Network Coding for Computing: Cut-Set Bounds

Rathinakumar Appuswamy, *Student Member, IEEE*, Massimo Franceschetti, *Member, IEEE*, Nikhil Karamchandani, *Student Member, IEEE*, and Kenneth Zeger, *Fellow, IEEE*

***Abstract*—The following *network computing* problem is considered. Source nodes in a directed acyclic network generate independent messages and a single receiver node computes a target function f of the messages. The objective is to maximize the average number of times f can be computed per network usage, i.e., the "computing capacity". The *network coding* problem for a single-receiver network is a special case of the network computing problem in which all of the source messages must be reproduced at the receiver. For network coding with a single receiver, routing is known to achieve the capacity by achieving the network min-cut upper bound. We extend the definition of min-cut to the network computing problem and show that the min-cut is still an upper bound on the maximum achievable rate and is tight for computing (using coding) any target function in multi-edge tree networks. It is also tight for computing linear target functions in any network. We also study the bound's tightness for different classes of target functions. In particular, we give a lower bound on the computing capacity in terms of the Steiner tree packing number and a different bound for symmetric functions. We also show that for certain networks and target functions, the computing capacity can be less than an arbitrarily small fraction of the min-cut bound.**

***Index Terms*—Capacity, cut-set bound, function computation, information theory, network coding, throughput.**

I. Introduction

We consider networks where source nodes generate independent messages and a single receiver node computes a target function f of these messages. The objective is to characterize the maximum rate of computation, that is, the maximum number of times f can be computed per network usage.

Giridhar and Kumar [18] have stated:

"In its most general form, computing a function in a network involves communicating possibly correlated messages, to a specific destination, at a desired fidelity with respect to a joint distortion criterion dependent on the given function of interest. This combines the complexity of source coding of correlated sources, with rate distortion, different possible network collaborative strategies for computing and communication, and the inapplicability of the separation theorem demarcating source and channel coding."

The overwhelming complexity of network computing suggests that simplifications be examined in order to obtain some understanding of the field.

We present a natural model of network computing that is closely related to the network coding model of Ahlswede, Cai, Li, and Yeung [1], [49]. Network coding is a widely studied communication mechanism in the context of network information theory. In network coding, some nodes in the network are labeled as sources and some as receivers. Each receiver needs to reproduce a subset of the messages generated by the source nodes, and all nodes can act as relays and encode the information they receive on in-edges, together with the information they generate if they are sources, into codewords which are sent on their out-edges. In existing computer networks, the encoding operations are purely routing: at each node, the codeword sent over an out-edge consists of a symbol either received by the node, or generated by it if it is a source. It is known that allowing more complex encoding than routing can in general be advantageous in terms of communication rate [1], [22], [38]. Network coding with a single receiver is equivalent to a special case of our function computing problem, namely when the function to be computed is the identity, that is, when the receiver wants to reproduce all the messages generated by the sources. In this paper, we study network computation for target functions different than the identity.

Some other approaches to network computation have also appeared in the literature. In [8], [11], [12], [28], [34], [39] network computing was considered as an extension of distributed source coding, allowing the sources to have a joint distribution and requiring that a function be computed with small error probability. For example, [28] considered a network where two correlated uniform binary sources are both connected to the receiver and then determined the maximum rate of computing the parity of the messages generated by the two sources. A rate-distortion approach to the problem has been studied in [10], [15], [47]. However, the complexity of network computing has restricted prior work to the analysis of elementary networks. Networks with noisy links were studied in [3], [14], [16], [17], [19], [26], [35], [37], and [50]. For example, [17] considered broadcast networks where any transmission by a node is received by each of its neighbors via an independent binary symmetric channel. Randomized gossip algorithms [6] have been proposed as practical schemes for information dissemination in large unreliable networks and were studied in the context of distributed computation in [4]–[6], [9], [27], and [36].

In the present paper, our approach is somewhat (tangentially) related to the field of communication complexity [30], [48] which studies the minimum number of messages that two nodes need to exchange in order to compute a function of their inputs with zero error. Other studies of computing in networks have been considered in [18], and [43], but these were restricted

Manuscript received April 18, 2010; revised August 10, 2010; accepted September 11, 2010. Date of current version January 19, 2011. This work was supported by the National Science Foundation and the UCSD Center for Wireless Communications.

This paper is part of the special issue on "Facets of Coding Theory: From Algorithms to Networks," dedicated to the scientific legacy of Ralf Koetter.

The authors are with the Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093-0407 (e-mail: rathnam@ucsd.edu; massimo@ece.ucsd.edu; nikhil@ucsd.edu; zeger@ucsd.edu).

Communicated by M. Effros, Associate Editor for the special issue on "Facets of Coding Theory: From Algorithms to Networks."

Digital Object Identifier 10.1109/TIT.2010.2095070

TABLE I
EXAMPLES OF TARGET FUNCTIONS

Target function f	Alphabet $\mathcal{A}$	$f(x_1, \ldots, x_s)$	Comments
identity	arbitrary	$(x_1, \ldots, x_s)$	$\mathcal{B} = \mathcal{A}^s$
arithmetic sum	$\{0, 1, \ldots, q-1\}$	$x_1 + x_2 + \cdots + x_s$	'+' is ordinary integer addition, $\mathcal{B} = \{0, 1, \cdots, s(q-1)\}$
mod r sum	$\{0, 1, \ldots, q-1\}$	$x_1 \oplus x_2 \oplus \ldots \oplus x_s$	$\oplus$ is mod r addition, $\mathcal{B} = \mathcal{A}$
histogram	$\{0, 1, \ldots, q-1\}$	$(c_0, c_1, \ldots, c_{q-1})$	$c_i = \|\{j : x_j = i\}\|$ for each $i \in \mathcal{A}$
linear	any finite field	$a_1x_1 + a_2x_2 + \ldots + a_sx_s$	arithmetic in the field, $\mathcal{B} = \mathcal{A}$
maximum	any ordered set	$\max\{x_1, \ldots, x_s\}$	$\mathcal{B} = \mathcal{A}$

to the wireless communication protocol model of Gupta and Kumar [20].

In contrast, our approach is more closely associated with wired networks with independent noiseless links. Our work is closest in spirit to the recent work of [31], [40]–[42] on computing the sum (over a finite field) of source messages in networks. We note that in independent work, Kowshik and Kumar [29] obtain the asymptotic maximum rate of computation in tree networks and present bounds for computation in networks where all nodes are sources.

Our main contributions are summarized in Section I-C, after formally introducing the network model.

A. Network Model and Definitions

In this paper, a *network* $\mathcal{N}$ consists of a finite, directed acyclic multigraph $G = (\mathcal{V}, \mathcal{E})$, a set of *source nodes* $S = \{\sigma_1, \ldots, \sigma_s\} \subseteq \mathcal{V}$, and a *receiver* $\rho \in \mathcal{V}$. Such a network is denoted by $\mathcal{N} = (G, S, \rho)$. We will assume that $\rho \notin S$ and that the graph[1] G contains a directed path from every node in $\mathcal{V}$ to the receiver ρ. For each node $u \in \mathcal{V}$, let $\mathcal{E}_i(u)$ and $\mathcal{E}_o(u)$ denote the set of in-edges and out-edges of u respectively. We will also assume (without loss of generality) that if a network node has no in-edges, then it is a source node.

An *alphabet* $\mathcal{A}$ is a finite set of size at least two. For any positive integer m, any vector $x \in \mathcal{A}^m$, and any $i \in \{1, 2, \ldots, m\}$, let x_i denote the i-th component of x. For any index set $I = \{i_1, i_2, \ldots, i_q\} \subseteq \{1, 2, \ldots, m\}$ with $i_1 < i_2 < \cdots < i_q$, let x_I denote the vector $(x_{i_1}, x_{i_2}, \ldots, x_{i_q}) \in \mathcal{A}^{|I|}$.

The *network computing* problem consists of a network $\mathcal{N}$ and a *target function* f of the form

$$f : \mathcal{A}^s \longrightarrow \mathcal{B}$$

(see Table I for some examples). We will also assume that any target function depends on all network sources (i.e., they cannot be constant functions of any one of their arguments). Let k and n be positive integers. Given a network $\mathcal{N}$ with source set S and alphabet $\mathcal{A}$, a *message generator* is any mapping

$$\alpha : S \longrightarrow \mathcal{A}^k.$$

For each source σ_i, $\alpha(\sigma_i)$ is called a *message vector* and its components $\alpha(\sigma_i)_1, \ldots, \alpha(\sigma_i)_k$ are called *messages*.[2]

[1]Throughout the paper, we will use "graph" to mean a directed acyclic multigraph, and "network" to mean a single-receiver network. We may sometimes write $\mathcal{E}(G)$ to denote the edges of graph G.

[2]For simplicity, we assume that each source has exactly one message vector associated with it, but all of the results in this paper can readily be extended to the more general case.

Definition I.1: A (k, n) *network code* in a network $\mathcal{N}$ consists of the following.

(i) For any node $v \in \mathcal{V} - \rho$ and any out-edge $e \in \mathcal{E}_o(v)$, an *encoding function*

$$h^{(e)} : \begin{cases} \left(\prod_{\hat{e} \in \mathcal{E}_i(v)} \mathcal{A}^n \right) \times \mathcal{A}^k \longrightarrow \mathcal{A}^n & \text{if } v \text{ is a source node} \\ \prod_{\hat{e} \in \mathcal{E}_i(v)} \mathcal{A}^n \longrightarrow \mathcal{A}^n & \text{otherwise.} \end{cases}$$

(ii) A *decoding function*:

$$\psi : \prod_{\hat{e} \in \mathcal{E}_i(v)} \mathcal{A}^n \longrightarrow \mathcal{B}^k.$$

Given a (k, n) network code, every edge $e \in \mathcal{E}$ carries a vector z_e of at most n alphabet symbols[3], which is obtained by evaluating the encoding function $h^{(e)}$ on the set of vectors carried by the in-edges to the node and the node's message vector if it is a source. The objective of the receiver is to compute the target function f of the source messages, for any arbitrary message generator α. More precisely, the receiver constructs a vector of k alphabet symbols such that for each $i \in \{1, 2, \ldots, k\}$, the i-th component of the receiver's computed vector equals the value of the desired target function f applied to the i-th components of the source message vectors, for any choice of message generator α. Let $e_1, e_2, \ldots, e_{|\mathcal{E}_i(\rho)|}$ denote the in-edges of the receiver.

Definition I.2: A (k, n) network code is said to compute f in $\mathcal{N}$ if the decoding function ψ is such that for each $j \in \{1, 2, \ldots, k\}$ and for every message generator α

$$\psi(z_{e_1}, \ldots, z_{e_{|\mathcal{E}_i(\rho)|}})_j = f(\alpha(\sigma_1)_j, \ldots, \alpha(\sigma_s)_j). \quad (1)$$

If there exists a (k, n) code that computes f, we say the rational number k/n is an *achievable computing rate*.

In the network coding literature, one definition of the *coding capacity* of a network is the supremum of all achievable coding rates [7], [13]. We adopt an analogous definition for computing capacity.

Definition I.3: The *computing capacity* of a network $\mathcal{N}$ with respect to target function f is

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) = \sup\left\{ \frac{k}{n} : \exists (k, n) \text{ network code that computes } f \text{ in } \mathcal{N} \right\}.$$

[3]By default, we will assume that edges carry exactly n symbols.

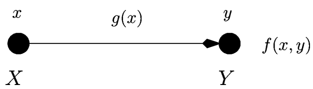

Fig. 1. X, Y are two sources with messages x and y respectively. X communicates $g(x)$ to Y so that Y can compute a function f of x and y.

Thus, the computing capacity is the supremum of all achievable computing rates for a given network $\mathcal{N}$ and a target function f.

Definition I.4: For any target function $f : \mathcal{A}^s \longrightarrow \mathcal{B}$, any index set $I \subseteq \{1, 2, \ldots, s\}$, and any $a, b \in \mathcal{A}^{|I|}$, we write $a \equiv b$ if for every $x, y \in \mathcal{A}^s$, we have $f(x) = f(y)$ whenever $x_I = a$, $y_I = b$, and $x_j = y_j$ for all $j \notin I$.

It can be verified that $\equiv$ is an equivalence relation[4] for every f and I.

Definition I.5: For every f and I, let $R_{I,f}$ denote the total number of equivalence classes induced by $\equiv$ and let

$$\Phi_{I,f} : \mathcal{A}^{|I|} \longrightarrow \{1, 2, \ldots, R_{I,f}\}$$

be any function such that $\Phi_{I,f}(a) = \Phi_{I,f}(b)$ iff $a \equiv b$.

That is, $\Phi_{I,f}$ assigns a unique index to each equivalence class, and

$$R_{I,f} = \left| \left\{ \Phi_{I,f}(a) : a \in \mathcal{A}^{|I|} \right\} \right|.$$

The value of $R_{I,f}$ is independent of the choice of $\Phi_{I,f}$. We call $R_{I,f}$ the *footprint size* of f with respect to I.

