Integral Network Code $\mathbb{F}(\mathbb{N})$): A network code for $\mathbb{N}(G, s, T, h)$ is defined by the set of encoding functions $\mathbb{F}(\mathbb{N}) = \{f_e | e \in E\}$. If $e(v, u)$ is an outgoing link of the source node s, then f_e is a mapping from Σ^h to Σ. Otherwise, f_e is a mapping from $\Sigma^{c_{\text{in}}(v)}$ to Σ^{c_e}, where $c_{\text{in}}(v) = \sum_{w:(w,v)\in E} c_{(w,v)}$ is the total capacity of the incoming links of v.

For $v \neq s$, the encoding function f_e of $e(v, u)$ determines the packets transmitted on link e for any possible combination of packets received over the incoming links of v. As s is assumed to have exactly h outgoing links, for $v = s$ the encoding function f_e of $e(v, u)$ is, without loss of generality, an identity function of a single variable.

We focus on linear network codes $\mathbb{F}(\mathbb{N})$, i.e., for each $e \in E$ the encoding function f_e is a linear function over Σ. With linear network coding, each packet transmitted over link $e \in E$ is a linear combination of the h packets available at source s. Accordingly, for each link $e \in E$ we define the *global encoding function* $F_e : \Sigma^h \mapsto \Sigma^{c_e}$ that determines the packets transmitted on link e as a function of the packets available at s. If e is an outgoing link of the source node, then F_e is identical to f_e. For any other link $e(v,u) \in E$, F_e is defined as $F_e \equiv f_e(F_{e_1^v}, \ldots, F_{e^v_{d_{\text{in}}(v)}})$, where $\{e_1^v, \ldots, e^v_{d_{\text{in}}(v)}\}$ is the set of incoming links of v and $d_{\text{in}}(v)$ is the in-degree of v.

A network code $\mathbb{F}(\mathbb{N})$ for an acyclic coding network $\mathbb{N}(G,s,T,h)$ is said to be feasible if for each destination node $t \in T$, there exists a decoding function $g_t : \Sigma^{c_{\text{in}}(t)} \mapsto \Sigma^h$ such that $g_t(F_{e_1^t}, \ldots, F_{e^t_{d_{\text{in}}(t)}})$ is the identity function with respect to the original packets, where $\{e_1^t, \ldots, e^t_{d_{\text{in}}(t)}\}$ is the set of incoming links of t and $d_{\text{in}}(t)$ is the in-degree of t.

To define the feasibility of a network code over a cyclic coding network, we need to consider multiple rounds of transmission, at each round the source sends h packets over the network. We say that the network code allows transmission at rate h if each terminal $t \in T$ can decode the packets sent by the source, such that each packet is decoded after a fixed number of rounds.

An instance $\mathbb{N}(G,s,T,h)$ of the multicast network coding problem is said to be feasible if there exists a feasible network code for $\mathbb{N}$. If $\mathbb{N}(G,s,T,h)$ is feasible we refer to h as the *rate* of the multicast coding network. The multicast capacity of the communication network G with respect to source s and set T of terminals is defined to be the maximum value of h such that the coding network $\mathbb{N}(G,s,T,h)$ is feasible. The *capacity* of the network is determined by the minimum capacity of a cut that separates the source s and any terminal $t \in T$ [2], where the capacity of a cut is the sum of the capacities of the links that belong to the cut.

A coding network $\mathbb{N}(G,s,T,h)$ is said to be *minimal* with respect to link removal if i) $\mathbb{N}(G,s,T,h)$ is feasible. ii) Removal of any link from G would violate the feasibility of $\mathbb{N}(G,s,T,h)$.

Let $\mathbb{N}(G,s,T,h)$ be a feasible coding network. We say that a link $e \in G$ is *vital* if after removing e from G the resulting network is no longer feasible. Note that every link of the minimal network is vital.

Definition 2 (Encoding and Forwarding Links and Nodes): Let $\mathbb{F}(\mathbb{N})$ be a network code. A link $e(v,u)$ is referred to as a *forwarding link* if it is an outgoing link of the source node s or if f_e can be decomposed to c_e functions $f_e^1, \ldots, f_e^{c_e}$ that map $\Sigma^{c_{\text{in}}(v)}$ to Σ such that each function f_e^i depends only on one variable, where $c_{\text{in}}(v) = \sum_{w:(w,v)\in E} c_{(w,v)}$. Otherwise, link e is referred to as an *encoding 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, then the node is referred to as a *forwarding node*.

Encoding links generate new packets by combining the packets received over the incoming links; forwarding links can only forward incoming packets.

We will use the following lemma implied by [6].

Lemma 3 ([6]): Let $\mathbb{N}(G,s,T,h)$ be an acyclic coding network such that $|T| = 2$, all links in G are of unit capacity, the total degree of each internal node in G is at most 3, and G is minimal with respect to link removal. Then, the number of internal nodes in G with in-degree larger than 1 is bounded by h^3, and the number of internal nodes in G with out-degree larger than 1 is bounded by $h^3 + h$.

Let $\mathbb{N}(G(V,E),s,T,h)$ and $\hat{\mathbb{N}}(\hat{G}(\hat{V},\hat{E}),\hat{s},\hat{T},\hat{h})$ be multicast coding networks. We say that $\hat{\mathbb{N}}$ *models* $\mathbb{N}$ if the following three conditions hold: i) $\mathbb{N}$ is feasible if and only if $\hat{\mathbb{N}}$ is feasible. ii) For any feasible network code $\hat{\mathbb{F}}(\hat{\mathbb{N}})$ for $\hat{\mathbb{N}}$, there exists a corresponding network code $\mathbb{F}(\mathbb{N})$ for $\mathbb{N}$ that includes the same number of encoding nodes or less. iii) Given a feasible network code $\hat{\mathbb{F}}(\hat{\mathbb{N}})$ for $\hat{\mathbb{N}}$, the corresponding network code $\mathbb{F}(\mathbb{N})$ for $\mathbb{N}$ can be found through an efficient procedure whose running time is bounded by $O(|E| + |\hat{E}|)$.

In some parts of our paper we use a notion of network flows [9].

Definition 4 (Flow): An *integral* (s,t)*-flow* θ is a function $\theta : E \mapsto \mathbb{R}$ that satisfies the following two properties.

1) For all $e(u,v) \in E$, it holds that $\theta(e)$ is an integer number that satisfies $0 \leq \theta(e) \leq c_e$.
2) For each internal node $v \in V$, $v \neq s$, $v \notin T$ it holds that

$$\sum_{w:(w,v)\in E} \theta((w,v)) = \sum_{w:(v,w)\in E} \theta((v,w)).$$

The *value* $|\theta|$ of a flow θ is defined as

$$|\theta| = \sum_{v:(s,v)\in E} \theta((s,v)).$$

If each link $e \in E$ is associated with a cost ω_e then the cost $\omega(\theta)$ of a flow θ is defined as follows:

$$\omega(\theta) = \sum_{(u,v)\in E} \omega_{(u,v)} \cdot \theta((u,v)). \tag{1}$$

A minimum cost (s,t)-flow θ can be decomposed into a set of $|\theta|$ paths between s and t [9].

III. Algorithm for Computing a Feasible Network Code

In this section, we present an algorithm that receives as input an acyclic coding network $\mathbb{N}(G,s,T,h)$ and computes a feasible integral network code for $\mathbb{N}$ over a field of size $k = |T|$. The computational complexity of our algorithm is $O(|E|kh + |V|k^2h^2 + h^4k^3(h+k))$.

A. Algorithm Overview

Our algorithm uses three auxiliary coding networks $\mathbb{N}'(G',s,T,h)$, $\mathbb{N}^*(G^*,s,T,h)$, and $\hat{\mathbb{N}}(\hat{G},s,T,h)$, all of them model $\mathbb{N}(G,s,T,h)$.

The coding network $\mathbb{N}'(G',s,T,h)$ is constructed by Procedure EXPAND, described in Section III-B. This network has the following properties: i) All links in G' are of capacity one. ii) The total number of links in G' is bounded by $|V|hk$. iii) Each internal node $v \in G'$ has either in-degree one or out-degree one.

The last property implies that any two link-disjoint paths in G' are also node-disjoint.