Remark I.6: Let $I^c = \{1, 2, \ldots, s\} - I$. The footprint size $R_{I,f}$ has the following interpretation (see Fig. 1). Suppose a network has two nodes, X and Y, and both are sources. A single directed edge connects X to Y. Let X generate $x \in \mathcal{A}^{|I|}$ and Y generate $y \in \mathcal{A}^{|I^c|}$. X communicates a function $g(x)$ of its input, to Y so that Y can compute $f(a)$ where $a \in \mathcal{A}^s$, $a_I = x$, and $a_{I^c} = y$. Then for any $x, \hat{x} \in \mathcal{A}^{|I|}$ such that $x \not\equiv \hat{x}$, we need $g(x) \neq g(\hat{x})$. Thus, $|g(\mathcal{A}^{|I|})| \geq R_{I,f}$, which implies a lower bound on a certain amount of "information" that X needs to send to Y to ensure that it can compute the function f. Note that $g = \Phi_{I,f}$ achieves the lower bound. We will use this intuition to establish a cut-based upper bound on the computing capacity $\mathcal{C}_{\text{cod}}(\mathcal{N}, f)$ of any network $\mathcal{N}$ with respect to any target function f, and to devise a capacity-achieving scheme for computing any target function in multi-edge tree networks.

Definition I.7: A set of edges $C \subseteq \mathcal{E}$ in network $\mathcal{N}$ is said to *separate* sources $\sigma_{m_1}, \ldots, \sigma_{m_d}$ from the receiver ρ, if for each $i \in \{1, 2, \ldots, d\}$, every directed path from σ_{m_i} to ρ contains at least one edge in C. The set C is said to be a *cut* in $\mathcal{N}$ if it separates at least one source from the receiver. For any network $\mathcal{N}$, define $\Lambda(\mathcal{N})$ to be the collection of all cuts in $\mathcal{N}$. For any cut $C \in \Lambda(\mathcal{N})$ and any target function f, define

$$\begin{aligned} I_C &= \{i : C \text{ separates } \sigma_i \text{ from the receiver}\} \\ R_{C,f} &= R_{I_C,f}. \end{aligned} \tag{2}$$

[4]Witsenhausen [46] represented this equivalence relation in terms of the independent sets of a characteristic graph and his representation has been used in various problems related to function computation [11], [12], [39]. Although $\equiv$ is defined with respect to a particular index set I and a function f, we do not make this dependence explicit—the values of I and f will be clear from the context.

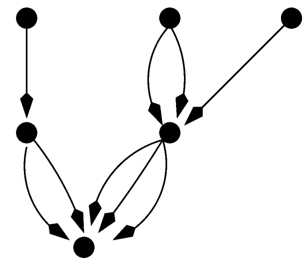

Fig. 2. Example of a multi-edge tree.

Since target functions depend on all sources, we have $R_{C,f} \geq 2$ for any cut C and any target function f.

A *multi-edge tree* is a graph such that for every node $v \in \mathcal{V}$, there exists a node u such that all the out-edges of v are in-edges to u, i.e., $\mathcal{E}_o(v) \subseteq \mathcal{E}_i(u)$ (e.g., see Fig. 2).

B. Classes of Target Functions

We study the following four classes of target functions: (1) divisible, (2) symmetric, (3) λ-exponential, (4) λ-bounded.

Definition I.8: A target function $f : \mathcal{A}^s \longrightarrow \mathcal{B}$ is *divisible* if for every index set $I \subseteq \{1, \ldots, s\}$, there exists a finite set $\mathcal{B}_I$ and a function $f^I : \mathcal{A}^{|I|} \longrightarrow \mathcal{B}_I$ such that the following hold:

(1) $f^{\{1,\ldots,s\}} = f$;

(2) $|f^I(\mathcal{A}^{|I|})| \leq |f(\mathcal{A}^s)|$;

(3) for every partition $\{I_1, \ldots, I_\gamma\}$ of I, there exists a function $g : \mathcal{B}_{I_1} \times \cdots \times \mathcal{B}_{I_\gamma} \longrightarrow \mathcal{B}_I$ such that for every $x \in \mathcal{A}^{|I|}$, we have $f^I(x) = g\left(f^{I_1}(x_{I_1}), \ldots, f^{I_\gamma}(x_{I_\gamma})\right)$.

Examples of divisible target functions include the identity, maximum, $\bmod r$ sum, and arithmetic sum.

Divisible functions have been studied previously[5] by Giridhar and Kumar [18] and Subramanian, Gupta, and Shakkottai [43]. Divisible target functions can be computed in networks in a divide-and-conquer fashion as follows. For any arbitrary partition $\{I_1, \ldots, I_\gamma\}$ of the source indices $\{1, \ldots, s\}$, the receiver ρ can evaluate the target function f by combining evaluations of $f^{I_1}, \ldots, f^{I_\gamma}$. Furthermore, for every $i = 1, \ldots, \gamma$, the target function f^{I_i} can be evaluated similarly by partitioning I_i and this process can be repeated until the function value is obtained.

Definition I.9: A target function $f : \mathcal{A}^s \longrightarrow \mathcal{B}$ is *symmetric* if for any permutation π of $\{1, 2, \ldots, s\}$ and any vector $x \in \mathcal{A}^s$,

$$f(x_1, x_2, \ldots, x_s) = f(x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(s)}).$$

That is, the value of a symmetric target function is invariant with respect to the order of its arguments and hence, it suffices to evaluate the histogram target function for computing any symmetric target function. Examples of symmetric functions include the arithmetic sum, maximum, and $\bmod r$ sum. Symmetric functions have been studied in the context of computing in networks by Giridhar *et al.* [18], Subramanian *et al.* [43], Ying *et al.* [50], and [26].

Definition I.10: Let $\lambda \in (0, 1]$. A target function $f : \mathcal{A}^s \longrightarrow \mathcal{B}$ is said to be λ-*exponential* if its footprint size satisfies

$$R_{I,f} \geq |\mathcal{A}|^{\lambda |I|} \text{ for every } I \subseteq \{1, 2, \ldots, s\}.$$

[5]The definitions in [18], [43] are similar to ours but slightly more restrictive.

Let $\lambda \in (0, \infty)$. A target function $f : \mathcal{A}^s \longrightarrow \mathcal{B}$ is said to be *λ-bounded* if its footprint size satisfies

$$R_{I,f} \leq |\mathcal{A}|^{\lambda} \text{ for every } I \subseteq \{1, 2, \ldots, s\}.$$

Example I.11: The following facts are easy to verify:

- The identity function is 1-exponential.
- Let $\mathcal{A}$ be an ordered set. The maximum (or minimum) function is 1-bounded.
- Let $\mathcal{A} = \{0, 1, \ldots, q-1\}$ where $q \geq 2$. The $\bmod\ r$ sum target function with $q \geq r \geq 2$ is $\log_q r$-bounded.

Remark I.12: Giridhar and Kumar [18] defined two classes of functions: *type-threshold* and *type-sensitive* functions. Both are sub-classes of symmetric functions. In addition, type-threshold functions are also divisible and λ-bounded, for some constant λ that is independent of the network size. However, [18] uses a model of interference for simultaneous transmissions and their results do not directly compare with ours.

Following the notation in Leighton and Rao [33], the *min-cut* of any network $\mathcal{N}$ with unit-capacity edges is

$$\text{min-cut}(\mathcal{N}) = \min_{C \in \Lambda(\mathcal{N})} \frac{|C|}{|I_C|}. \tag{3}$$

A more general version of the network min-cut plays a fundamental role in the field of multi-commodity flow [33], [44]. The min-cut provides an upper bound on the maximum flow for any multi-commodity flow problem. The min-cut is also referred to as "sparsity" by some authors, such as Harvey, Kleinberg, and Lehman [22] and Vazirani [44]. We next generalize the definition in (3) to the network computing problem.

Definition I.13: If $\mathcal{N}$ is a network and f is a target function, then

$$\text{min-cut}(\mathcal{N}, f) = \min_{C \in \Lambda(\mathcal{N})} \frac{|C|}{\log_{|\mathcal{A}|} R_{C,f}}. \tag{4}$$

Example I.14:

- If f is the identity target function, then
$$\text{min-cut}(\mathcal{N}, f) = \min_{C \in \Lambda(\mathcal{N})} \frac{|C|}{|I_C|}.$$
Thus for the identity function, the definition of min-cut in (3) and (4) coincide.
- Let $\mathcal{A} = \{0, 1, \ldots, q-1\}$. If f is the arithmetic sum target function, then
$$\text{min-cut}(\mathcal{N}, f) = \min_{C \in \Lambda(\mathcal{N})} \frac{|C|}{\log_q((q-1)|I_C| + 1)}. \tag{5}$$
- Let $\mathcal{A}$ be an ordered set. If f is the maximum target function, then
$$\text{min-cut}(\mathcal{N}, f) = \min_{C \in \Lambda(\mathcal{N})} |C|.$$

C. Contributions

The main results of this paper are as follows. In Section II, we show (Theorem II.1) that for any network $\mathcal{N}$ and any target function f, the quantity $\text{min-cut}(\mathcal{N}, f)$ is an upper bound on the computing capacity $\mathcal{C}_{\text{cod}}(\mathcal{N}, f)$. In Section III, we note that the computing capacity for any network with respect to the identity target function (Theorem III.1) and linear target functions over finite fields (Theorem III.2) is equal to the min-cut upper bound. We show that the min-cut bound on computing capacity can also be achieved for all multi-edge tree networks with any target function (Theorem III.3). For any network and any target function, a lower bound on the computing capacity is given in terms of the Steiner tree packing number (Theorem III.5). Another lower bound is given for networks with symmetric target functions (Theorem III.7). In Section IV, the tightness of the above-mentioned bounds is analyzed for divisible (Theorem IV.2), symmetric (Theorem IV.3), λ-exponential (Theorem IV.4), and λ-bounded (Theorem IV.5) target functions. For λ-exponential target functions, the computing capacity is at least λ times the min-cut. If every nonreceiver node in a network is a source, then for λ-bounded target functions the computing capacity is at least a constant times the min-cut divided by λ. It is also shown, with an example target function, that there are networks for which the computing capacity is less than an arbitrarily small fraction of the min-cut bound (Theorem IV.7). In Section V, we discuss an example network and target function in detail to illustrate the above bounds. In Section VI, conclusions are given and various lemmas are proven in the Appendix.

II. Min-Cut Upper Bound on Computing Capacity

The following shows that the maximum rate of computing a target function f in a network $\mathcal{N}$ is at most $\text{min-cut}(\mathcal{N}, f)$.

Theorem II.1: If $\mathcal{N}$ is a network with target function f, then

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) \leq \text{min-cut}(\mathcal{N}, f).$$

Proof: Let the network alphabet be $\mathcal{A}$ and consider any (k, n) code that computes f in $\mathcal{N}$. Let C be a cut and for each $i \in \{1, 2, \ldots, k\}$, let $a^{(i)}, b^{(i)} \in \mathcal{A}^{|I_C|}$. Suppose $j \in \{1, 2, \ldots, k\}$ is such that $a^{(j)} \not\equiv b^{(j)}$, where $\equiv$ is the equivalence relation from Definition I.4. Then there exist $x, y \in \mathcal{A}^s$ satsifying: $f(x) \neq f(y)$, $x_{I_C} = a^{(j)}$, $y_{I_C} = b^{(j)}$, and $x_i = y_i$ for every $i \notin I_C$.

The receiver ρ can compute the target function f only if, for every such pair $\{a^{(1)}, \ldots, a^{(k)}\}$ and $\{b^{(1)}, \ldots, b^{(k)}\}$ corresponding to the message vectors generated by the sources in I_C, the edges in cut C carry distinct vectors. Since the total number of equivalence classes for the relation $\equiv$ equals the footprint size $R_{C,f}$, the edges in cut C should carry at least $(R_{C,f})^k$ distinct vectors. Thus, we have

$$\mathcal{A}^{n|C|} \geq (R_{C,f})^k$$

and hence for any cut C,

$$\frac{k}{n} \leq \frac{|C|}{\log_{|\mathcal{A}|} R_{C,f}}.$$

Since the cut C is arbitrary, the result follows from Definition I.3 and (4). ∎

The min-cut upper bound has the following intuition. Given any cut $C \in \Lambda(\mathcal{N})$, at least $\log_{|\mathcal{A}|} R_{C,f}$ units of information need to be sent across the cut to successfully compute a target function f. In subsequent sections, we study the tightness of this bound for different classes of functions and networks.

III. Lower Bounds on the Computing Capacity

The following result shows that the computing capacity of any network $\mathcal{N}$ with respect to the identity target function equals the coding capacity for ordinary network coding.

Theorem III.1: If $\mathcal{N}$ is a network with the identity target function f, then

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) = \text{min-cut}(\mathcal{N}, f) = \text{min-cut}(\mathcal{N}).$$

Proof: Rasala Lehman and Lehman [32, p.6, Theorem 4.2] showed that for any single-receiver network, the conventional coding capacity (when the receiver demands the messages generated by all the sources) always equals the $\text{min-cut}(\mathcal{N})$. Since the target function f is the identity, the computing capacity is the coding capacity and $\text{min-cut}(\mathcal{N}, f) = \text{min-cut}(\mathcal{N})$, so the result follows. ■

Theorem III.2: If $\mathcal{N}$ is a network with a finite field alphabet and with a linear target function f, then

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) = \text{min-cut}(\mathcal{N}, f).$$

Proof: Follows from [41, Theorem 2]. ■

Theorems III.1 and III.2 demonstrate the achievability of the min-cut bound for arbitrary networks with particular target functions. In contrast, the following result demonstrates the achievability of the min-cut bound for arbitrary target functions and a particular class of networks. The following theorem concerns multi-edge tree networks, which were defined in Section I-A.