Next, we apply Algorithm MIN-GLOBAL, described in Section III-D below. The algorithm constructs the auxiliary network $\mathbb{N}^*(G^*, s, T, h)$ by deleting redundant links from G' such that the number of nodes of in-degree more than two in G^* is bounded by $O(h^3k^2)$. Finally, we invoke Procedure SHRINK, described in Section III-E below. This procedure constructs the coding network $\hat{\mathbb{N}}(\hat{G}, s, T, h)$ by contracting all nodes in $\mathbb{N}^*(G^*, s, T, h)$ of degree two. The number of links in $\hat{\mathbb{N}}(\hat{G}, s, T, h)$ is bounded by $O(h^3k^2)$.

We find a network code for $\mathbb{N}$ by performing the following steps:

1) construct an auxiliary coding network $\hat{\mathbb{N}}(\hat{G}, s, T, h)$;
2) find a feasible network code $\hat{\mathbb{F}}(\hat{\mathbb{N}})$ for $\hat{\mathbb{N}}$, e.g., by applying the algorithm due to Jaggi *et al.* [1];
3) find a network code $\mathbb{F}(\mathbb{N})$ for $\mathbb{N}$ that corresponds to $\hat{\mathbb{F}}(\hat{\mathbb{N}})$.

B. *Procedure* EXPAND

Procedure EXPAND begins by assigning a unit cost for each link $e \in E$. Then, the procedure finds, for each terminal $t_i \in T$, a minimum cost (s, t_i)-flow θ_i of value h. In order to find a minimum cost flow we employ the Successive Shortest Path algorithm [9, Ch. 9]. Next, each link $e(v, u) \in E$ is substituted by $\max_{t_i \in T} \theta_i(e)$ parallel links of unit capacity that connect v and u. All links for which $\max_{t_i \in T} \theta_i(e) = 0$ are removed from the graph. Note that the resulting graph contains h link-disjoint paths between source s and any terminal $t_i \in T$. We denote a set of h link-disjoint paths between s and t_i by $\mathbb{P}_i$. The sets $\{\mathbb{P}_i \,|t_i \in T\}$ can be found by invoking the flow decomposition algorithm [9].

Finally, the procedure substitutes each internal node v in the resulting graph that has both input and output degrees more than two by a gadget Γ_v, constructed as follows: Let E_v^{in} and E_v^{out} be the incoming and outgoing links of v, respectively. For every link $(x, v) \in E_v^{\text{in}}$, we add a node x' and a link (x, x') to Γ_v. Similarly, for every link $(v, y) \in_v^{\text{out}}$ we add a node y' and a link (y', y) to Γ_v. For each path $P_i \in \{\mathbb{P}_i | t_i \in T\}$ let x' be a node in Γ_v that corresponds to link $(x, v) \in P_i$ and y' be a node in Γ_v that corresponds to link $(v, y) \in P_i$. Then, if Γ_v does not already include link (x', y'), we add (x', y') to Γ_v. Fig. 3 demonstrates the construction of the subgraph Γ_v. The resulting graph is denoted by $G'(V', E')$. Note that each internal node $v \in G'$ has either in-degree one or out-degree one.

The formal description of Procedure EXPAND appears in Fig. 2. We proceed to analyze the computational complexity of the procedure. Finding flows $\{\theta_i | t_i \in T\}$ and decomposing them into paths $\{\mathbb{P}_i | t_i \in T\}$ can be done in time $O(|E|kh)$ [9]. Each path P in $\bigcup_{t_i \in T} \mathbb{P}_i$ is of length at most $|V|$, and corresponds to a path of length at most $2|V|$ in G'. The latter follows from the structure of gadget Γ_v. Indeed, after substituting a node v by a gadget Γ_v any path (x, v, y) of G is replaced by a path (x, x', y', y) in G'. This implies that the number of links in G' is bounded by $O(|V|kh)$. Thus, the computational complexity of Procedure EXPAND is bounded by $O(|E|kh)$. In Theorem 7 below we show that the coding network $\mathbb{N}'(G', s, T, h)$ returned by Procedure EXPAND models the original coding network $\mathbb{N}(G, s, T, h)$.

Procedure EXPAND ($\mathbb{N}(G, s, T, h)$):

Input:
$\mathbb{N}$ - a feasible coding network;

1 Assign a unit cost for every link $e \in E$.
2 **for** each terminal $t_i \in T$ **do**
3 Find a minimum cost (s, t_i)-flow θ_i of value h.
4 **for** each link $e(v, u) \in E$ **do**
5 **if** $\max_{t_i \in T} \theta_i(e) > 0$ **then**
6 Replace e by $\max_{t_i \in T} \theta_i(e)$ parallel links of capacity one between v and u
7 **else**
8 Remove e
9 **for** each terminal $t_i \in T$ **do**
10 Find a set $\mathbb{P}_i$ of h link-disjoint paths between s and t_i.
11 For each node $v \in G$, $v \neq s$, $v \notin T$ whose degree is more than 3, replace v by a gadget Γ_v.
12 Denote by G' the resulting graph.
13 Return $\mathbb{N}'(G', s, T, h)$.

Fig. 2. Procedure EXPAND.

C. *Algorithm* MIN-LOCAL

We proceed to describe Algorithm MIN-LOCAL, which is an important building block of Algorithm MIN-GLOBAL, presented in the next section.

Algorithm MIN-LOCAL receives as input a coding network $\mathbb{N}'(G', s, T, h)$ (returned by Procedure EXPAND) and two sets of h node-disjoint paths $\mathbb{P}_i$, $\mathbb{P}_j$ that connect source s to terminals t_i and t_j, respectively. The algorithm finds two sets of h node-disjoint paths $\mathbb{P}_i^*, \mathbb{P}_j^*$ connecting s to t_i and s to t_j such that the subgraph G_{ij}^* of G' induced by paths in $\mathbb{P}_i^* \cup \mathbb{P}_j^*$ is minimal with respect to link removal. That is, removal of any link from G_{ij}^* results in a reduction of the value of the minimum cut between s and t_i or s and t_j. The formal description of Algorithm MIN-LOCAL appears in Fig. 4.

Lemma 5: Let G_{ij}^* be a subgraph of G induced by path sets $\mathbb{P}_i^*$ and $\mathbb{P}_j^*$ returned by Algorithm MIN-LOCAL and let $\mathbb{N}_{ij}^*(G_{ij}^*, s, \{t_i, t_j\}, h)$ be a coding network formed by G_{ij}^* and the two terminals, t_i and t_j. Then, all links in $\mathbb{N}_{ij}^*(G_{ij}^*, s, \{t_i, t_j\}, h)$ are vital.

Proof: We start by showing that all links that belong to paths in $\mathbb{P}_j^*$ are vital in $\mathbb{N}_{ij}^*(G_{ij}^*, s, \{t_i, t_j\}, 2)$. We assume, by way of contradiction, that there exists a link $e \in \mathbb{P}_j^*$, which is not vital in $\mathbb{N}_{ij}^*$. Then, there exist h link-disjoint paths $\hat{\mathbb{P}}_j$ between s and t_j in G_{ij}^* that do not use link e.

Consider graph G'_{ij} created at Step 4 (with link costs assigned at Step 2). Since G_{ij}^* is a subgraph of G'_{ij}, all paths in $\hat{\mathbb{P}}_j$ exist in G'_{ij}. We denote by $\hat{f}$ and f^* the flows defined by sets of disjoint paths $\hat{\mathbb{P}}_j$ and $\mathbb{P}_j^*$, respectively. We denote by $G'_{ij}(f^*)$ the residual graph of G'_{ij} with respect to flow f^*. That is, $G'_{ij}(f^*)$ is formed from G'_{ij} by reversing links that belong to paths in $\mathbb{P}_j^*$ and negating their cost. Note that the cost of every link $e \in G'_{ij}(f^*)$ is either 0 or -1. By the Augmenting Cycle Theorem [9, Ch. 3], flow $\hat{f}$ is equal to flow f^* plus flow along a set of directed cycles $\mathbb{C}$ in $G'_{ij}(f^*)$. Let C be a cycle in $\mathbb{C}$. Note that $\mathbb{C}$ must contain at least one cycle because flows f^* and $\hat{f}$ are not identical. Indeed, flow $\hat{f}$ contains link e, while f^* does not contain it.

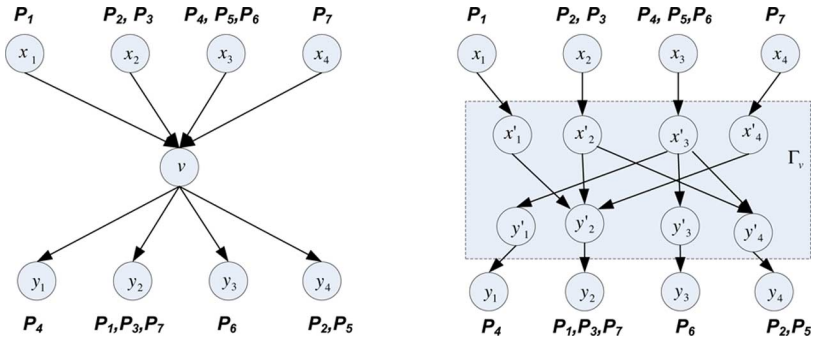

Fig. 3. Substituting a node v by a gadget Γ_v.

Algorithm MIN-LOCAL *($\mathbb{N}'(G', s, T, h)$, $\mathbb{P}_i, \mathbb{P}_j$):*

Input:

$\mathbb{N}'$ - a feasible coding network,
$\mathbb{P}_i$ - a set of h node-disjoint paths between s and t_i,
$\mathbb{P}_j$ - a set of h node-disjoint paths between s and t_j;

1 $G_{ij} \leftarrow$ the subgraph of G' induced by links that belong to paths in $\mathbb{P}_i$ and $\mathbb{P}_j$.
2 Assign zero cost to every link that belongs to a path in $\mathbb{P}_i$ and assign a unit cost to every other link in G_{ij}.
3 Find h link-disjoint paths $\mathbb{P}_j^*$ in G_{ij} between s and t_j of minimum total cost.
4 $G'_{ij} \leftarrow$ the subgraph of G_{ij} induced by links that belong to paths in $\mathbb{P}_i$ and $\mathbb{P}_j^*$.
5 Assign zero cost to every link that belongs to a path in $\mathbb{P}_j^*$ and assign a unit cost to every other link in G'_{ij}.
6 Find h link-disjoint paths $\mathbb{P}_i^*$ in G'_{ij} between s and t_i of minimum total cost.
7 Return $\mathbb{P}_i^*$ and $\mathbb{P}_j^*$.

Fig. 4. Algorithm MIN-LOCAL.

Algorithm MIN-GLOBAL *($\mathbb{N}'(G', s, T, h)$):*

Input:

$\mathbb{N}'(G', s, T, h)$ - a feasible coding network;

1 $\mathbb{P} \leftarrow \emptyset$.
2 **for** $i \leftarrow 1$ to k **do**
3 Assign zero cost to all links in $E(\mathbb{P})$. Assign unit costs to all other links in G'.
4 Find a set of h link-disjoint paths $\mathbb{P}_i$ in G' between s and t_i of minimum total cost.
5 **if** $i > 1$ **do**
6 **for** $j \leftarrow 1$ **to** $i-1$ **do**
7 $\mathbb{P}_i, \mathbb{P}_j \leftarrow$ MIN-LOCAL$(\mathbb{N}'(G', s, T, h), \mathbb{P}_i, \mathbb{P}_j)$
8 $\mathbb{P} \leftarrow \bigcup_{j=1}^{i} \mathbb{P}_j$ and $E_i \leftarrow \bigcup_{j=1}^{i} E(\mathbb{P}_j)$.
9 $G^* \leftarrow$ a subgraph of G' induced by links in E_k.
10 Return $\mathbb{N}^*(G^*, s, T, h)$.

Fig. 5. Algorithm MIN-GLOBAL.

We note that the cost of $\hat{f}$ equals the cost of f^* plus the cost of the flow on cycles in $\mathbb{C}$, where the cost of a flow is defined with respect to costs assigned at Step 2. Since $\mathbb{P}_j^*$ is a minimum cost set of disjoint paths, the cost of flow $\hat{f}$ is greater than or equal to that of f^*. This implies that all cycles in $\mathbb{C}$ contain only zero cost links, i.e., links that belong to paths in $\mathbb{P}_i$. This, however, contradicts the fact that $\mathbb{P}_i$ is a set of node-disjoint paths and that terminal t_i has no outgoing links in $G'_{ij}(f^*)$.

We have proven so far that every link in $\mathbb{P}_j^*$ is vital. It remains to show that every link that belongs to a path in $\mathbb{P}_i^*$ but does not belong to a path in $\mathbb{P}_j^*$ is vital. We note that each such link was assigned cost one at Step 5. Hence, if one of these links is not vital, this would contradict the minimality of $\mathbb{P}_i^*$. □

It is not hard to verify that Algorithm MIN-LOCAL runs in time $O(|V|h^2)$ (again we use the augmenting path approach to find h link-disjoint paths).

D. *Algorithm* MIN-GLOBAL

Algorithm MIN-GLOBAL receives, as input, a feasible coding network $\mathbb{N}'(G', s, T, h)$ returned by Procedure EXPAND. First, the algorithm iteratively constructs, for each $t_i \in T$, a set $\mathbb{P}_i$ of h link-disjoint paths between s and t_i. We denote by $E(\mathbb{P}_i)$ the set of links that belong to paths in $\mathbb{P}_i$, by E_i the union $\cup_{j=1}^{i} E(\mathbb{P}_j)$, and by G^* the subgraph of G' induced by links in E_k. Our goal is to ensure that the total number of links in G^* which are incoming links of nodes of in-degree 2 or more is bounded by $O(h^3k^2)$. To that end, we first minimize the number of links in $E_i \setminus E_{i-1}$. In addition, we apply Algorithm MIN-LOCAL for $\mathbb{P}_i$ and $\mathbb{P}_j$, $1 \le j < i$, in order to further delete nonvital links from E_i. The algorithm returns a coding network $\mathbb{N}^*(G^*, s, T, h)$. The formal description of Algorithm MIN-GLOBAL appears in Fig. 5.

Theorem 6: Let $\mathbb{N}^*(G^*(V^*, E^*), s, T, h)$ be the coding network returned by Algorithm MIN-GLOBAL$(\mathbb{N}'(G', s, T, h))$. Let $\bar{V}^*$ be the subset of $V^* \setminus T$ that includes nodes of in-degree two or more and let $\bar{E}^*$ be the set of incoming links of nodes in $\bar{V}^*$. Then, it holds that $|\bar{E}^*| = O(h^3k^2)$.

Proof: We denote by $G_i(V_i, E_i)$ the subgraph of G' induced by links in E_i. We also denote by $\bar{V}_i$ the subset of $V_i \setminus T$ that includes nodes of in-degree two or more and by $\bar{E}_i$ the set of incoming links of nodes in $\bar{V}_i$. We prove, by induction on i, that $|\bar{E}_i|$ is bounded by $2h^3ki$.

For the base step, we note that $|\bar{E}_2|$ is bounded by $2h^3$. Indeed, Lemma 5 implies that the subgraph $G_{1,2}$ induced by links in $E(\mathbb{P}_1) \cup E(\mathbb{P}_2)$ is minimal with respect to link removal. Hence, by Lemma 3, the number of nodes in $\bar{V}_2$ is bounded by h^3, each node in $\bar{V}_2$ has two incoming links.

For the induction step, we prove that for $i = 3, \ldots, k$ it holds that $|\bar{E}_i| \le 2h^3ki$. We divide the set $\bar{E}_i$ into two subsets $\bar{E}_i^1 =$

$\bar{E}_i \cap \bar{E}_{i-1}$ and $\bar{E}_i^2 = \bar{E}_i \setminus \bar{E}_i^1$. By the inductive argument, the number of links in $\bar{E}_i^1$ is bounded by $2h^3k(i-1)$. Thus, in order to complete the proof we need to bound the number of links that belong to $\bar{E}_i^2$.

We denote by G_{ij} the subgraph of G induced by links in $E(\mathbb{P}_i) \cup E(\mathbb{P}_j)$ in step 7 and by $\bar{E}_{ij}$ the set of incoming links of nodes of in-degree two in G_{ij}. Lemma 5 implies that the coding network $\mathbb{N}_{ij}(G_{ij}, s, \{t_i, t_j\}, h)$ is minimal with respect to link removal. Hence, by Lemma 3, each of the sets $\bar{E}_{ij}$, $j \in \{1, \ldots, i-1\}$ contains at most $2h^3$ links. We show that each link in $\bar{E}_i^2$ belongs to $\bar{E}_{ij}$ for some $j \in \{1, \ldots, i-1\}$, which in turn, implies that $|\bar{E}_i^2| \leq 2h^3k$ which concludes our assertion.