Theorem III.3: If $\mathcal{N}$ is a multi-edge tree network with target function f, then

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) = \text{min-cut}(\mathcal{N}, f).$$

Proof: Let $\mathcal{A}$ be the network alphabet. From Theorem II.1, it suffices to show that $\mathcal{C}_{\text{cod}}(\mathcal{N}, f) \geq \text{min-cut}(\mathcal{N}, f)$. Since $\mathcal{E}_o(v)$ is a cut for node $v \in \mathcal{V} - \rho$, and using (2), we have

$$\text{min-cut}(\mathcal{N}, f) \leq \min_{v \in \mathcal{V}-\rho} \frac{|\mathcal{E}_o(v)|}{\log_{|\mathcal{A}|} R_{\mathcal{E}_o(v), f}}. \quad (6)$$

Consider any positive integers k, n such that

$$\frac{k}{n} \leq \min_{v \in \mathcal{V}-\rho} \frac{|\mathcal{E}_o(v)|}{\log_{|\mathcal{A}|} R_{I_{\mathcal{E}_o(v)}, f}}. \quad (7)$$

Then we have

$$|\mathcal{A}|^{|\mathcal{E}_o(v)|n} \geq R^k_{I_{\mathcal{E}_o(v)}, f} \text{ for every node } v \in \mathcal{V} - \rho. \quad (8)$$

We outline a (k, n) code that computes f in the multi-edge tree network $\mathcal{N}$. Each source $\sigma_i \in S$ generates a message vector $\alpha(\sigma_i) \in \mathcal{A}^k$. Denote the vector of i-th components of the source messages by

$$x^{(i)} = (\alpha(\sigma_1)_i, \ldots, \alpha(\sigma_s)_i).$$

Every node $v \in \mathcal{V} - \rho$ sends out a unique index (as guaranteed by (8)) over $A^{|\mathcal{E}_o(v)|n}$ corresponding to the equivalence classes

$$\Phi_{I_{\mathcal{E}_o(v)}, f}\left(x^{(l)}_{I_{\mathcal{E}_o(v)}}\right) \text{ for } l \in \{1, \ldots, k\}. \quad (9)$$

If v has no in-edges, then by assumption, it is a source node, say σ_j. The set of equivalence classes in (9) is a function of its own messages $\alpha(\sigma_j)_l$ for $l \in \{1, \ldots, k\}$. On the other hand, if v has in-edges, then let $u_1, u_2, \ldots, u_j$ be the nodes with out-edges to v. For each $i \in \{1, 2, \ldots, j\}$, using the uniqueness of the index received from u_i, node v recovers the equivalence classes

$$\Phi_{I_{\mathcal{E}_o(u_i)}, f}\left(x^{(l)}_{I_{\mathcal{E}_o(u_i)}}\right) \text{ for } l \in \{1, \ldots, k\}. \quad (10)$$

Furthermore, the equivalence classes in (9) can be identified by v from the equivalance classes in (10) (and $\alpha(v)$ if v is a source node) using the fact that for a multi-edge tree network $\mathcal{N}$, if v is not a source node, then we have a disjoint union

$$I_{\mathcal{E}_o(v)} = \bigcup_{i=1}^{j} I_{\mathcal{E}_o(u_i)}.$$

If each node v follows the above steps, then the receiver ρ can identify the equivalence classes $\Phi_{I_{\mathcal{E}_i(\rho)}, f}(x^{(i)})$ for $i \in \{1, \ldots, k\}$. The receiver can evaluate $f(x^{(l)})$ for each l from these equivalence classes. The above network code computes f and achieves a computing rate of k/n. From (7), it follows that

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) \geq \min_{v \in \mathcal{V}-\rho} \frac{|\mathcal{E}_o(v)|}{\log_{|\mathcal{A}|} R_{I_{\mathcal{E}_o(v)}, f}}. \quad (11)$$

■

We next establish a general lower bound on the computing capacity for arbitrary target functions (Theorem III.5) and then another lower bound specifically for symmetric target functions (Theorem III.7).

For any network $\mathcal{N} = (G, S, \rho)$ with $G = (\mathcal{V}, \mathcal{E})$, define a *Steiner tree*[6] of $\mathcal{N}$ to be a minimal (with respect to nodes and edges) subgraph of G containing S and ρ such that every source in S has a directed path to the receiver ρ. Note that every nonreceiver node in a Steiner tree has exactly one out-edge. Let $\mathcal{T}(\mathcal{N})$ denote the collection of all Steiner trees in $\mathcal{N}$. For each edge $e \in \mathcal{E}(G)$, let $J_e = \{i : t_i \in \mathcal{T}(\mathcal{N}) \text{ and } e \in \mathcal{E}(t_i)\}$. The fractional Steiner tree packing number $\Pi(\mathcal{N})$ is defined as the linear program

$$\Pi(\mathcal{N}) = \max \sum_{t_i \in \mathcal{T}(\mathcal{N})} u_i \text{ subject to } \begin{cases} u_i \geq 0, & \forall\, t_i \in \mathcal{T}(\mathcal{N}) \\ \sum_{i \in J_e} u_i \leq 1, & \forall\, e \in \mathcal{E}(G). \end{cases} \quad (12)$$

Note that $\Pi(\mathcal{N}) \geq 1$ for any network $\mathcal{N}$, and the maximum value of the sum in (12) is attained at one or more vertices of the closed polytope corresponding to the linear constraints. Since all coefficients in the constraints are rational, the maximum value

[6]Steiner trees are well known in the literature for undirected graphs. For directed graphs a "Steiner tree problem" has been studied and our definition is consistent with such work (e.g., see [25]).

in (12) can be attained with rational u_i's. The following theorem provides a lower bound[7] on the computing capacity for any network $\mathcal{N}$ with respect to a target function f and uses the quantity $\Pi(\mathcal{N})$. In the context of computing functions, u_i in the above linear program indicates the fraction of the time the edges in tree t_i are used to compute the desired function. The fact that every edge in the network has unit capacity implies $\sum_{i\in J_e} u_i \le 1$.

Lemma III.4: For any Steiner tree G' of a network $\mathcal{N}$, let $\mathcal{N}' = (G', S, \rho)$. Let C' be a cut in $\mathcal{N}'$. Then there exists a cut C in $\mathcal{N}$ such that $I_C = I_{C'}$.

(Note that $I_{C'}$ is the set of indices of sources separated in $\mathcal{N}'$ by C'. The set $I_{C'}$ may differ from the indices of sources separated in $\mathcal{N}$ by C'.)

Proof: Define the cut

$$C = \bigcup_{i' \in I_{C'}} \mathcal{E}_o(\sigma_{i'}). \tag{13}$$

C is the collection of out-edges in $\mathcal{N}$ of a set of sources disconnected by the cut C' in $\mathcal{N}'$. If $i \in I_{C'}$, then, by (13), C disconnects σ_i from ρ in $\mathcal{N}$, and thus $I_{C'} \subseteq I_C$.

Let σ_i be a source such that $i \in I_C$ and let P be a path from σ_i to ρ in $\mathcal{N}$. From (13), it follows that there exists $i' \in I_{C'}$ such that P contains at least one edge in $\mathcal{E}_o(\sigma_{i'})$. If P also lies in $\mathcal{N}'$ and does not contain any edge in C', then $\sigma_{i'}$ has a path to ρ in $\mathcal{N}'$ that does not contain any edge in C', thus contradicting the fact that $\sigma_{i'} \in I_{C'}$. Therefore, either P does not lie in $\mathcal{N}'$ or P contains an edge in C'. Thus $\sigma_i \in I_{C'}$, i.e., $I_C \subseteq I_{C'}$. ∎

Theorem III.5: If $\mathcal{N}$ is a network with alphabet $\mathcal{A}$ and target function f, then

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) \ge \Pi(\mathcal{N}) \cdot \min_{C \in \Lambda(\mathcal{N})} \frac{1}{\log_{|\mathcal{A}|} R_{C,f}}.$$

Proof: Suppose $\mathcal{N} = (G, S, \rho)$. Consider a Steiner tree $G' = (\mathcal{V}', \mathcal{E}')$ of $\mathcal{N}$, and let $\mathcal{N}' = (G', S, \rho)$. From Lemma III.4 (taking C' to be $\mathcal{E}_o(v)$ in $\mathcal{N}'$), we have

$$\forall\, v \in \mathcal{V}' - \rho,\ \exists C \in \Lambda(\mathcal{N}) \text{ such that } I'_{\mathcal{E}_o(v)} = I_C. \tag{14}$$

Now we lower bound the computing capacity for the network $\mathcal{N}'$ with respect to target function f:

$$\begin{aligned}
&\mathcal{C}_{\text{cod}}(\mathcal{N}', f) \\
&= \text{min-cut}(\mathcal{N}, f) \quad \text{[from Theorem III.3]} &(15)\\
&= \min_{v \in \mathcal{V}' - \rho} \frac{1}{\log_{|\mathcal{A}|} R_{I'_{\mathcal{E}_o(v)}, f}} \quad \text{[from Theorem II.1, (6), (11)]}\\
&\ge \min_{C \in \Lambda(\mathcal{N})} \frac{1}{\log_{|\mathcal{A}|} R_{I_C, f}} \quad \text{[from (14)]}. &(16)
\end{aligned}$$

The lower bound in (16) is the same for every Steiner tree of $\mathcal{N}$. We will use this uniform bound to lower bound the computing capacity for $\mathcal{N}$ with respect to f. Denote the Steiner trees of $\mathcal{N}$ by $t_1, \ldots, t_T$. Let $\epsilon > 0$ and let r denote the quantity on the right hand side of (16). On every Steiner tree t_i, a computing rate of at least $r - \epsilon$ is achievable by (16). Using standard arguments for time-sharing between the different Steiner trees of the network $\mathcal{N}$, it follows that a computing rate of at least $(r - \epsilon) \cdot \Pi(\mathcal{N})$ is achievable in $\mathcal{N}$, and by letting $\epsilon \to 0$, the result follows. ∎

The lower bound in Theorem III.5 can be readily computed and is sometimes tight. The procedure used in the proof of Theorem III.5 may potentially be improved by maximizing the sum

$$\sum_{t_i \in \mathcal{T}(\mathcal{N})} u_i r_i \quad \text{subject to} \quad \begin{cases} u_i \ge 0 & \forall\, t_i \in \mathcal{T}(\mathcal{N}) \\ \sum\limits_{i \in J_e} u_i \le 1 & \forall\, e \in \mathcal{E}(G) \end{cases} \tag{17}$$

where r_i is any achievable rate[8] for computing f in the Steiner tree network $\mathcal{N}_i = (t_i, S, \rho)$.

We now obtain a different lower bound on the computing capacity in the special case when the target function is the arithmetic sum. This lower bound is then used to give an alternative lower bound (in Theorem III.7) on the computing capacity for the class of symmetric target functions. The bound obtained in Theorem III.7 is sometimes better than that of Theorem III.5, and sometimes worse (Example III.8 illustrates instances of both cases).

Theorem III.6: If $\mathcal{N}$ is a network with alphabet $\mathcal{A} = \{0, 1, \ldots, q-1\}$ and the arithmetic sum target function f, then

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) \ge \min_{C \in \Lambda(\mathcal{N})} \frac{|C|}{\log_q P_{q,s}}$$

where $P_{q,s}$ denotes the smallest prime number greater than $s(q-1)$.

Proof: Let $p = P_{q,s}$ and let $\mathcal{N}'$ denote the same network as $\mathcal{N}$ but whose alphabet is $\mathbb{F}_p$, the finite field of order p.

Let $\epsilon > 0$. From Theorem III.2, there exists a (k, n) code that computes the $\mathbb{F}_p$-sum of the source messages in $\mathcal{N}'$ with an achievable computing rate satisfying

$$\frac{k}{n} \ge \min_{C \in \Lambda(\mathcal{N})} |C| - \epsilon.$$

This (k, n) code can be repeated to derive a (ck, cn) code that also computes f for any integer $c \ge 1$ (note that edges in the network $\mathcal{N}$ carry symbols from the alphabet $\mathcal{A} = \{0, 1, \ldots, q-1\}$, while those in the network $\mathcal{N}'$ carry symbols from a larger alphabet $\mathbb{F}_p$). Any (ck, cn) code that computes the $\mathbb{F}_p$-sum in $\mathcal{N}'$ can be "simulated" in the network $\mathcal{N}$ by a $(ck, \lceil cn \log_q p \rceil)$ code (e.g., see [2]). Furthermore, since $p \ge s(q-1)+1$ and the source alphabet is $\{0, 1, \ldots, q-1\}$, the $\mathbb{F}_p$-sum of the source messages in network $\mathcal{N}$ is equal to their arithmetic sum. Thus, by choosing c large enough, the arithmetic sum target function is computed in $\mathcal{N}$ with an achievable computing rate of at least

$$\frac{\min\limits_{C \in \Lambda(\mathcal{N})} |C|}{\log_q p} - 2\epsilon.$$

Since ϵ is arbitrary, the result follows. ∎

[7]In order to compute the lower bound, the fractional Steiner tree packing number $\Pi(\mathcal{N})$ can be evaluated using linear programming. Also note that if we construct the reverse multicast network by letting each source in the original network $\mathcal{N}$ become a receiver, letting the receiver in the $\mathcal{N}$ become the only source, and reversing the direction of each edge, then it can be verified that the routing capacity for the reverse multicast network is equal to $\Pi(\mathcal{N})$.

[8]From Theorem III.3, r_i can be arbitrarily close to $\text{min-cut}(t_i, f)$.

Theorem III.7: If $\mathcal{N}$ is a network with alphabet $\mathcal{A} = \{0, 1, \ldots, q-1\}$ and a symmetric target function f, then

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) \geq \frac{\min_{C \in \Lambda(\mathcal{N})} |C|}{(q-1) \cdot \log_q P(s)}$$

where $P(s)$ is the smallest prime number[9] greater than s.

Proof: From Definition I.9, it suffices to evaluate the histogram target function $\hat{f}$ for computing f. For any set of source messages $(x_1, x_2, \ldots, x_s) \in \mathcal{A}^s$, we have

$$\hat{f}(x_1, \ldots, x_s) = (c_0, c_1, \ldots, c_{q-1})$$

where $c_i = |\{j : x_j = i\}|$ for each $i \in \mathcal{A}$. Consider the network $\mathcal{N}' = (G, S, \rho)$ with alphabet $\mathcal{A}' = \{0, 1\}$. Then for each $i \in \mathcal{A}$, c_i can be evaluated by computing the arithmetic sum target function in $\mathcal{N}'$ where every source node σ_j is assigned the message 1 if $x_j = i$, and 0 otherwise. Since we know that

$$\sum_{i=0}^{q-1} c_i = s$$

the histogram target function $\hat{f}$ can be evaluated by computing the arithmetic sum target function $q-1$ times in the network $\mathcal{N}'$ with alphabet $\mathcal{A}' = \{0, 1\}$. Let $\epsilon > 0$. From Theorem III.6 in the Appendix, there exists a (k, n) code that computes the arithmetic sum target function in $\mathcal{N}'$ with an achievable computing rate of at least

$$\frac{k}{n} \geq \frac{\min_{C \in \Lambda(\mathcal{N})} |C|}{\log_2 P(s)} - \epsilon.$$

The above (k, n) code can be repeated to derive a (ck, cn) code that computes f for any integer $c \geq 1$. Note that edges in the network $\mathcal{N}$ carry symbols from the alphabet $\mathcal{A} = \{0, 1, \ldots, q-1\}$, while those in the network $\mathcal{N}'$ carry symbols from $\mathcal{A}' = \{0, 1\}$. Any (ck, cn) code that computes the arithmetic sum target function in $\mathcal{N}'$ can be simulated in the network $\mathcal{N}$ by a $(ck, \lceil cn \log_q 2 \rceil)$ code[10]. Thus by choosing c large enough, the above-mentioned code can be simulated in the network $\mathcal{N}$ to derive a code that computes the histogram target function $\hat{f}$ with an achievable computing rate[11] of at least

$$\frac{1}{(q-1)} \cdot \frac{1}{\log_q 2} \cdot \frac{\min_{C \in \Lambda(\mathcal{N})} |C|}{\log_2 P(s)} - 2\epsilon.$$

Since ϵ is arbitrary, the result follows. ■

Example III.8: Consider networks $\mathcal{N}_2$ and $\mathcal{N}_3$ in Fig. 3, each with alphabet $\mathcal{A} = \{0, 1\}$ and the (symmetric) arithmetic sum target function f. Theorem III.7 provides a larger lower bound on the computing capacity $\mathcal{C}_{\text{cod}}(\mathcal{N}_2, f)$ than Theorem III.5, but a smaller lower bound on $\mathcal{C}_{\text{cod}}(\mathcal{N}_3, f)$.

[9]From Bertrand's Postulate [21, p.343], we have $P(s) \leq 2s$.

[10]To see details of such a simulation, we refer the interested reader to [2].

[11]Theorem III.7 provides a uniform lower bound on the achievable computing rate for any symmetric function. Better lower bounds can be found by considering specific functions; for example Theorem III.6 gives a better bound for the arithmetic sum target function.

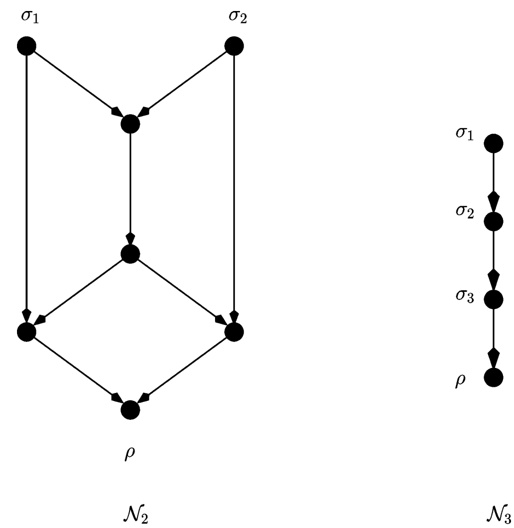

Fig. 3. Reverse Butterfly Network $\mathcal{N}_2$ has two binary sources $\{\sigma_1, \sigma_2\}$ and network $\mathcal{N}_3$ has three binary sources $\{\sigma_1, \sigma_2, \sigma_3\}$, each with $\mathcal{A} = \{0, 1\}$. Each network's receiver ρ computes the arithmetic sum of the source messages.

- For network $\mathcal{N}_2$ (in Fig. 3), we have $\max_{C \in \Lambda(\mathcal{N})} R_{C,f} = 3$ and $\min_{C \in \Lambda(\mathcal{N})} |C| = 2$, both of which occur, for example, when C consists of the two in-edges to the receiver ρ. Also, $(q-1) \log_q P(s, q) = \log_2 3$ and $\Pi(\mathcal{N}) = 3/2$, so

$$\begin{aligned} \mathcal{C}_{\text{cod}}(\mathcal{N}_2, f) &\geq (3/2)/\log_2 3 \quad \text{[from Theorem III.5]} \\ \mathcal{C}_{\text{cod}}(\mathcal{N}_2, f) &\geq 2/\log_2 3 \quad \text{[from Theorem III.7]}. \end{aligned} \tag{18}$$

  In fact, we get the upper bound $\mathcal{C}_{\text{cod}}(\mathcal{N}_2, f) \leq 2/\log_2 3$ from Theorem II.1, and thus from (18), $\mathcal{C}_{\text{cod}}(\mathcal{N}_2, f) = 2/\log_2 3$.
- For network $\mathcal{N}_3$, we have $\max_{C \in \Lambda(\mathcal{N})} R_{C,f} = 4$ and $\min_{C \in \Lambda(\mathcal{N})} |C| = 1$, both of which occur when $C = \{(\sigma_3, \rho)\}$. Also, $(q-1) \log_q P(s, q) = \log_2 5$ and $\Pi(\mathcal{N}) = 1$, so

$$\begin{aligned} \mathcal{C}_{\text{cod}}(\mathcal{N}_3, f) &\geq 1/\log_2 4 \quad \text{[from Theorem III.5]} \\ \mathcal{C}_{\text{cod}}(\mathcal{N}_3, f) &\geq 1/\log_2 5 \quad \text{[from Theorem III.7]}. \end{aligned}$$

  From Theorem III.3, we have $\mathcal{C}_{\text{cod}}(\mathcal{N}_3, f) = 1/\log_2 4$.

Remark III.9: An open question in network coding, pointed out in [7], is whether the coding capacity of a network can be irrational. Like the coding capacity, the computing capacity is the supremum of ratios k/n for which a (k, n) code exists that computes the target function. Example III.8 demonstrates that the computing capacity of a network (e.g., $\mathcal{N}_2$) with unit capacity links can be irrational when the target function is the arithmetic sum function.

IV. On the Tightness of the Min-Cut Upper Bound

In the previous section, Theorems III.1—III.3 demonstrated three special instances for which the $\text{min-cut}(\mathcal{N}, f)$ upper bound is tight. In this section, we use Theorem III.5 and Theorem III.7 to establish further results on the tightness of the $\text{min-cut}(\mathcal{N}, f)$ upper bound for different classes of target functions.

The following lemma provides a bound on the footprint size $R_{I,f}$ for any divisible target function f.

Lemma IV.1: For any divisible target function $f : \mathcal{A}^s \longrightarrow \mathcal{B}$ and any index set $I \subseteq \{1, 2, \ldots, s\}$, the footprint size satisfies

$$R_{I,f} \leq |f(\mathcal{A}^s)|.$$

Proof: From the definition of a divisible target function, for any $I \subseteq \{1, 2, \ldots, s\}$, there exist maps f^I, f^{I^c}, and g such that

$$f(x) = g\left(f^I(x_I), f^{I^c}(x_{I^c})\right) \quad \forall\, x \in \mathcal{A}^s$$

where $I^c = \{1, 2, \ldots, s\} - I$. From the definition of the equivalence relation $\equiv$ (see Definition I.4), it follows that $a, b \in \mathcal{A}^{|I|}$ belong to the same equivalence class whenever $f^I(a) = f^I(b)$. This fact implies that $R_{I,f} \leq |f^I(\mathcal{A}^{|I|})|$. We need $|f^I(\mathcal{A}^{|I|})| \leq |f(\mathcal{A}^s)|$ to complete the proof which follows from Definition I.8(2). ■

Theorem IV.2: If $\mathcal{N}$ is a network with a divisible target function f, then

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) \geq \frac{\Pi(\mathcal{N})}{|\mathcal{E}_i(\rho)|} \cdot \text{min-cut}(\mathcal{N}, f)$$

where $\mathcal{E}_i(\rho)$ denotes the set of in-edges of the receiver ρ.

Proof: Let $\mathcal{A}$ be the network alphabet. From Theorem III.5

$$\begin{aligned} \mathcal{C}_{\text{cod}}(\mathcal{N}, f) &\geq \Pi(\mathcal{N}) \cdot \min_{C \in \Lambda(\mathcal{N})} \frac{1}{\log_{|\mathcal{A}|} R_{C,f}} \\ &\geq \Pi(\mathcal{N}) \cdot \frac{1}{\log_{|\mathcal{A}|} |f(\mathcal{A}^s)|} \\ &\quad \text{[from Lemma IV.1]}. \end{aligned} \tag{19}$$

On the other hand, for any network $\mathcal{N}$, the set of edges $\mathcal{E}_i(\rho)$ is a cut that separates the set of sources S from ρ. Thus

$$\begin{aligned} &\text{min-cut}(\mathcal{N}, f) \\ &\quad \leq \frac{|\mathcal{E}_i(\rho)|}{\log_{|\mathcal{A}|} R_{\mathcal{E}_i(\rho),f}} \quad \text{[from (4)]} \\ &\quad = \frac{|\mathcal{E}_i(\rho)|}{\log_{|\mathcal{A}|} |f(\mathcal{A}^s)|} \\ &\qquad \text{[from } I_{\mathcal{E}_i(\rho)} = S \text{ and Definition I.5]}. \end{aligned} \tag{20}$$

Combining (19) and (20) completes the proof. ■

Theorem IV.3: If $\mathcal{N}$ is a network with alphabet $\mathcal{A} = \{0, 1, \ldots, q-1\}$ and symmetric target function f, then

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) \geq \frac{\log_q \hat{R}_f}{(q-1) \cdot \log_q P(s)} \cdot \text{min-cut}(\mathcal{N}, f)$$

where $P(s)$ is the smallest prime number greater than s and[12]

$$\hat{R}_f = \min_{I \subseteq \{1,\ldots,s\}} R_{I,f}.$$

[12]From our assumption, $\hat{R}_f \geq 2$ for any target function f.

Proof: The result follows immediately from Theorem III.7 and since for any network $\mathcal{N}$ and any target function f

$$\begin{aligned} &\text{min-cut}(\mathcal{N}, f) \\ &\quad \leq \frac{1}{\log_q \hat{R}_f} \cdot \min_{C \in \Lambda(\mathcal{N})} |C| \\ &\qquad \text{[from (4) and the definition of } \hat{R}_f]. \end{aligned}$$

■

The following results provide bounds on the gap between the computing capacity and the min-cut for λ-exponential and λ-bounded functions (see Definition I.10).