Let $e = (u, v)$ be a link in $\bar{E}_i^2$. We consider two cases.

1) Link $e = (u, v)$ belongs to $E_i \setminus E_{i-1}$. This implies that e belongs to $E(\mathbb{P}_i)$. In fact, e belongs to $E(\mathbb{P}_i)$ at any time during iteration i of the main loop (the loop that begins on line 2). Indeed, otherwise, there would exist a set of disjoint paths between s and t_i that has a smaller cost than that selected in line 4, resulting in a contradiction. Since the in-degree of v is at least two, v has an additional incoming link $e' = (w, v)$. Note that $e' \notin \mathbb{P}_i$ because $\mathbb{P}_i$ only contains node-disjoint paths. We conclude that e belongs to $\bar{E}_{ij}$ for some $j \in \{1, \ldots, i-1\}$.
2) Link $e = (u, v)$ belongs to E_{i-1}. This implies that in G_{i-1} node v has in-degree one. Since the in-degree of v in E_i is at least two, v has an additional incoming link $e' = (w, v)$. Such link must belong to $E_i \setminus E_{i-1}$, and, in turn to $E(\mathbb{P}_i)$ (due to the same argument as in case 1). This implies that e belongs to $\bar{E}_{ij}$ for some $j \in \{1, \ldots, i-1\}$. □

Note that $\mathbb{N}^*(G^*, s, T, h)$ is a feasible network obtained from $\mathbb{N}'(G', s, T, h)$ by deleting redundant links. Thus, $\mathbb{N}^*(G^*, s, T, h)$ models $\mathbb{N}'(G', s, T, h)$, and, in turn, models $\mathbb{N}(G, s, T, h)$.

Algorithm MIN-GLOBAL invokes Algorithm MIN-LOCAL k^2 times, hence its running time is $O(|V|k^2h^2)$.

E. Procedure SHRINK

Procedure SHRINK receives as input the coding network $\mathbb{N}^*(G^*, s, T, h)$. The procedure forms an auxiliary network $\hat{\mathbb{N}}(\hat{G}, s, T, h)$ by repeatedly contracting nodes of total degree 2. Specifically, we remove every node $v \in G^*$ that has one incoming link (u, v) and one outgoing link (v, w) and substitute links (u, v) and (v, w) by a single link (u, w). By Theorem 6, the total number of links in $\hat{\mathbb{N}}(\hat{G}, s, T, h)$ is bounded by $O(h^3k^2)$. The computational complexity of Procedure SHRINK is $O(|V|)$.

F. Algorithm Analysis

We are ready to formally prove the correctness of our algorithm for finding a feasible network code and analyze its performance.

Theorem 7: Let $\mathbb{N}(G, s, T, h)$ be an acyclic coding network. If $\mathbb{N}(G, s, T, h)$ is feasible, then there exists a deterministic algorithm that computes a network code $\mathbb{F}(\mathbb{N})$ for $\mathbb{N}$ in time $O(|E|kh + |V|k^2h^2 + h^4k^3(k+h))$. Moreover, the number of encoding nodes in $\mathbb{F}(\mathbb{N})$ is bounded by $O(h^3k^2)$.

Proof: We begin by observing that the output $\hat{\mathbb{N}}(\hat{G}, s, T, h)$ of Procedure SHRINK is a feasible coding network. Let $\hat{\mathbb{F}}(\hat{\mathbb{N}})$ be a feasible network code for $\hat{\mathbb{N}}$. The number of encoding nodes in $\hat{\mathbb{F}}$ is bounded by the number of nodes in $\hat{G}$ of in-degree two or more. Theorem 6 implies that the number of such nodes in G^* and, in turn, in $\hat{G}$ is at most $O(h^3k^2)$.

Next, we show how to construct a feasible network code $\mathbb{F}^*(\mathbb{N}^*)$ for coding network $\mathbb{N}^*(G^*, s, T, h)$ (returned by Algorithm MIN-GLOBAL). By our construction, every link e in $\hat{G}$ corresponds to a path P in G^*. In $\mathbb{F}^*$, the first link of P has the same encoding functions as e in $\hat{\mathbb{F}}$. All other links in P are emanating from nodes of in-degree one, hence such links just forward incoming packets. Clearly, if $\hat{\mathbb{F}}(\hat{\mathbb{N}})$ is a feasible network code, so is $\mathbb{F}^*(\mathbb{N}^*)$. Since G^* is a subgraph of G', $\mathbb{F}^*(\mathbb{N}^*)$ can be immediately extended into a feasible network code $\mathbb{F}'(\mathbb{N}')$ for $\mathbb{N}'(G', s, T, h)$. The number of encoding nodes in $\mathbb{F}'(\mathbb{N}')$ is bounded by $O(h^3k^2)$.

Finally, we construct a feasible network code for the original network $\mathbb{N}(G, s, T, h)$. The graph G' is constructed from G in two phases (Procedure EXPAND). First, links are removed from G and then the remaining high degree nodes v are replaced by the gadget Γ_v. Let $e = (v, u)$ be a link in G. If e was removed in Procedure EXPAND, then the encoding function $f(e)$ of e is set to be equal to the zero element of Σ. Next, if the node v in G was not replaced by a gadget, then we set the encoding function of e to be equal to that of link e' that corresponds to e in G'. Finally, we consider the case in which node v was substituted by gadget Γ_v. Let $\{x_i, v\}$ be the set of incoming links of v, $\{x_i, x_i'\}$ be the set of links in G' that correspond to $\{x_i, v\}$, and $e' = (u', u)$ be the link in G' that corresponds to (v, u). (We use the same notation as in Section III-B). A packet sent over e' is a linear function of the packets sent over links $\{x_i, x_i'\}$. We set this function to be the encoding function $f(e)$ of e. It is easy to verify that this definition yields a feasible network code for $\mathbb{N}(G, s, T, h)$. Note that the number of encoding nodes in $\mathbb{F}(\mathbb{N})$ is bounded by the number of encoding nodes in $\mathbb{F}'(\mathbb{N}')$.

We proceed to determine the computational complexity of the algorithm. Recall that the running time of Procedure EXPAND is bounded by $O(|E|kh)$. The running time of Algorithm MIN-GLOBAL is $O(|V|k^2h^2)$. Since the graph $\hat{G}$ contains $O(h^3k^2)$ links, finding a feasible network code $\hat{\mathbb{F}}(\hat{\mathbb{N}})$ for $\hat{\mathbb{N}}(\hat{G}, s, T, h)$ requires $O(h^4k^3(k+h))$ (using the algorithm of [1]). We conclude that the total running time of the algorithm is bounded by $O(|E|kh + |V|k^2h^2 + h^4k^3(k+h))$. □

For the special case of $h = 2$, the computational complexity of the algorithm can be improved by using the algorithm due to [10].

Corollary 8: Let $\mathbb{N}(G, s, T, h)$ be an acyclic coding network with links of integral capacity in which $h = 2$. If $\mathbb{N}(G, s, T, h)$ is feasible, then there exists a deterministic algorithm that computes a network code $\mathbb{F}(\mathbb{N})$ for $\mathbb{N}$ in time $O(|E| + |V|k^2 + k^4)$. Moreover, the number of encoding nodes in $\mathbb{F}(\mathbb{N})$ is bounded by $O(k^2)$.

Proof: In Procedure EXPAND, when finding $h = 2$ link disjoint paths between s and every terminal t_i we use the algorithm of [10] which performs this task in time $O(|E|)$. □

IV. Minimizing the Number of Encoding Nodes

In this section, we analyze the computational complexity of finding a feasible network code with a minimum number of encoding nodes. We consider both integral and fractional network codes. In fractional network codes, a packet can be split into a number of smaller packets, while in integral network codes a packet cannot be split.

We begin by analyzing integral network codes. We then define network coding in a fractional setting. We show that the number of encoding nodes in fractional network codes can be lower than that required by integral codes. Finally, we present our results on finding a feasible network code with a minimum number of encoding nodes for fractional codes. Our results are summarized in Fig. 1.