Theorem IV.4: If $\lambda \in (0, 1]$ and $\mathcal{N}$ is a network with a λ-exponential target function f, then

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) \geq \lambda \cdot \text{min-cut}(\mathcal{N}, f).$$

Proof: We have

$$\begin{aligned} &\text{min-cut}(\mathcal{N}, f) \\ &\quad = \min_{C \in \Lambda(\mathcal{N})} \frac{|C|}{\log_{|\mathcal{A}|} R_{C,f}} \\ &\quad \leq \min_{C \in \Lambda(\mathcal{N})} \frac{|C|}{\lambda |I_C|} \quad \text{[from } f \text{ being } \lambda\text{-exponential]} \\ &\quad = \frac{1}{\lambda} \cdot \text{min-cut}(\mathcal{N}) \quad \text{[from (3)]}. \end{aligned}$$

Therefore

$$\frac{\text{min-cut}(\mathcal{N}, f)}{\mathcal{C}_{\text{cod}}(\mathcal{N}, f)} \leq \frac{1}{\lambda} \cdot \frac{\text{min-cut}(\mathcal{N})}{\mathcal{C}_{\text{cod}}(\mathcal{N}, f)} \leq \frac{1}{\lambda}$$

where the last inequality follows because a computing rate of $\text{min-cut}(\mathcal{N})$ is achievable for the identity target function from Theorem III.1, and the computing capacity for any target function f is lower bounded by the computing capacity for the identity target function (since any target function can be computed from the identity function), i.e., $\mathcal{C}_{\text{cod}}(\mathcal{N}, f) \geq \text{min-cut}(\mathcal{N})$. ■

Theorem IV.5: Let $\lambda > 0$. If $\mathcal{N}$ is a network with alphabet $\mathcal{A}$ and a λ-bounded target function f, and all nonreceiver nodes in the network $\mathcal{N}$ are sources, then

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) \geq \frac{\log_{|\mathcal{A}|} \hat{R}_f}{\lambda} \cdot \text{min-cut}(\mathcal{N}, f)$$

where

$$\hat{R}_f = \min_{I \subseteq \{1,\ldots,s\}} R_{I,f}.$$

Proof: For any network $\mathcal{N}$ such that all nonreceiver nodes are sources, it follows from Edmond's Theorem [45, p.405, Theorem 8.4.20] that

$$\Pi(\mathcal{N}) = \min_{C \in \Lambda(\mathcal{N})} |C|.$$

Then

$$\begin{aligned}&\mathcal{C}_{\text{cod}}(\mathcal{N}, f)\\ &\geq \min_{C\in\Lambda(\mathcal{N})} |C| \cdot \min_{C\in\Lambda(\mathcal{N})} \frac{1}{\log_{|\mathcal{A}|} R_{C,f}}\\ &\quad \text{[from Theorem III.5]}\\ &\geq \min_{C\in\Lambda(\mathcal{N})} \frac{|C|}{\lambda} \quad \text{[from } f \text{ being } \lambda\text{-bounded].} \qquad (21)\end{aligned}$$

On the other hand,

$$\begin{aligned}&\text{min-cut}(\mathcal{N}, f)\\ &= \min_{C\in\Lambda(\mathcal{N})} \frac{|C|}{\log_{|\mathcal{A}|} R_{C,f}}\\ &\leq \min_{C\in\Lambda(\mathcal{N})} \frac{|C|}{\log_{|\mathcal{A}|} \hat{R}_f} \quad \text{[from the definition of } \hat{R}_f\text{].} \qquad (22)\end{aligned}$$

Combining (21) and (22) gives

$$\begin{aligned}\frac{\text{min-cut}(\mathcal{N}, f)}{\mathcal{C}_{\text{cod}}(\mathcal{N}, f)} &\leq \min_{C\in\Lambda(\mathcal{N})} \frac{|C|}{\log_{|\mathcal{A}|} \hat{R}_f} \cdot \frac{1}{\min_{C\in\Lambda(\mathcal{N})} \frac{|C|}{\lambda}}\\ &= \frac{\lambda}{\log_{|\mathcal{A}|} \hat{R}_f}.\end{aligned}$$

■

Since the maximum and minimum target functions are 1-bounded, and $\hat{R}_f = |\mathcal{A}|$ for each one, we get the following corollary.

Corollary IV.6: Let $\mathcal{A}$ be any ordered alphabet and let $\mathcal{N}$ be any network such that all nonreceiver nodes in the network are sources. If the target function f is either the maximum or the minimum function, then

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) = \text{min-cut}(\mathcal{N}, f).$$

Theorems IV.2—IV.5 provide bounds on the tightness of the $\text{min-cut}(\mathcal{N}, f)$ upper bound for different classes of target functions. In particular, we show that for λ-exponential (respectively, λ-bounded) target functions, the computing capacity $\mathcal{C}_{\text{cod}}(\mathcal{N}, f)$ is at least a constant fraction of the $\text{min-cut}(\mathcal{N}, f)$ for any constant λ and any network $\mathcal{N}$ (respectively, any network $\mathcal{N}$ where all nonreceiver nodes are sources). The following theorem shows by means of an example target function f and a network $\mathcal{N}$, that the $\text{min-cut}(\mathcal{N}, f)$ upper bound cannot always approximate the computing capacity $\mathcal{C}_{\text{cod}}(\mathcal{N}, f)$ up to a constant fraction. Similar results are known in network coding as well as in multicommodity flow. It was shown in [33] that when s source nodes communicate independently with the same number of receiver nodes, there exist networks whose maximum multicommodity flow is $O(1/\log s)$ times a well known cut-based upper bound. It was shown in [23] that with network coding there exist networks whose maximum throughput is $O(1/\log s)$ times the best known cut bound (i.e., "meagerness"). Whereas these results do not hold for single-receiver networks (by Theorem III.1), the following similar bound holds for network computing in single-receiver networks. The proof of Theorem IV.7 uses Lemma VII.1 which is presented in the Appendix.

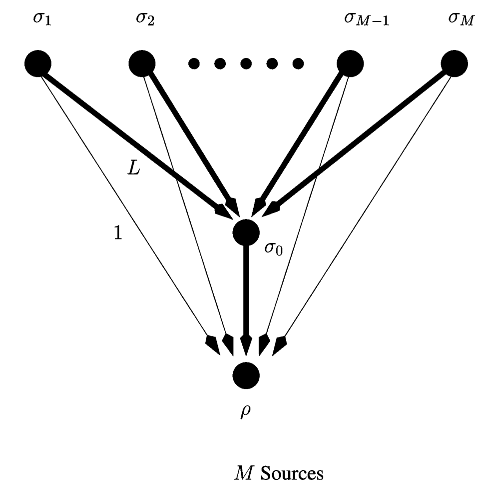

Fig. 4. Network $\mathcal{N}_{M,L}$ has M binary sources $\{\sigma_1, \sigma_2, \ldots, \sigma_M\}$, with $\mathcal{A} = \{0, 1\}$, connected to the receiver node ρ via a relay σ_0. Each bold edge denotes L parallel capacity-one edges and ρ computes the arithmetic sum of the source messages.

Theorem IV.7: For any $\epsilon > 0$, there exists a network $\mathcal{N}$ such that for the arithmetic sum target function f

$$\mathcal{C}_{\text{cod}}(\mathcal{N}, f) = O\left(\frac{1}{(\log s)^{1-\epsilon}}\right) \cdot \text{min-cut}(\mathcal{N}, f).$$

Proof: Consider the network $\mathcal{N}_{M,L}$ depicted in Fig. 4 with alphabet $\mathcal{A} = \{0, 1\}$ and the arithmetic sum target function f. Then we have

$$\text{min-cut}(\mathcal{N}_{M,L}, f) = \min_{C\in\Lambda(\mathcal{N}_{M,L})} \frac{|C|}{\log_2(|I_C| + 1)} \quad \text{[from (5)].}$$

Let m be the number of sources disconnected from the receiver ρ by a cut C in the network $\mathcal{N}_{M,L}$. For each such source σ, the cut C must contain the edge (σ, ρ) as well as either the L parallel edges (σ, σ_0) or the L parallel edges (σ_0, ρ). Thus

$$\text{min-cut}(\mathcal{N}_{M,L}, f) = \min_{1\leq m\leq M} \left\{\frac{L+m}{\log_2(m+1)}\right\}. \qquad (23)$$

Let m^* attain the minimum in [23] and define $c^* = \text{min-cut}(\mathcal{N}_{M,L}, f)$. Then

$$\begin{aligned}c^*/\ln 2 &\geq \min_{1\leq m\leq M} \left\{\frac{m+1}{\ln(m+1)}\right\}\\ &\geq \min_{x\geq 2} \left\{\frac{x}{\ln x}\right\} = e > 1 \qquad (24)\\ L &= c^* \log_2(m^* + 1) - m^* \quad \text{[from (23)]}\\ &\leq c^* \log_2\left(\frac{c^*}{\ln 2}\right) - \left(\frac{c^*}{\ln 2} - 1\right) \qquad (25)\end{aligned}$$

where (25) follows since the function $c^* \log_2(x+1) - x$ attains its maximum value over $(0, \infty)$ at $x = (c^*/\ln 2) - 1$ ($\geq e - 1$ by (24)). Let us choose $L = \lceil (\log M)^{1-(\epsilon/2)} \rceil$. We have

$$\begin{aligned}&L = O(\text{min-cut}(\mathcal{N}_{M,L}, f) \log_2(\text{min-cut}(\mathcal{N}_{M,L}, f)))\\ &\qquad\qquad \text{[from (25)]} \qquad (26)\\ &\text{min-cut}(\mathcal{N}_{M,L}, f) = \Omega((\log M)^{1-\epsilon}) \quad \text{[from (26)]} \qquad (27)\end{aligned}$$

$$\mathcal{C}_{\rm cod}(\mathcal{N}_{M,L}, f) = O(1) \quad \text{[from Lemma VII.1]}$$
$$= O\left(\frac{1}{(\log M)^{1-\epsilon}}\right) \cdot \text{min-cut}(\mathcal{N}_{M,L}, f) \quad \text{[from (27)].}$$

■

V. An Example Network

In this section, we evaluate the computing capacity for an example network and a target function (which is divisible and symmetric) and show that the min-cut bound is not tight. In addition, the example demonstrates that the lower bounds discussed in Section III are not always tight and illustrates the combinatorial nature of the computing problem.

Theorem V.1: The computing capacity of network $\hat{\mathcal{N}}$ with respect to the arithmetic sum target function f is

$$\mathcal{C}_{\rm cod}(\hat{\mathcal{N}}, f) = \frac{2}{1+\log_2 3}.$$

Proof: For any (k, n) code that computes f, let $w^{(1)}$, $w^{(2)}$, $w^{(3)} \in \{0,1\}^k$ denote the message vectors generated by sources $\sigma_1, \sigma_2, \sigma_3$, respectively, and let z_1, $z_2 \in \{0,1\}^n$ be the vectors carried by edges (σ_1, ρ) and (σ_2, ρ), respectively.

Consider any positive integers k, n such that k is even and

$$\frac{k}{n} \le \frac{2}{1+\log_2 3}. \tag{28}$$

Then we have

$$2^n \ge 3^{k/2} 2^{k/2}. \tag{29}$$

We will describe a (k,n) network code that computes f in the network $\hat{\mathcal{N}}$. Define vectors $y^{(1)}$, $y^{(2)} \in \{0,1\}^k$ by

$$y_i^{(1)} = \begin{cases} w_i^{(1)} + w_i^{(3)}, & \text{if } 1 \le i \le k/2 \\ w_i^{(1)}, & \text{if } k/2 \le i \le k \end{cases}$$
$$y_i^{(2)} = \begin{cases} w_i^{(2)}, & \text{if } 1 \le i \le k/2 \\ w_i^{(2)} + w_i^{(3)}, & \text{if } k/2 \le i \le k. \end{cases}$$

The first $k/2$ components of $y^{(1)}$ can take on the values 0, 1, 2, and the last $k/2$ components can take on the values 0, 1, so there are a total of $3^{k/2} 2^{k/2}$ possible values for $y^{(1)}$, and similarly for $y^{(2)}$. From (29), there exists a mapping that assigns unique values to z_1 for each different possible value of $y^{(1)}$, and similarly for z_2 and $y^{(2)}$. This induces a code for $\hat{\mathcal{N}}$ that computes f as summarized below.

The source σ_3 sends its full message vector $w^{(3)} (k < n)$ to each of the two nodes it is connected to. Source σ_1 (respectively, σ_2) computes the vector $y^{(1)}$ (respectively, $y^{(2)}$), then computes the vector z_1 (respectively, z_2), and finally sends z_1 (respectively, z_2) on its out-edge. The receiver ρ determines $y^{(1)}$ and $y^{(2)}$ from z_1 and z_2, respectively, and then computes $y^{(1)} + y^{(2)}$, whose i-th component is $w_i^{(1)} + w_i^{(2)} + w_i^{(3)}$, i.e., the arithmetic sum target function f. The above code achieves a computing rate of k/n. From [28], it follows that

$$\mathcal{C}_{\rm cod}(\hat{\mathcal{N}}, f) \ge \frac{2}{1+\log_2 3}. \tag{30}$$

We now prove a matching upper bound on the computing capacity $\mathcal{C}_{\rm cod}(\hat{\mathcal{N}}, f)$. Consider any (k,n) code that computes the arithmetic sum target function f in network $\hat{\mathcal{N}}$. For any $p \in \{0,1,2,3\}^k$, let

$$A_p = \left\{(z_1, z_2) : w^{(1)} + w^{(2)} + w^{(3)} = p\right\}.$$

That is, each element of A_p is a possible pair of input edge-vectors to the receiver when the function value equals p.

Let j denote the number of components of p that are either 0 or 3. Without loss of generality, suppose the first j components of p belong to $\{0,3\}$ and define $\tilde{w}^{(3)} \in \{0,1\}^k$ by

$$\tilde{w}_i^{(3)} = \begin{cases} 0, & \text{if } p_i \in \{0,1\} \\ 1, & \text{if } p_i \in \{2,3\}. \end{cases}$$

Let

$$T = \Big\{(w^{(1)}, w^{(2)}) \in \{0,1\}^k \times \{0,1\}^k : w^{(1)} + w^{(2)} + \tilde{w}^{(3)} = p\Big\}$$

and notice that

$$\left\{(z_1, z_2) : (w^{(1)}, w^{(2)}) \in T,\ w^{(3)} = \tilde{w}^{(3)}\right\} \subseteq A_p. \tag{31}$$

If $w^{(1)} + w^{(2)} + \tilde{w}^{(3)} = p$, then:

(i) $p_i - \tilde{w}_i^{(3)} = 0$ implies $w_i^{(1)} = w_i^{(2)} = 0$;

(ii) $p_i - \tilde{w}_i^{(3)} = 2$ implies $w_i^{(1)} = w_i^{(2)} = 1$;

(iii) $p_i - \tilde{w}_i^{(3)} = 1$ implies $\left(w_i^{(1)}, w_i^{(2)}\right) = (0,1)$ or (1,0).

Thus, the elements of T consist of k-bit vector pairs $(w^{(1)}, w^{(2)})$ whose first j components are fixed and equal (i.e., both are 0 when $p_i = 0$ and both are 1 when $p_i = 3$), and whose remaining $k - j$ components can each be chosen from two possibilities (i.e., either (0, 1) or (1, 0), when $p_i \in \{1,2\}$). This observation implies that

$$|T| = 2^{k-j}. \tag{32}$$

Notice that if only $w^{(1)}$ changes, then the sum $w^{(1)} + w^{(2)} + w^{(3)}$ changes, and so z_1 must change (since z_2 is not a function of $w^{(1)}$) in order for the receiver to compute the target function. Thus, if $w^{(1)}$ changes and $w^{(3)}$ does not change, then z_1 must still change, regardless of whether $w^{(2)}$ changes or not. More generally, if the pair $(w^{(1)}, w^{(2)})$ changes, then the pair (z_1, z_2) must change. Thus

$$\left|\left\{(z_1, z_2) : (w^{(1)}, w^{(2)}) \in T, w^{(3)} = \tilde{w}^{(3)}\right\}\right| \ge |T| \tag{33}$$

and therefore

$$\begin{aligned} |A_p| &\ge \left|\left\{(z_1, z_2) : (w^{(1)}, w^{(2)}) \in T, w^{(3)} = \tilde{w}^{(3)}\right\}\right| \\ &\quad \text{[from (31)]} \\ &\ge |T| \quad \text{[from (33)]} \\ &= 2^{k-j}. \quad \text{[from (32)].} \end{aligned} \tag{34}$$

We have the following inequalities:

$$\begin{aligned}4^n &\geq \left|\{(z_1,z_2) : w^{(1)}, w^{(2)}, w^{(3)} \in \{0,1\}^k\}\right| \\ &= \sum_{p\in\{0,1,2,3\}^k} |A_p| \qquad (35)\\ &= \sum_{j=0}^{k} \sum_{\substack{p\in\{0,1,2,3\}^k \\ |\{i:p_i\in\{0,3\}\}|=j}} |A_p| \\ &\geq \sum_{j=0}^{k} \sum_{\substack{p\in\{0,1,2,3\}^k \\ |\{i:p_i\in\{0,3\}\}|=j}} 2^{k-j} \quad \text{[from (34)]} \\ &= \sum_{j=0}^{k} \binom{k}{j} 2^k 2^{k-j} \\ &= 6^k \qquad (36)\end{aligned}$$

where (35) follows since the A_p's must be disjoint in order for the receiver to compute the target function. Taking logarithms of both sides of (36), gives

$$\frac{k}{n} \leq \frac{2}{1+\log_2 3}$$

which holds for all k and n, and therefore

$$\mathcal{C}_{\mathrm{cod}}(\hat{\mathcal{N}}, f) \leq \frac{2}{1+\log_2 3}. \qquad (37)$$

Combining (30) and (37) concludes the proof. ■

Corollary V.2: For the network $\hat{\mathcal{N}}$ with the arithmetic sum target function f

$$\mathcal{C}_{\mathrm{cod}}(\hat{\mathcal{N}}, f) < \text{min-cut}(\hat{\mathcal{N}}, f).$$

Proof: Consider the network $\hat{\mathcal{N}}$ depicted in Fig. 5 with the arithmetic sum target function f. It can be shown that the footprint size is $R_{C,f} = |I_C| + 1$ for any cut C, and thus

$$\text{min-cut}(\hat{\mathcal{N}}, f) = 1 \quad \text{[from (5)]}.$$

The result then follows immediately from Theorem V.1. ■

Remark V.3: In light of Theorem V.1, we compare the various lower bounds on the computing capacity of the network $\hat{\mathcal{N}}$ derived in Section III with the exact computing capacity. It can be shown that $\Pi(\hat{\mathcal{N}}) = 1$. If f is the arithmetic sum target function, then

$$\begin{aligned}&\mathcal{C}_{\mathrm{cod}}(\hat{\mathcal{N}}, f) \geq 1/2 \quad \text{[from Theorem III.5]}\\ &\mathcal{C}_{\mathrm{cod}}(\hat{\mathcal{N}}, f) \geq 1/\log_2 5 \quad \text{[from Theorem III.7]}\\ &\mathcal{C}_{\mathrm{cod}}(\hat{\mathcal{N}}, f) \geq 1/2 \quad \text{[from Theorem IV.2]}.\end{aligned}$$

Thus, this example demonstrates that the lower bounds obtained in Section III are not always tight and illustrates the combinatorial nature of the problem.

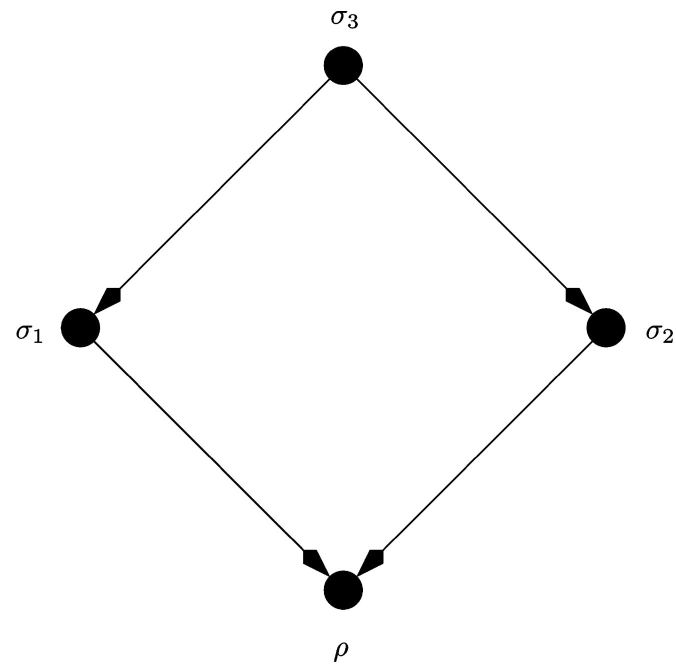

Fig. 5. Network $\hat{\mathcal{N}}$ has three binary sources, σ_1, σ_2, and σ_3 with $\mathcal{A} = \{0,1\}$ and the receiver ρ computes the arithmetic sum of the source messages.

VI. Conclusions

We examined the problem of network computing. The network coding problem is a special case when the function to be computed is the identity. We have focused on the case when a single receiver node computes a function of the source messages and have shown that while for the identity function the min-cut bound is known to be tight for all networks, a much richer set of cases arises when computing arbitrary functions, as the min-cut bound can range from being tight to arbitrarily loose. One key contribution of the paper is to show the theoretical breadth of the considered topic, which we hope will lead to further research. This work identifies target functions (most notably, the arithmetic sum function) for which the min-cut bound is not always tight (even up to a constant factor) and future work includes deriving more sophisticated bounds for these scenarios. Extensions to computing with multiple receiver nodes, each computing a (possibly different) function of the source messages, are of interest.

Appendix

Define the function

$$Q : \prod_{i=1}^{M} \{0,1\}^k \longrightarrow \{0,1,\ldots,M\}^k$$

as follows. For every $a = (a^{(1)}, a^{(2)}, \ldots, a^{(M)})$ such that each $a^{(i)} \in \{0,1\}^k$

$$Q(a)_j = \sum_{i=1}^{M} a_j^{(i)} \quad \text{for every } j \in \{1,2,\ldots,k\}. \qquad (38)$$

We extend Q for $X \subseteq \prod_{i=1}^{M} \{0,1\}^k$ by defining $Q(X) = \{Q(a) : a \in X\}$.

We now present Lemma VII.1. The proof uses Lemma VII.2, which is presented thereafter. We define the following function which is used in the next lemma. Let

$$\gamma(x) = \mathcal{H}^{-1}\left(\frac{1}{2}\left(1 - \frac{1}{x}\right)\right) \bigcap \left[0, \frac{1}{2}\right] \quad \text{for } x \geq 1 \tag{39}$$

where $\mathcal{H}^{-1}$ denotes the inverse of the binary entropy function $\mathcal{H}(x) = -x \log_2 x - (1-x)\log_2(1-x)$. Note that $\gamma(x)$ is an increasing function of x.

Lemma VII.1: If $\lim_{M\to\infty} \frac{L}{\log_2 M} = 0$, then $\lim_{M\to\infty} \mathcal{C}_{\text{cod}}(\mathcal{N}_{M,L}, f) = 1$.

Proof: For any M and L, a code that computes f with computing rate 1 is obtained by having each source σ_i send its message directly to the receiver on the edge (σ_i, ρ). Hence, $\mathcal{C}_{\text{cod}}(\mathcal{N}_{M,L}, f) \geq 1$. Now suppose that $\mathcal{N}_{M,L}$ has a (k,n) code that computes f with computing rate $k/n > 1$ and for each $i \in \{1, 2, \ldots, M\}$, let

$$g_i : \{0,1\}^k \longrightarrow \{0,1\}^n$$

be the corresponding encoding function on the edge (σ_i, ρ). Then for any $A_1, A_2, \ldots, A_M \subseteq \{0,1\}^k$, we have

$$\left(\prod_{i=1}^{M} |g_i(A_i)|\right) \cdot 2^{nL} \geq \left| Q\left(\prod_{i=1}^{M} A_i\right)\right|. \tag{40}$$

Each A_i represents a set of possible message vectors of source σ_i. The left-hand side of (40) is the maximum number of different possible instantiations of the information carried by the in-edges to the receiver ρ (i.e., $|g_i(A_i)|$ possible vectors on each edge (σ_i, ρ) and 2^{nL} possible vectors on the L parallel edges (σ_0, ρ)). The right-hand side of (40) is the number of distinct sum vectors that the receiver needs to discriminate, using the information carried by its in-edges.

For each $i \in \{1, 2, \ldots, M\}$, let $z_i \in \{0,1\}^n$ be such that $\left|g_i^{-1}(z_i)\right| \geq 2^{k-n}$ and choose $A_i = g_i^{-1}(z_i)$ for each i. Also, let $U^{(M)} = \prod_{i=1}^{M} A_i$. Then we have

$$|Q(U^{(M)})| \leq 2^{nL} \quad [\text{from } |g_i(A_i)| = 1 \text{ and } (40)]. \tag{41}$$

Thus (41) is a necessary condition for the existence of a (k,n) code that computes f in the network $\mathcal{N}_{M,L}$. Lemma VII.2 shows that[13]

$$|Q(U^{(M)})| \geq (M+1)^{\gamma(k/n)k} \tag{42}$$

where the function γ is defined in (39). Combining (41) and (42), any (k,n) code that computes f in the network $\mathcal{N}_{M,L}$ with rate $r = k/n > 1$ must satisfy

$$r\,\gamma(r)\log_2(M+1) \leq \frac{1}{n}\log_2 |Q(U^{(M)})| \leq L. \tag{43}$$

[13]One can compare this lower bound to the upper bound $|Q(U^{(M)})| \leq (M+1)^k$ which follows from [38].

From (43), we have

$$r\,\gamma(r) \leq \frac{L}{\log_2(M+1)}. \tag{44}$$

The quantity $r\,\gamma(r)$ is monotonic increasing from 0 to ∞ on the interval $[1, \infty)$ and the right hand side of (44) goes to zero as $M \to \infty$. Thus, the rate r can be forced to be arbitrarily close to 1 by making M sufficiently large, i.e., $\mathcal{C}_{\text{cod}}(\mathcal{N}_{M,L}, f) \leq 1$. In summary,

$$\lim_{M\longrightarrow\infty} \mathcal{C}_{\text{cod}}(\mathcal{N}_{M,L}, f) = 1.$$

■

Lemma VII.2: Let k, n, M be positive integers such that $k > n$. For each $i \in \{1, 2, \ldots, M\}$, let $A_i \subseteq \{0,1\}^k$ be such that $|A_i| \geq 2^{k-n}$ and let $U^{(M)} = \prod_{i=1}^{M} A_i$. Then

$$|Q(U^{(M)})| \geq (M+1)^{\gamma(k/n)k}.$$

Proof: The result follows from Lemmas VII.4 and VII.7. ■

The remainder of this Appendix is devoted to the proofs of lemmas used in the proof of Lemma VII.2. Before we proceed, we need to define some more notation. For every $j \in \{1, 2, \ldots, k\}$, define the map

$$h^{(j)} : \{0, 1, \ldots, M\}^k \longrightarrow \{0, 1, \ldots, M\}^k$$

by

$$(h^{(j)}(p))_i = \begin{cases} \max\{0, p_i - 1\}, & \text{if } i = j \\ p_i, & \text{otherwise.} \end{cases} \tag{45}$$

That is, the map $h^{(j)}$ subtracts one from the j-th component of the input vector (as long as the result is nonnegative) and leaves all the other components the same. For every $j \in \{1, 2, \ldots, k\}$, define a map

$$\hat{\phi}^{(j)} : 2^{\{0,1\}^k} \times \{0,1\}^k \longrightarrow \{0,1\}^k$$

by

$$\hat{\phi}^{(j)}(A, a) = \begin{cases} h^{(j)}(a), & \text{if } h^{(j)}(a) \notin A \\ a, & \text{otherwise} \end{cases} \tag{46}$$

for every $A \subseteq \{0,1\}^k$ and $a \in \{0,1\}^k$. Define

$$\phi^{(j)} : 2^{\{0,1\}^k} \longrightarrow 2^{\{0,1\}^k}$$

by

$$\phi^{(j)}(A) = \{\hat{\phi}^{(j)}(A, a) : a \in A\}. \tag{47}$$

Note that

$$|\phi^{(j)}(A)| = |A|. \tag{48}$$

A set A is said to be *invariant* under the map $\phi^{(j)}$ if the set is unchanged when $\phi^{(j)}$ is applied to it, in which case from (46) and (47) we would have that for each $a \in A$,

$$h^{(j)}(a) \in A. \tag{49}$$

Lemma VII.3: For any $A \subseteq \{0,1\}^k$ and all integers m and t such that $1 \leq m \leq t \leq k$, the set $\phi^{(t)}(\phi^{(t-1)}(\cdots \phi^{(1)}(A)))$ is invariant under the map $\phi^{(m)}$.