A. Integral Network Codes

For a given coding network $\mathbb{N}(G, s, T, h)$ we denote by $\mathrm{Opt}_1(\mathbb{N})$ the minimum number of encoding nodes in a feasible integral network code for $\mathbb{N}$.[2] We begin by stating the results due to [6] and [11].

Theorem 9 ([6], [11]): Computing $\mathrm{Opt}_1(\mathbb{N})$ is $\mathcal{NP}$-hard for general networks $\mathbb{N}(G, s, T, h)$ in which $k = h = 2$.

Next, we show that in acyclic networks the problem of finding $\mathrm{Opt}_1(\mathbb{N})$ is $\mathcal{NP}$-hard even if ether $h = 2$ or $k = 2$.

Theorem 10: In acyclic networks, the problem of computing $\mathrm{Opt}_1(\mathbb{N})$ is NP-hard even under the restriction of either $h = 2$ or $k = 2$.

Proof: We use a reduction from the *minimum set cover* (MSC) problem. The input to Problem MSC is a universe $U = (x_1, \ldots, x_a)$ of a elements and a collection $C = \{S_1, \ldots, S_r\}$ of r subsets of U. The objective of the problem is to find a subset $C' \subseteq C$ of minimum size such that every element in U belongs to at least one member of C'.

We start with the case of two terminals ($k = 2$). Given an instance of Problem MSC, we construct a coding network $\mathbb{N}(G, s, \{t_1, t_2\}, 2ra)$, as depicted in Fig. 6. The underlying communication graph G contains a node x_i for each element $x_i \in U$ and a node S_j for each $S_j \in C$. For each set $S_j \in C$ and for each element $x_i \in S_j$, there is a link (S_j, x_i) in G of unit capacity. For each element x_i there is also a link to t_2 of unit capacity. All other links in G are of capacity a, except for links (s, s^*) and (s^*, t_1) of capacity ra and link (s, t_2) of capacity $(r-1)a$. It is easy to verify that if the instance of Problem MSC is feasible then there exist two flows θ_1 and θ_2 of value $2ra$ from s to t_1 and s to t_2, respectively. Thus, $\mathbb{N}(G, s, \{t_1, t_2\}, 2ra)$ is a feasible coding network. We proceed to show that the minimum size of a set cover is equal to $\mathrm{Opt}_1(\mathbb{N})$.

First, we show that there exists a set cover C', such that $|C'| \leq \mathrm{Opt}_1(\mathbb{N})$. Let $\mathbb{F}(\mathbb{N})$ be an optimal network code (i.e., a code that has $\mathrm{Opt}_1(\mathbb{N})$ encoding nodes). This implies that there exists a subgraph G' of G that satisfies the following properties.

1) The capacity of any cut in G' that separates a source s and a terminal t_1 or t_2 is at least $2ra$.
2) The number of nodes in G' of in-degree 2 or more is bounded by $\mathrm{Opt}_1(\mathbb{N})$.

[2]The index 1 represents the fact that we are considering integral codes.

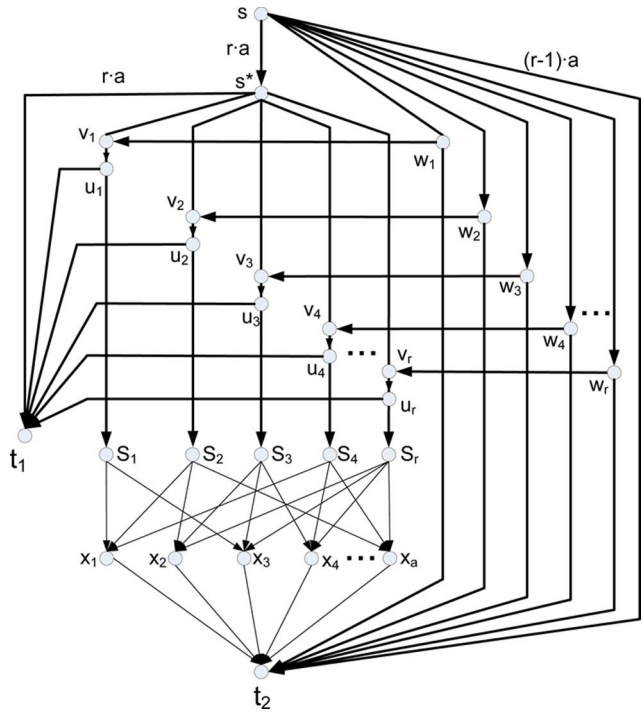

Fig. 6. Reduction from Problem MSC to the problem of finding the minimum value of $\mathrm{Opt}_1(\mathbb{N})$ for $|T| = 2$. All bold links are of capacity a, except for links (s, s^*) and (s^*, t_1) of capacity $r \cdot a$ and link (s, t_2) of capacity $(r-1)a$. All thick lines are of unit capacity.

To see this, note that nodes of in-degree 2 or more in G belong to the set $\{v_1, \ldots, v_r; x_1, \ldots, x_a\}$. At most, $\mathrm{Opt}_1(\mathbb{N})$ of these nodes are encoding nodes in $\mathbb{F}(\mathbb{N})$. For each node in $\{v_1, \ldots, v_r; x_1, \ldots, x_a\}$ which is not an encoding node in $\mathbb{F}(\mathbb{N})$ we can delete all its incoming links but one without violating the minimum cut condition.

We note that for any flow θ_1 in G from s to t_1 of size $2ra$ it must be the case that $\theta_1((w_j, v_j)) = a$ for each link (w_j, v_j), $1 \leq j \leq r$. Thus, since G' satisfies the minimum cut condition, it must include all links (w_j, v_j), $1 \leq j \leq r$. This implies, in turn, that G' includes at most $\mathrm{Opt}_1(\mathbb{N})$ links from the set $\{(s^*, v_j) | 1 \leq j \leq r\}$. We denote by I the set $\{j | (s^*, v_j) \in G'\}$. Again, since G' satisfies the minimum cut condition, there exists a flow θ_2 from s to t_2 of size $2ra$. Such a flow must satisfy $\theta_2((x_i, t_2)) = 1$ for all links (x_i, t_2), $1 \leq i \leq a$, and $\theta_2((w_j, t_2)) = a$ for all links (w_j, t_2), $1 \leq j \leq r$. This implies that for each x_i, $1 \leq i \leq a$ there exists a node S_j, such that $j \in I$ and $(S_j, x_i) \in G'$. We conclude that the set $C' = \{S_j | j \in I\}$ is a valid set cover that satisfies $|C'| \leq \mathrm{Opt}_1(\mathbb{N})$.

Let C' be any set cover. We show how to construct an integral network code $\mathbb{F}$ for $\mathbb{N}(G, s, \{t_1, t_2\}, 2ra)$ with at most $|C'|$ encoding nodes. To that end, we construct a subgraph G' of G as above that satisfies the following conditions. 1) The capacity of any cut in G' that separates a source s and a terminal t_1 or t_2 is at least $2ra$. 2) The number of nodes in G' of in-degree 2, except for t_1 and t_2, is equal to $|C'|$. Given such a G', a network code with at most $|C'|$ encoding nodes can be constructed, by invoking the algorithm due to [1] on $\mathbb{N}(G', s, \{t_1, t_2\}, 2ra)$. G' is formed from G by deleting all nodes S_j that do not correspond to the elements of the set cover. For such indices j we

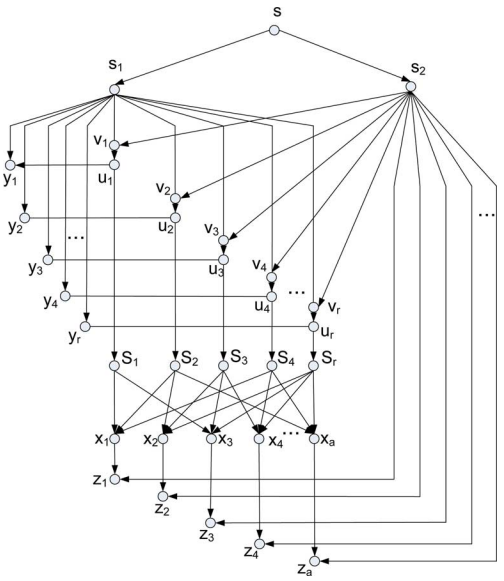

Fig. 7. Reduction from Problem MSC to the problem of finding the minimum value of $\mathrm{Opt}_1(\mathbb{N})$ for $h = 2$. All links in the network are of unit capacity.

also remove links leaving S_j, links entering S_j, and the link (s^*, v_j). Finally, for each x_i, we remove all but one incoming link. It is easy to verify that such a subgraph satisfies both conditions above. This implies that there exists a network code for $\mathbb{N}'(G', s, \{t_1, t_2\}, 2ra)$ (and hence $\mathbb{N}$) with at most $|C'|$ encoding nodes.

We proceed with the case of $h = 2$. Given an instance of Problem MSC, we construct a coding network $\mathbb{N}(G, s, T, 2)$, as depicted in Fig. 7. The set T of terminals includes nodes $\{y_1, \ldots, y_r\}$ and $\{z_1, \ldots, z_a\}$. For each set $S_j \in C$ and for each element $x_i \in S_j$, there is a link (S_j, x_i) in G. All links in the network are of unit capacity. Using arguments similar to those used for the case of $k = 2$ it can be shown that the feasibility of Problem MSC implies the feasibility of $\mathbb{N}(G, s, T, 2)$ and that the size of the optimal set cover is equal to $\mathrm{Opt}_1(\mathbb{N})$. □

Theorem 11: Let $\mathbb{N}(G, s, T, h)$ be a feasible acyclic network. Then, a feasible integral network code with $\mathrm{Opt}_1(\mathbb{N})$ encoding nodes can be constructed in time $|V|^{O(h^6k^4)}$.

Proof: (Sketch) We show how to construct a feasible network code in time $|V|^{O(h^6k^4)}$. Our approach can be summarized as follows. First, we define a graph $\hat{P}(\hat{V}, \hat{E})$ of small size that captures the structure of the optimal solution and analyze its properties. Next, we show that if the graph $\hat{P}$ is known, then we can find a feasible network code with $\mathrm{Opt}_1(\mathbb{N})$ encoding nodes. Finally, we present an algorithm that finds $\hat{P}$ or a graph similar to it (i.e., a graph that has the same properties as $\hat{P}$) through an exhaustive search.

We assume, without loss of generality, that all links in the graph G are of unit capacity, that the source s has no incoming links and exactly h outgoing links, and that each terminal $t \in T$ has no outgoing links. Let $\hat{\mathbb{F}}(\mathbb{N})$ be an optimal integral network code for $\mathbb{N}(G, s, T, h)$. Let $\hat{V}^1 \subseteq V$ be the set of encoding nodes in $\hat{\mathbb{F}}(\mathbb{N})$. Note that the packets generated by nodes in $\{s\} \cup \hat{V}^1$ are forwarded to other encoding nodes in $\hat{V}^1$ and to the terminals T via a set of $l = h + |\hat{V}^1|$ link disjoint trees $Y_1, \ldots, Y_l$ (h such trees rooted at the source and the remaining trees rooted at a node in $\hat{V}_1$). For each tree $Y_i, 1 \le i \le l$ we denote by $V(Y_i)$ the set of Steiner nodes in Y_i, i.e., the set of nodes that forward packets received over an incoming link to two or more outgoing links (the incoming and outgoing links belong to Y_i). We also define $\hat{V}^2$ to be $\cup_{1 \le i \le l} V(Y_i)$.

The structure of the optimal solution described above can be captured by the following graph $\hat{P}$. We define the graph $\hat{P}(\hat{V}, \hat{E})$ iteratively *assuming* that the optimal solution $\hat{\mathbb{F}}(\mathbb{N})$ is known. Clearly, we do not know $\hat{\mathbb{F}}$ (as this is our goal). However, the definition of $\hat{P}$ will help us in finding it (or an equivalent alternative).

- First, we add to $\hat{V}$ the source node s, all terminals $t \in T$, and all nodes $v \in \hat{V}^1$
- Next, for each tree $Y_i, 1 \le i \le l$, we perform the following operations.
  1) For each node $v \in V(Y_i)$, add a new node v' to $\hat{V}$.
  2) Let r_i be the root of Y_i. As Y_i is a tree, note that it can be decomposed into a set of paths in which each path connects between nodes in $\{r_i\} \cup V(Y_i) \cup \{\text{the set of leaves in } Y_i\}$. We denote the set of such paths by $\mathbb{P}_i$.
  3) For each path $\{v_j, \ldots, u_j\}$ in $\mathbb{P}_i$, we add a link (v'_j, u'_j) to $\hat{E}$, where v'_j and u'_j are nodes that correspond to v_j and u_j in $\hat{V}$.

Note that for a node $v \in \hat{V}^2$ there might be more than one corresponding node in $\hat{P}(\hat{V}, \hat{E})$ (this happens when v is a Steiner node in more than one tree in $\{Y_i | 1 \le i \le l\}$). Moreover, each node in $\hat{V}^2$ has a single incoming link (this will later imply that they cannot be encoding nodes in any network code for $\hat{P}$).

We now discuss some properties of $\hat{P}$. First of all, in [6] it was shown that the number of encoding nodes $\mathrm{Opt}_1(\mathbb{N}) = |\hat{V}_1|$ in $\hat{\mathbb{F}}(\mathbb{N})$ is bounded by h^3k^2. Thus, the total number of links (and thus also the size of $\hat{V}^2$) in $\hat{P}$ is bounded by $(h^3k^2 + k)(h^3k^2 + h) = O(h^6k^4)$. This follows by the fact that each tree Y_i has at most $|T| + |\hat{V}^1|$ leaves.

Second, the graph $\hat{P}(\hat{V}, \hat{E})$ captures several properties of the optimal solution. In particular, it is not hard to verify that one may easily construct a feasible network code for $\hat{P}$ with $\mathrm{Opt}_1(\mathbb{N})$ encoding nodes (these are exactly the nodes in $\hat{V}^1$).

Finally, a feasible network code with $|V_1| = \mathrm{Opt}_1(\mathbb{N})$ encoding nodes for $\hat{P}$ implies one for our original network $\mathbb{N}$. Specifically, this can be done by solving the *directed subgraph homeomorphism problem* [12] on G (the original underlying graph) and $\hat{P}$. The homeomorphism maps nodes of $\hat{P}$ to nodes in G and links of $\hat{P}$ to simple paths in G, such that the paths in G corresponding to links in $\hat{P}$ are pairwise link disjoint.[3] It is easy to verify that in directed acyclic graphs this problem can be solved in time $O(|V|^{2 \cdot |\hat{E}|} |V|^{|\hat{V}|}) = |V|^{O(h^6k^4)}$.

It is left to find $\hat{P}$. Finding $\hat{P}$ (or a suitable alternative) is done exhaustively. For each $i, 0 \le i \le h^3k^2$ we consider all graphs P that satisfy the following two conditions.

1) The set of nodes of P is formed by s, T, and two sets V^1 and V^2, such that $|V^1| = i$ and $|V^2| \le (h^3k^2 + k)(h^3k^2 + h)$.

[3]Reference [12] considers a version of the directed homeomorphism problem in which the paths in G corresponding to links in $\hat{P}$ are pairwise *node* disjoint. However, the algorithm presented in [12] can be adapted to the link-disjoint version of the homeomorphism problem with only minor changes.

2) The number of links in P is at most $(h^3k^2+k)(h^3k^2+h)$.
3) All nodes in V^2 have a single incoming link.

For each such P, we check whether there exists h link-disjoint paths between s and every $t \in T$. If this holds, we solve the subgraph homomorphism problem for P. If the subgraph homomorphism problem has a feasible solution, we determine a network code for P (e.g., by using the algorithm due to [1]) and map the obtained network code to the subgraph G' of G which is a solution of the homomorphism problem for P. Notice, as nodes in V^2 have a single incoming link, all encoding nodes in P must belong to V^1.

The algorithm outputs the smallest value of i for which the subgraph homomorphism problem has a feasible solution and the corresponding network code associated with it.