Proof: For any $A' \subseteq \{0,1\}^k$, we have

$$\phi^{(i)}(\phi^{(i)}(A')) = \phi^{(i)}(A') \quad \forall\, i \in \{1,2,\ldots,k\}. \tag{50}$$

The proof of the lemma is by induction on t. For the base case $t=1$, the proof is clear since $\phi^{(1)}(\phi^{(1)}(A)) = \phi^{(1)}(A)$ from (50). Now suppose the lemma is true for all $t < \tau$ (where $\tau \geq 2$) and let $t = \tau$ and let $B = \phi^{(\tau-1)}(\phi^{(\tau-2)}(\cdots \phi^{(1)}(A)))$. Since $\phi^{(\tau)}(\phi^{(\tau)}(B)) = \phi^{(\tau)}(B)$ from (50), the lemma is true when $m = t = \tau$. In the following arguments, we take $m < \tau$. From the induction hypothesis, B is invariant under the map $\phi^{(m)}$, i.e.,

$$\phi^{(m)}(B) = B. \tag{51}$$

Consider any vector $c \in \phi^{(\tau)}(B)$. From (49), we need to show that $h^{(m)}(c) \in \phi^{(\tau)}(B)$. We have the following cases:

$$\begin{aligned}
&c_\tau = 1: \\
&\qquad c, h^{(\tau)}(c) \in B \quad [\text{from } c_\tau = 1, c \in \phi^{(\tau)}(B)] && (52)\\
&\qquad h^{(m)}(c) \in B \quad [\text{from } (51), (52)] && (53)\\
&\qquad h^{(\tau)}(h^{(m)}(c)) = h^{(m)}(h^{(\tau)}(c)) \\
&\qquad\qquad \in B \quad [\text{from } (51),\ (52)] && (54)\\
&\qquad h^{(m)}(c) \in \phi^{(\tau)}(B) \quad [\text{from } (53), (54)].
\end{aligned}$$

$$\begin{aligned}
&c_\tau = 0: \\
&\exists b \in B \text{ with } h^{(\tau)}(b) = c \quad [\text{from } c_\tau = 0, c \in \phi^{(\tau)}(B)] && (55)\\
&\qquad h^{(m)}(b) \in B \quad [\text{from } (51), (55)] && (56)\\
&\qquad h^{(m)}(h^{(\tau)}(b)) = h^{(\tau)}(h^{(m)}(b)) \\
&\qquad\qquad \in \phi^{(\tau)}(B) \quad [\text{from } (56)] \\
&\qquad h^{(m)}(c) \in \phi^{(\tau)}(B) \quad [\text{from } (55), (57)]. && (57)
\end{aligned}$$

Thus, the lemma is true for $t = \tau$ and the induction argument is complete. ■

Let $A_1, A_2, \ldots, A_M \subseteq \{0,1\}^k$ be such that $|A_i| \geq 2^{k-n}$ for each i. Let $U^{(M)} = \prod_{i=1}^{M} A_i$ and extend the definition of $\phi^{(j)}$ in (47) to products by

$$\phi^{(j)}(U^{(M)}) = \prod_{i=1}^{M} \phi^{(j)}(A_i).$$

$U^{(M)}$ is said to be *invariant under* $\phi^{(j)}$ if

$$\phi^{(j)}(U^{(M)}) = U^{(M)}.$$

It can be verifed that $U^{(M)}$ is invariant under $\phi^{(j)}$ iff each A_i is invariant under $\phi^{(j)}$. For each $i \in \{1,2,\ldots,M\}$, let

$$B_i = \phi^{(k)}\left(\phi^{(k-1)}\left(\cdots \phi^{(1)}(A_i)\right)\right)$$

and from (48) note that

$$|B_i| = |A_i| \geq 2^{k-n}. \tag{58}$$

Let

$$V^{(M)} = \phi^{(k)}(\phi^{(k-1)}(\cdots \phi^{(1)}(U^{(M)}))) = \prod_{i=1}^{M} B_i$$

and recall the definition of the function Q in (38).

Lemma VII.4:

$$|Q(U^{(M)})| \geq |Q(V^{(M)})|.$$

Proof: We begin by showing that

$$|Q(U^{(M)})| \geq |Q(\phi^{(1)}(U^{(M)}))|. \tag{59}$$

For every $p \in \{0,1,\ldots,M\}^{k-1}$, let

$$\varphi(p) = \left\{r \in Q(U^{(M)}) : (r_2,\ldots,r_k) = p\right\}$$
$$\varphi_1(p) = \left\{s \in Q(\phi^{(1)}(U^{(M)})) : (s_2,\ldots,s_k) = p\right\}$$

and note that

$$Q(U^{(M)}) = \bigcup_{p \in \{0,1,\ldots,M\}^{k-1}} \varphi(p) \tag{60}$$

$$Q(\phi^{(1)}(U^{(M)})) = \bigcup_{p \in \{0,1,\ldots,M\}^{k-1}} \varphi_1(p) \tag{61}$$

where the two unions are in fact disjoint unions. We show that for every $p \in \{0,1,\ldots,M\}^{k-1}$

$$|\varphi(p)| \geq |\varphi_1(p)| \tag{62}$$

which by (60) and (61) implies (59).

If $|\varphi_1(p)| = 0$, then (62) is trivial. Now consider any $p \in \{0,1,\ldots,M\}^{k-1}$ such that $|\varphi_1(p)| \geq 1$ and let

$$K_p = \max\{i : (i, p_1, \ldots, p_{k-1}) \in \varphi_1(p)\}.$$

Then we have

$$|\varphi_1(p)| \leq K_p + 1. \tag{63}$$

Since $(K_p, p_1, \ldots, p_{k-1}) \in \varphi_1(p)$, there exists $(a^{(1)}, a^{(2)}, \ldots, a^{(M)}) \in U^{(M)}$ such that

$$\sum_{i=1}^{M} \hat{\phi}^{(1)}\left(A_i, a^{(i)}\right) = (K_p, p_1, \ldots, p_{k-1}). \tag{64}$$

Then from the definition of the map $\hat{\phi}^{(1)}$ in (46), there are K_p of the $a^{(i)}$'s from amongst $\{a^{(1)}, a^{(2)}, \ldots, a^{(M)}\}$ such that $a_1^{(i)} = 1$ and $\hat{\phi}^{(1)}\left(A_i, a^{(i)}\right) = a^{(i)}$. Let $I = \{i_1, i_2, \ldots, i_{K_p}\} \subseteq \{1, 2, \ldots, M\}$ be the index set for these vectors and let $\hat{a}^{(i)} = h^{(1)}(a^{(i)})$ for each $i \in I$. Then for each $i \in I$, we have

$$\begin{aligned} a^{(i)} &= \left(1, a_2^{(i)}, \ldots, a_k^{(i)}\right) \in A_i \\ \hat{a}^{(i)} &= \left(0, a_2^{(i)}, \ldots, a_k^{(i)}\right) \\ &\in A_i \;\; [\text{from } \phi^{(1)}\left(A_i, a^{(i)}\right) = a^{(i)}, (46)]. \end{aligned}$$

Let

$$R = \left\{ \sum_{i=1}^{M} b^{(i)} : \begin{array}{ll} b^{(i)} \in \{a^{(i)}, \hat{a}^{(i)}\} & \text{for } i \in I, \\ b^{(i)} = a^{(i)} & \text{for } i \notin I \end{array} \right\} \subseteq \varphi(p). \tag{65}$$

From (64) and (65), for every $r \in R$ we have

$$\begin{aligned} r_1 &\in \{0, 1, \ldots, |I|\}, \\ r_i &= p_i \quad \forall\, i \in \{2, 3, \ldots, k\} \end{aligned}$$

and thus

$$|R| = |I| + 1 = K_p + 1. \tag{66}$$

Hence, we have

$$\begin{aligned} |\varphi(p)| &\geq |R| \quad [\text{from } (65)] \\ &= K_p + 1 \quad [\text{from } (66)] \\ &\geq |\varphi_1(p)| \quad [\text{from } (63)] \end{aligned}$$

and then from (60) and (61), it follows that

$$|Q(U^{(M)})| \geq |Q(\phi^{(1)}(U^{(M)}))|.$$

For any $A \subseteq \{0,1\}^k$ and any $j \in \{1, 2, \ldots, k\}$, we know that $|\phi^{(j)}(A)| \subseteq \{0,1\}^k$. Thus, the same arguments as above can be repeated to show that

$$\begin{aligned} |Q(\phi^{(1)}(U^{(M)}))| &\geq |Q(\phi^{(2)}(\phi^{(1)}(U^{(M)})))| \\ &\geq |Q(\phi^{(3)}(\phi^{(2)}(\phi^{(1)}(U^{(M)}))))| \\ &\;\;\vdots \\ &\geq |Q(\phi^{(k)}(\phi^{(k-1)}(\cdots \phi^{(1)}(U^{(M)}))))| \\ &= |Q(V^{(M)})|. \end{aligned}$$

■

For any $s, r \in \mathbb{Z}^k$, we say that $s \leq r$ if $s_l \leq r_l$ for every $l \in \{1, 2, \ldots, k\}$.

Lemma VII.5: Let $p \in Q(V^{(M)})$. If $q \in \{0, 1, \ldots, M\}^k$ and $q \leq p$, then $q \in Q(V^{(M)})$.

Proof: Since $q \leq p$, it can be obtained by iteratively subtracting 1 from the components of p, i.e., there exist $t \geq 0$ and $i_1, i_2, \ldots, i_t \in \{1, 2, \ldots, k\}$ such that

$$q = h^{(i_1)}(h^{(i_2)}(\cdots(h^{(i_t)}(p)))).$$

Consider any $i \in \{1, 2, \ldots, k\}$. We show that $h^{(i)}(p) \in Q(V^{(M)})$, which implies by induction that $q \in Q(V^{(M)})$. If $p_i = 0$, then $h^{(i)}(p) = p$ and we are done. Suppose that $p_i > 0$. Since $p \in Q(V^{(M)})$, there exists $b^{(j)} \in B_j$ for every $j \in \{1, 2, \ldots, M\}$ such that

$$p = \sum_{j=1}^{M} b^{(j)}$$

and $b_i^{(m)} = 1$ for some $m \in \{1, 2, \ldots, M\}$. From Lemma VII.3, $V^{(M)}$ is invariant under $\phi^{(i)}$ and thus from (49), $h^{(i)}(b^{(m)}) \in B_m$ and

$$h^{(i)}(p) = \sum_{j=1}^{m-1} b^{(j)} + h^{(i)}(b^{(m)}) + \sum_{j=m+1}^{M} b^{(j)}$$

is an element of $Q(V^{(M)})$. ■

The lemma below is presented in [3] without proof, as the proof is straightforward.

Lemma VII.6: For all positive integers k, n, M, and $\delta \in (0,1)$

$$\min_{\substack{0 \leq m_i \leq M, \\ \sum_{i=1}^{k} m_i \geq \delta M k}} \prod_{i=1}^{k} (1 + m_i) \geq (M+1)^{\delta k}. \tag{67}$$

For any $a \in \{0,1\}^k$, let $|a|_H$ denote the Hamming weight of a, i.e., the number of nonzero components of a. The next lemma uses the function γ defined in (39).