We proceed to analyze the computational complexity of our algorithm. For each value of $i \leq h^3k^2$, there are $2^{O(h^6k^4)}$ possible ways to construct a graph P (one for each subset of possible $O(h^6k^4)$ links). For each such graph, the complexity of our algorithm is dominated by the solution of the subgraph homomorphism problem, which can be done in time $|V|O(h^6k^4)$. We conclude that the total complexity of our algorithm is $|V|^{O(h^6k^4)}$. □

B. Fractional Network Codes

In this subsection, we define a notion of an m-fractional network code for a coding network $\mathbb{N}(G,s,T,h)$. Recall that a capacity c_e of a link e was defined in Section II as the number of packets that can be transmitted by e in one time unit. For fractional network codes, it is more convenient to use the notion of *bit capacity* c'_e of link e, defined as the maximum number of bits that can be transmitted by link e in one time unit. The bit capacity of the network can be defined in a similar manner. We assume that all integral (original) packets are elements in $\Sigma = \mathrm{GF}(2^n)$, which implies that each such packet can be represented by n bits. Thus, for each link $e \in E$ it holds that $c'_e = c_e n$.

Assume that m is a divisor of n. Then, in an m-fractional network code, each packet is an element of the finite field $\Sigma' = \mathrm{GF}(2^{n/m})$ and can be represented by n/m bits. Thus, a link e of bit capacity c'_e can transmit $c'_e m/n = c_e m$ fractional packets.

Definition 12 (m-Fractional Network Code $\mathbb{F}_m(\mathbb{N})$): Let $\mathbb{N}(G,s,T,h)$ be a feasible coding network, n be the size of the integral packet, and m be a divisor of n. Also, let $\mathbb{N}_m(G_m,s,T,mh)$ be the coding network in which G_m is formed from G by splitting each link e in G of bit capacity c'_e into m parallel links $e_1,\ldots,e_m$ of bit capacity c'_e/m. Finally, let $\mathbb{F}(\mathbb{N}_m)$ be a feasible integral network code for $\mathbb{N}_m(G_m,s,T,mh)$ over $\Sigma' = GF(2^{n/m})$. We refer to $\mathbb{F}(\mathbb{N}_m)$ as a feasible m-fractional network code $\mathbb{F}_m(\mathbb{N})$ for $\mathbb{N}(G,s,T,h)$.

For example, in a 2-fractional network code each packet is split into two packets of length $n/2$ bits and each link $e \in E$ is split into two parallel links of bit capacity $c'_e/2$

Note that a 1-fractional network code for $\mathbb{N}$ is an integral network code. Note also that coding networks $\mathbb{N}$ and $\mathbb{N}_m$ have the same (bit) capacity. Thus, for any m, the maximum (bit) rate achieved by an m-fractional code $\mathbb{F}_m(\mathbb{N})$ is equal to that achievable by an integral network code $\mathbb{F}(\mathbb{N})$. However, as we show in Section IV-C below, a fractional code may require a smaller number of encoding nodes.

For a given coding network $\mathbb{N}(G,s,T,h)$ and a divisor m of n, we denote by $\mathrm{Opt}_m(\mathbb{N})$ the minimum number of encoding nodes in a feasible m-fractional network code for $\mathbb{N}$. We then define

$$\mathrm{Opt}(\mathbb{N}) = \min_m \mathrm{Opt}_m(\mathbb{N}) \tag{2}$$

where the maximum is taken over all m that divide n.

We will use the following theorem which appears in a slightly modified form in [8].

Theorem 13: Let $\mathbb{N}(G(V,E),s,T,h)$ be a coding network. Let V_1 and V_2 be a partition of V. Then, there exists a linear program with $O(|E|2^k)$ variables, $O(|E|)$ constraints, and coefficients in $\{-1,0,1\}$ which is feasible if and only if there exists, for some integer $m>0$, an m-fractional network code for $\mathbb{N}(G(V,E),s,T,h)$, in which only nodes in V_2 are encoding nodes. Further, such a code can be obtained in polynomial time from the solution of the linear program (e.g., by using the algorithm presented in [1]).

C. Integral Versus Fractional Network Codes

In this subsection, we show that the number of encoding nodes in fractional network codes can be substantially lower than that in integral codes.

Lemma 14: Let r be any positive integer. There exist a coding network $\mathbb{N}(G,s,T,h)$ such that $\mathrm{Opt}(\mathbb{N})=0$ while $\mathrm{Opt}_1(\mathbb{N})=r$.

Proof: Consider the coding network $\mathbb{N}(G,s,\{t_1,t_2\},2)$ depicted in Fig. 8. In this network, all links are of bit capacity n (i.e., they can transmit one integer packet per time unit). Fig. 8(a) depicts an integral network code for $\mathbb{N}$ with one encoding node. It is not hard to verify that any feasible integral network code for $\mathbb{N}(G,s,\{t_1,t_2\},2)$ requires at least one encoding node, hence $\mathrm{Opt}_1(\mathbb{N}) = 1$. Fig. 8(b) depicts a 2-fractional network code for $\mathbb{N}$ with no encoding nodes. The code splits the original packets a and b into two packets of smaller size a_1 and a_2, and b_1, and b_2, respectively.

By using $\mathbb{N}(G,s,\{t_1,t_2\},2)$ as a building block, we can construct a coding network $\mathbb{N}^*(G^*,s,T,h)$ for which it holds that $\mathrm{Opt}(\mathbb{N}) = 0$ while $\mathrm{Opt}_1(\mathbb{N}) = r$. The communication graph G^* of $\mathbb{N}^*$ consists of r copies of G, each copy connected to the same source node and a different set of terminals. □

Remark 15: In the coding network $\mathbb{N}^*(G^*,s,T,h)$ constructed in the Proof of Lemma 14, the number of terminals T depends on r. Alternatively, we can prove the lemma by constructing a coding network in which the number of terminals is fixed, but the number of packets to be transmitted depends on r. In addition, the lemma can be proven by constructing a cyclic network with a fixed number of packets and terminals.

D. Our Results for Fractional Codes

Theorem 16: Let $\mathbb{N}(G,s,T,h)$ be a feasible acyclic network. Then, a feasible m-fractional network code $\mathbb{F}(\mathbb{N})$ for $\mathbb{N}$ with $\mathrm{Opt}(\mathbb{N})$ encoding nodes can be constructed in time $|V|^{O(h^3k^2)}$.

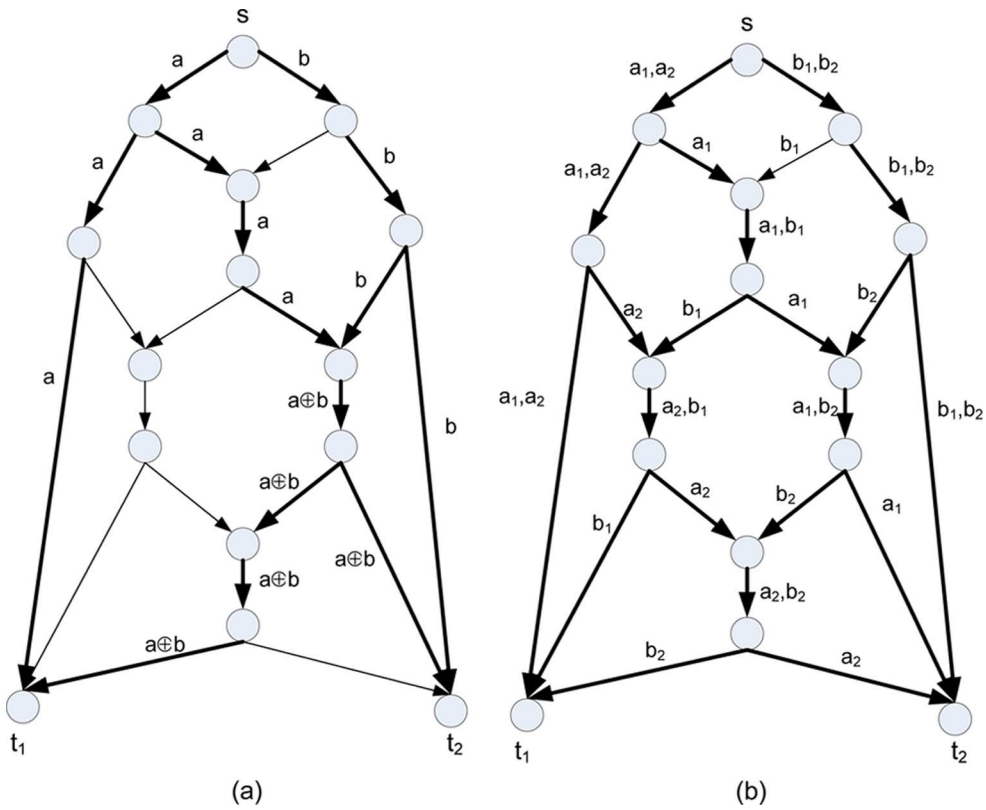

Fig. 8. Coding network $\mathbb{N}(G, s, \{t_1, t_2\}, 2)$ with source node s and two terminals, t_1 and t_2. All links are of unit capacity (bit capacity n). (a) A integral network for $\mathbb{N}$. Each integral network code requires at least one encoding node. (b) A fractional network code for $\mathbb{N}$ that does not require encoding nodes. The code splits packets a and b into packets a_1 and a_2, and b_1 and b_2, respectively.

Proof: (Sketch) In [6] it was shown that $\mathbb{N}$ has an integral network code with at most h^3k^2 encoding nodes. Thus, $\mathrm{Opt}(\mathbb{N}) \leq h^3k^2$. This implies the following procedure for constructing $\mathbb{F}(\mathbb{N})$. For all subsets of nodes V' in G of size at most h^3k^2, we construct and solve a linear program of Theorem 13 in which $V_1 = V \setminus V'$ and $V_2 = V'$. We return the feasible network code corresponding to the smallest set V'. The existence of a feasible network code follows from Theorem 13. Solving a linear program with n constrains and m variables requires $O((mn)^2L)$ time [13], where L is the total number of bits in the input. Thus, processing of each subset V' requires $|V|^{O(1)}2^{O(k)}$ time, hence, the total complexity is bounded by $|V|^{O(h^3k^2)}$. □

Theorem 17: Let $\mathbb{N}(G, s, T, h)$ be an acyclic network for which it holds that either $k = 2$ or $h = 2$. Then, computing $\mathrm{Opt}(\mathbb{N})$ is $\mathcal{NP}$-hard.

Proof: (Sketch) The proof follows the lines of the proof of Theorem 10. Namely, we use the same constructions with only minor and straightforward modifications. □

V. Conclusion

In this work, we considered the computational aspects of multicast network coding. Our goal was both to minimize the amount of computation performed by network nodes and to reduce the computational complexity of finding efficient network codes. In particular, we studied algorithms for finding network codes with a bounded number of encoding nodes.

Our results can be summarized as follows. First, we present a deterministic algorithm that constructs a feasible network code for acyclic networks in which the number of encoding nodes is bounded by $O(h^3k^2)$. To the best of our knowledge, this is the first efficient algorithm that provides a bound on the number of encoding nodes. The computational complexity of our algorithm is $O(|E|kh + |V|k^2h^2 + h^4k^3(k + h))$, where k is the number of destinations and h is the number of packets. Our result improves the best known running time of $O(|E|kh + |V|k^2h^2(k + h))$ of the algorithm due to Jaggi *et al.* [1] in the typical case of large communication graphs.

Second, we address the problem of finding a network code with a minimum number of encoding nodes in both integral and fractional coding networks. We prove that in the majority of settings this problem is $\mathcal{NP}$-hard. However, we show that if $h = O(1), k = O(1)$, and the underlying communication graph is acyclic, then there exists an algorithm that solves this problem in polynomial time.

The problem of finding a fractional network code with a minimum number of encoding nodes in general (cyclic) networks for a constant numbers of k and h was not resolved in this work. In the fractional setting, constant values of h imply a constant ratio between the minimum link capacity and the total number of fractional packets. This problem would be an interesting topic for future research.

References

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

[2] 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.
[3] 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.
[4] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, Oct. 2003.
[5] 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*, Lausanne, Switzerland, Jun./Jul. 2003.
[6] M. Langberg, A. Sprintson, and J. Bruck, "The encoding complexity of network coding," *IEEE Trans.Inf. Theory*, vol. 52, no. 6, pp. 2386–2397, Jun. 2006.
[7] N. J. A. Harvey, D. R. Karger, and K. Murota, "Deterministic network coding by matrix completion," in *Proc. ACM-SIAM Symp. Discrete Algorithms*, Vancouver, BC, Canada, Jan. 2005, pp. 489–498.
[8] K. Bhattad, N. Ratnakar, R. Koetter, and K. R. Narayanan, "Minimal network coding for multicast," in *Proc. IEEE Int. Symp. Information Theory*, Adelaide, Australia, Sep. 2005, pp. 1730–1734.
[9] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Networks Flows*. Upper Saddle River, NJ: Prentice-Hall, 1993.
[10] J. Suurballe and R. Tarjan, "A quick method for finding shortest pairs of disjoint paths," *Networks*, vol. 14, pp. 325–336, 1984.
[11] M. Mahdian and M. R. Salavatipour, "Hardness and approximation results for packing Steiner treesproblems ," in *Lecture Notes in Computer Science*. Berlin, Germany: Springer-Verlag, 2004, vol. 3221, pp. 181–191.
[12] S. Fortune, J. Hopcroft, and J. Wyllie, "The directed subgraph homeomorphism problem," *Theor. Comp. Sci.*, vol. 10, no. 2, pp. 111–121, 1980.
[13] P. M. Vaidya, "An algorithm for linear programming which requires $O(((m+n)n^2+(m+n)^{1.5}n)L)$ arithmetic operations," *Math. Program.*, vol. 47, no. 2, pp. 175–201, Jun. 1990.

Michael Langberg (M'05) received the B.Sc. degree in mathematics and computer science from Tel-Aviv University, Tel-Aviv, Israel, in 1996 and the M.Sc. and Ph.D. degrees in computer science from the Weizmann Institute of Science, Rehovot, Israel, in 1998 and 2003, respectively.

He was a Postdoctoral Scholar in the Computer Science and Electrical Engineering Departments at the California Institute of Technology, Pasadena, CA, between 2003 and 2006. He is now a faculty member in the Computer Science Department at the Open University of Israel, Raanan. His research is in the fields of theoretical computer science and information theory. His work focuses on the design and analysis of algorithms for combinatorial problems; with emphasis on geometrically oriented approximation algorithms for NP-hard problems, on algorithmic and combinatorial aspects of information theory, and on probabilistic methods in combinatorics.

Alexader Sprintson (S'00–M'03) received the B.Sc. (*summa cum laude*), M.Sc., and Ph.D. degrees in electrical engineering from the Technion–Israel Institute of Technology, Haifa, Israel, in 1995, 2001, and 2003, respectively.

From 2003 to 2005, he was a Postdoctoral Research fellow at the California Institute of Technology, Pasadena. During the summers of 2002 and 2003 he was with the Internet Management Research Department at Bell Laboratories, Murray Hill, NJ. He is now an Assistant Professor of Electrical and Computer Engineering at Texas A&M University, College Station. His research interests lie in the general area of communication networks with a focus on network coding, network survivability and robustness, and QoS routing.

Dr. Sprintson's current honors include having received the Wolf Award for distinguished Ph.D. students and the Viterbi Postdoctoral Fellowship. He is an Associate Editor for IEEE COMMUNICATIONS LETTERS and is a Member of the Technical Program Committee for IEEE Infocom 2006–2009.

Jehoshua Bruck (S'86–M'89–SM'93–F'01) received the B.Sc. and M.Sc. degrees in electrical engineering from the Technion–Israel Institute of Technology, Haifa, in 1982 and 1985, respectively, and the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA, in 1989.

He is the Gordon and Betty Moore Professor of Computation and Neural Systems and Electrical Engineering at the California Institute of Technology (Caltech), Pasadena. He served as the Founding Director of the Caltech Information Science and Technology (IST) program. His research focuses on the theory and practice of information communications and storage systems, and the analysis and design of biological circuits and systems. He has extensive industrial experience, including working with IBM Research where he participated in the design and implementation of the first IBM parallel computer. He was a cofounder and Chairman of Rainfinity, a spinoff company from Caltech that was the first to create a file virtualization solution for enterprise storage.

Prof. Bruck's awards include the National Science Foundation Young Investigator award and the Sloan fellowship. His papers were recognized in journals and conferences, including, winning the 2005 S. A. Schelkunoff Transactions prize paper award from the IEEE Antennas and Propagation society and the Best Paper Award in the 2003 Design Automation Conference.