Lemma VII.7:

$$|Q(V^{(M)})| \geq (M+1)^{\gamma(k/n)k}.$$

Proof: Let $\delta = \gamma(k/n)$. The number of distinct elements in $\{0,1\}^k$ with Hamming weight at most $\lfloor \delta k \rfloor$ equals

$$\begin{aligned} \sum_{j=0}^{\lfloor \delta k \rfloor} \binom{k}{j} &\leq 2^{k\mathcal{H}(\delta)} \quad [\text{from [24, p. 15, Theorem 1]}] \\ &= 2^{(k-n)/2} \quad [\text{from } (39)]. \end{aligned}$$

For each $i \in \{1, 2, \ldots, M\}$, $|B_i| \geq 2^{k-n}$ from (58) and hence there exists $b^{(i)} \in B_i$ such that $|b^{(i)}|_H \geq \delta k$. Let

$$p = \sum_{i=1}^{M} b^{(i)} \in Q(V^{(M)}).$$

It follows that $p_j \in \{0, 1, 2, \ldots, M\}$ for every $j \in \{1, 2, \ldots, k\}$, and

$$\sum_{j=1}^{k} p_j = \sum_{i=1}^{M} |b^{(i)}|_H \geq \delta M k. \tag{68}$$

The number of vectors q in $\{0, 1, \ldots, M\}^k$ such that $q \preceq p$ equals $\prod_{j=1}^{k} (1 + p_j)$, and from Lemma VII.5, each such vector is also in $Q(V^{(M)})$. Therefore

$$|Q(V^{(M)})| \geq \prod_{j=1}^{k}(1+p_j)$$
$$\geq (M+1)^{\delta k} \quad \text{[from (68) and Lemma VII.6]}.$$

Since $\delta = \gamma(k/n)$, the result follows. ■

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their comments which have greatly improved the presentation of this paper and would also like to thank Prof. Jacques Verstraete at UC San Diego for his help with the proof of Lemma VII.2.

REFERENCES

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 7, pp. 1204–1216, Jul. 2000.
[2] R. Appuswamy, M. Franceschetti, N. Karamchandani, and K. Zeger, "Network computing capacity for the reverse butterfly network," in *Proc. IEEE Int. Symp. Information Theory*, 2009, pp. 259–262.
[3] O. Ayaso, D. Shah, and M. Dahleh, "Lower bounds on information rates for distributed computation via noisy channels," in *Proc. 45th Allerton Conf. Computation, Communication and Control*, 2007.
[4] O. Ayaso, D. Shah, and M. Dahleh, "Counting bits for distributed function computation," in *Proc. IEEE Int. Symp. Information Theory*, 2008, pp. 652–656.
[5] F. Benezit, A. G. Dimakis, P. Thiran, and M. Vetterli, "Gossip along the way: Order-optimal consensus through randomized path averaging," in *Proc. 45th Allerton Conf. Computation, Communication and Control*, 2007.
[6] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, Jun. 2006.
[7] J. Cannons, R. Dougherty, C. Freiling, and K. Zeger, "Network routing capacity," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 777–788, Mar. 2006.
[8] P. Cuff, H. Su, and A. E. Gamal, "Cascade multiterminal source coding," in *Proc. IEEE Int. Symp. Information Theory*, 2009, pp. 1199–1203.
[9] A. G. Dimakis, A. D. Sarwate, and M. J. Wainwright, "Geographic gossip: Efficient aggregation for sensor networks," in *Proc. 5th Int. Conf. Information Processing in Sensor Networks*, 2006, pp. 69–76.
[10] V. Doshi, D. Shah, and M. Médard, "Source coding with distortion through graph coloring," in *Proc. IEEE Int. Symp. Information Theory*, 2007, pp. 1501–1505.
[11] V. Doshi, D. Shah, M. Médard, and S. Jaggi, "Graph coloring and conditional graph entropy," in *Proc. 40th Asilomar Conf. Signals, Systems and Computers*, 2006, pp. 2137–2141.
[12] V. Doshi, D. Shah, M. Médard, and S. Jaggi, "Distributed functional compression through graph coloring," in *Proc. Data Compression Conf.*, 2007, pp. 93–102.
[13] R. Dougherty, C. Freiling, and K. Zeger, "Unachievability of network coding capacity," *IEEE Trans. Inf. Theory & IEEE/ACM Trans. Netw. (Joint Issue)*, vol. 52, no. 6, pp. 2365–2372, Jun. 2006.
[14] C. Dutta, Y. Kanoria, D. Manjunath, and J. Radhakrishnan, "A tight lower bound for parity in noisy communication networks," in *Proc. 19th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2008, pp. 1056–1065.
[15] H. Feng, M. Effros, and S. Savari, "Functional source coding for networks with receiver side information," in *Proc. 42nd Allerton Conf. Computation, Communication and Control*, 2004, pp. 1419–1427.
[16] R. G. Gallager, "Finding parity in a simple broadcast network," *IEEE Trans. Inf. Theory*, vol. 34, no. , pp. 176–180, Mar. 1988.
[17] A. El Gamal, "Reliable communication of highly distributed information," in *Open Prob. in Commun. and Comput.*, T. M. Cover and B. Gopinath, Eds. New York: Springer-Verlag, 1987, pp. 60–62.
[18] A. Giridhar and P. R. Kumar, "Computing and communicating functions over sensor networks," *IEEE J. Select. Areas Commun.*, vol. 23, no. 4, pp. 755–764, Apr. 2005.
[19] N. Goyal, G. Kindler, and M. Saks, "Lower bounds for the noisy broadcast problem," *SIAM J. Comput.*, vol. 37, no. 6, pp. 1806–1841, Mar. 2008.
[20] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inf. Theory*, vol. 46, no. 3, pp. 388–404, Mar. 2000.
[21] G. H. Hardy and E. M. Wright, *An Introduction to the Theory of Numbers*, 5th ed. Oxford, U.K.: Oxford University Press, 1979.
[22] N. J. A. Harvey, R. Kleinberg, and A. Rasala Lehman, "On the capacity of information networks," *IEEE Trans. Inf. Theory & IEEE/ACM Trans. Netw. (Joint Issue)*, vol. 52, no. 6, pp. 2345–2364, Jun. 2006.
[23] N. J. A. Harvey, R. D. Kleinberg, and A. Rasala Lehman, "Comparing network coding with multicommodity flow for the k-pairs communication problem," *M.I.T. LCS, Tech. Rep. 964*, 2004.
[24] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *J. Amer. Statist. Assoc.*, vol. 58, no. 301, pp. 13–30, Mar. 1963.
[25] K. Jain, M. Mahdian, and M. R. Salavatipour, "Packing steiner trees," in *Proc. 14th Ann. ACM-SIAM Symp. Discrete Algorithms*, 2003, pp. 266–274.
[26] N. Karamchandani, R. Appuswamy, and M. Franceschetti, "Distributed computation of symmetric functions with binary inputs," in *Proc. IEEE Information Theory Workshop*, 2009, pp. 76–80.
[27] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *Proc. 44th Ann. IEEE Symp. Foundations of Computer Science*, 2003, pp. 482–491.
[28] J. Körner and K. Marton, "How to encode the modulo-two sum of binary sources," *IEEE Trans. Inf. Theory*, vol. IT-25, pp. 29–221, Mar. 1979.
[29] H. Kowshik and P. R. Kumar, "Zero-error function computation in sensor networks," in *Proc. IEEE Conf. Decision and Control*, 2009, pp. 3787–3792.
[30] E. Kushilevitz and N. Nisan, *Communication Complexity*. Cambridge, U.K.: Cambridge University Press, 1997.
[31] M. Langberg and A. Ramamoorthy, "Communicating the sum of sources in a 3-sources/3-terminals network," in *Proc. IEEE Int. Symp. Information Theory*, 2009, pp. 2121–2125.
[32] A. Rasala Lehman and E. Lehman, "Complexity classification of network information flow problems," in *Proc. 15th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2003, pp. 142–150.
[33] T. Leighton and S. Rao, "Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms," *J. ACM*, vol. 46, no. 6, pp. 787–832, Nov. 1999.
[34] N. Ma and P. Ishwar, "Two-terminal distributed source coding with alternating messages for function computation," in *Proc. IEEE Int. Symp. Information Theory*, 2008, pp. 51–55.
[35] N. Ma, P. Ishwar, and P. Gupta, "Information-theoretic bounds for multiround function computation in collocated networks," in *Proc. IEEE Int. Symp. Information Theory*, 2009, pp. 2306–2310.
[36] D. Mosk-Aoyama and D. Shah, "Fast distributed algorithms for computing separable functions," *IEEE Trans. Inf. Theory*, vol. 54, no. 7, pp. 2997–3007, Jul. 2008.
[37] B. Nazer and M. Gastpar, "Computing over multiple-access channels," *IEEE Trans. Inf. Theory*, vol. 53, no. 10, pp. 3498–3516, Oct. 2007.
[38] C. K. Ngai and R. W. Yeung, "Network coding gain of combination networks," in *Proc. IEEE Information Theory Workshop*, 2004, pp. 283–287.
[39] A. Orlitsky and J. R. Roche, "Coding for computing," *IEEE Trans. Inf. Theory*, vol. 47, no. 3, pp. 903–917, Mar. 2001.
[40] B. K. Rai and B. K. Dey, "Feasible alphabets for communicating the sum of sources over a network," in *Proc. IEEE Int. Symp. Information Theory*, 2009, pp. 1353–1357.
[41] B. K. Rai, B. K. Dey, and S. Shenvi, "Some bounds on the capacity of communicating the sum of sources," in *Proc. ITW 2010*, Cairo, Egypt, 2010.
[42] A. Ramamoorthy, "Communicating the sum of sources over a network," in *Proc. IEEE Int. Symp. Information Theory*, 2008, pp. 1646–1650.
[43] S. Subramanian, P. Gupta, and S. Shakkottai, "Scaling bounds for function computation over large networks," in *Proc. IEEE Int. Symp. Information Theory*, 2007, pp. 136–140.
[44] V. V. Vazirani, *Approximation Algorithms*, 1st ed. New York: Springer, 2004.
[45] D. B. West, *Introduction to Graph Theory*. Upper Saddle River, NJ: Prentice-Hall, 2001.
[46] H. Witsenhausen, "The zero-error side information problem and chromatic numbers," *IEEE Trans. Inf. Theory*, vol. IT-22, pp. 592–593, Sep. 1976.
[47] H. Yamamoto, "Wyner—Ziv theory for a general function of the correlated sources," *IEEE Trans. Inf. Theory*, vol. IT-28, pp. 803–807, Sep. 1982.

[48] A. C. Yao, "Some complexity questions related to distributive computing," in *Proc. 11th Annu. ACM Symp. Theory of Computing*, 1979, pp. 209–213.
[49] R. W. Yeung, *A First Course in Information Theory*. : Springer, 2002.
[50] L. Ying, R. Srikant, and G. E. Dullerud, "Distributed symmetric function computation in noisy wireless sensor networks," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4826–4833, Dec. 2007.

Rathinakumar Appuswamy (S'05) received the B.Tech. degree from Anna University, Chennai, India, and the M.Tech. degree from the Indian Institute of Technology, Kanpur, India, both in electrical engineering in 2004, and the M.A. degree in mathematics from the University of California, San Diego, in 2008.

He is currently a doctoral student at the University of California, San Diego, where he is a member of the Information and Coding Laboratory, as well as the Advanced Network Sciences Group. His research interests include network coding, communication for computing, and network information theory.

Massimo Franceschetti (M'98) received the Laurea degree (*magna cum laude*) in computer engineering from the University of Naples, Naples, Italy, in 1997, and the M.S. and Ph.D. degrees in electrical engineering from the California Institute of Technology, Pasadena, CA, in 1999, and 2003, respectively.

He is an Associate Professor in the Department of Electrical and Computer Engineering, University of California at San Diego (UCSD). Before joining UCSD, he was a postdoctoral scholar at the University of California at Berkeley for two years. He has held visiting positions at the Vrije Universiteit Amsterdam, the Ecole Polytechnique Federale de Lausanne, and the University of Trento. His research interests are in communication systems theory and include random networks, wave propagation in random media, wireless communication, and control over networks.

Dr. Franceschetti is an Associate Editor for Communication Networks of the IEEE TRANSACTIONS ON INFORMATION THEORY (2009–2012) and has served as Guest Editor for two issues of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATION. He was awarded the C. H. Wilts Prize in 2003 for best doctoral thesis in electrical engineering at Caltech; the S.A. Schelkunoff Award in 2005 for best paper in the IEEE TRANSACTIONS ON ANTENNAS AND PROPAGATION; a National Science Foundation (NSF) CAREER award in 2006, an ONR Young Investigator Award in 2007; and the IEEE Communications Society Best Tutorial Paper Award in 2010.

Nikhil Karamchandani (S'05) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Bombay, in 2005 and the M.S. degree in electrical engineering in 2007 from the University of California at San Diego (UCSD), where he is currently pursuing the Ph.D. degree in the Department of Electrical and Computer Engineering.

His research interests are in communication theory and include network coding, information theory, and random networks.

Mr. Karamchandani received the California Institute for Telecommunications and Information Technology (CalIT2) Fellowship in 2005.

Kenneth Zeger (S'85–M'90–SM'95–F'00) was born in Boston, MA, in 1963. He received the B.S. and M.S. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, in 1984, and both the M.A. degree in mathematics in 1989 and the Ph.D. degree in electrical engineering in 1990 from the University of California at Santa Barbara.

He was an Assistant Professor of Electrical Engineering at the University of Hawaii from 1990 to 1992. He was with the Department of Electrical and Computer Engineering and the Coordinated Science Laboratory, both at the University of Illinois at Urbana-Champaign, first as an Assistant Professor (1992 to 1995) and then as an Associate Professor (1995 to 1996). He has been with the Department of Electrical and Computer Engineering, University of California at San Diego, since 1996, as an Associate Professor (1996 to 1998) and currently as a Professor (since 1998).

Dr. Zeger received a National Science Foundation (NSF) Presidential Young Investigator Award in 1991. He served as Associate Editor At-Large for the IEEE TRANSACTIONS ON INFORMATION THEORY during 1995–1998, and as a member of the Board of Governors of the IEEE Information Theory Society during 1998–2000, 2005–2007, and 2008–2010.