Electronic Colloquium on Computational Complexity, Report No. 33 (2003)

Bounds on Linear Codes for Network Multicast

Meir Feder, Dana Ron, and Ami Tavory[†]
Dept. of EE - Systems
Tel-Aviv University,
Tel Aviv, Israel.
E-mail: {meir, danar, atavory}@eng.tau.ac.il

Abstract

Traditionally, communication networks are composed of *routing* nodes, which relay and duplicate data. Work in recent years has shown that for the case of multicast, an improvement in both rate and code-construction complexity can be gained by replacing these routing nodes by *linear coding* nodes. These nodes transmit linear combinations of the inputs transmitted to them. In this work, we deal with bounds on the alphabet size of linear codes for multicast. We show both lower and upper bounds as a function of sink-set size and graph topology. We show that these bounds apply, as well, to a special case of multicast, *static broadcast*. We also show how node-memory addition can increase the effective alphabet-size available for coding.

Index Terms

network information-flow, multicast, alphabet size, bounds, static broadcast.

I. Introduction

In this paper, we consider the field size necessary for communicating information, by linear codes, from a single source to multiple sinks, across a network. Work in recent years (see [1], [6], [8], and [9], among others) has shown that for the case of network multicast, linear coding within a network can increase transfer rate and decrease code construction complexity.

We show that known bounds on *MDS* (maximum distance separable) codes can be applied to bound the alphabet-size necessary and sufficient for linear multicast-codes. We then show that these bounds also pertain to a case less considered in previous network-multicast work, *static broadcast*, wherein the source transmits the same set of data to sinks at different rates. Finally, we show how the addition of a small amount of memory to internal nodes can be used to increase the effective alphabet-size available for coding.

A. Paper Layout

We continue the introduction with definitions and notations in Subsection I-B, a brief review of the transmission scheme in Subsection I-C, and a short review of related work in Subsection I-D. In Section II we show an alphabet-size lower-bound. In Section III we show a field size upper bound. In Section IV we discuss *static broadcast*. In Section V we describe how node memory can increase the effective field-size available for coding. We conclude in Section VI.

B. Definitions and Notations.

We consider a network over an acyclic, directed, graph $G = (V, E)$, where parallel edges are allowed. We denote the number of nodes in G by, $n = |V|$. When there is no ambiguity, we denote an edge between nodes $u, v \in V$ by (u, v). For any node $v \in V$, we use $\mathsf{E}^-(v)$ and $\mathsf{E}^+(v)$ to denote the set of edges reaching and leaving node v, respectively. For any edge $e \in E$, we use $\mathsf{v}^-(e)$ and $\mathsf{v}^+(e)$ to denote the tail and head of e, respectively.

Node $s \in V$ is the *source* node, $T \subseteq V$ is the set of *sink* nodes, $d = |T|$ is the number of sink nodes, and h is the minimum, taken over all $t \in T$, of the size of the minimum cut separating s from t. The capacity of a link (*i.e.*, edge) $e \in E$, is a single symbol from a field $\mathcal{F}$, with size $q = |\mathcal{F}|$, and the value carried by the link is denoted by $y(e)$. A process at node s observes a random process X with entropy[1] $H(X) = h \cdot \log(q)$ per time unit, and wishes to transmit the information of this process to all $t \in T$. In some of the work, we explicitly consider a *sequential* setting, in which, for some $N \gg \max\{n, q, h\}$, s transmits the values which X attains in N consecutive time units.

† Also at IBM's Haifa Research Labs.

[1]We use $H(\cdot)$ and $\log(\cdot)$ to denote the binary entropy function and logarithm to base 2, respectively.

ISSN 1433-8092

C. The Transmission Scheme

The transmission scheme in linear multicast was described in [6], [8], and [9], and we repeat the description here in brief.

We introduce a virtual vertex $s' \notin V$, and h parallel virtual edges from s' to s, $e_1^{s'}, \ldots, e_h^{s'}$ (this is done for notational convenience throughout the paper). The values carried by these h edges, $y(e_1^{s'}), \ldots, y(e_h^{s'})$, are any h symbols from $\mathcal{F}$ which describe the process X. The code is determined by means of a set of auxiliary functions

$$m_e \; : \; \mathsf{E}^- \left(\mathsf{v}^- (e)\right) \to \mathcal{F}, \;\; (e \in E). \tag{1}$$

The determination of m_e will be described in Section III. For any edge $e \in E$ s.t. $y(e')$ has been determined for all $e' \in \mathsf{E}^- (\mathsf{v}^- (e))$, the value carried by e is

$$y(e) = \sum_{e' \in \mathsf{E}^- (\mathsf{v}^- (e))} m_e(e') y(e'). \tag{2}$$

We call a code *good*, if for any $t \in T$, there is a subset $E_t \subseteq \mathsf{E}^- (t)$ s.t. $y(e_1^{s'}), \ldots, y(e_h^{s'})$ can be reconstructed from $\{y(e) | e \in E_t\}$

D. Related Work

Ahlswede *et. al.* [1] have shown that a source can multicast information at a rate approaching h to all sinks, as the symbol size approaches infinity. Li *et. al.* [9] constructively showed that linear coding can be used for multicast with rate h and finite symbol-size. Koeter and Medard [6] showed, through an algebraic framework for network-coding which they developed, that a finite field of size[2] $O(d \cdot h)$ is sufficient for rate h multicast, and showed how to verify the validity of a given code. Sanders et al. [8] showed the first polynomial-time algorithms for constructing linear codes for multicast, and showed that a field of size $O(d)$ is sufficient for this. Our work is an extension of [8], and utilizes many of its ideas and notations.

II. Alphabet-Size Lower-Bound

In this section we prove a lower bound on the alphabet size:

Theorem 1: For some graphs, the alphabet size must obey:

$$q \geq 2 \cdot \sqrt{d} \cdot (1 - o(1)). \tag{3}$$

We prove the lower bound by constructing the graph G, which is shown in Figure 1. In this graph, let m by a number which we will later specify, and let $M = \binom{m}{2}$.

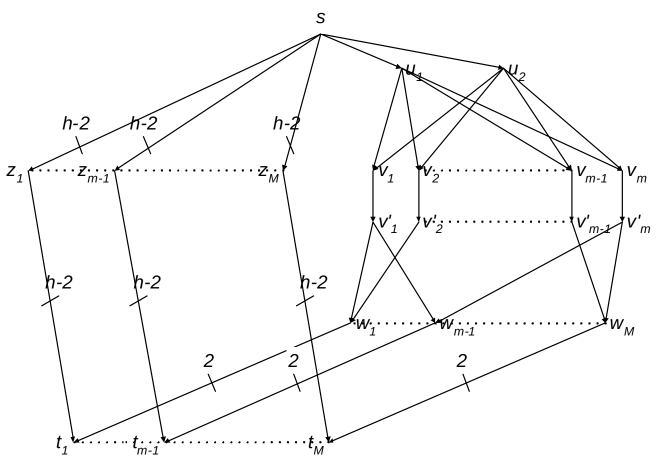

Fig. 1. Graph construction for the alphabet-size lower-bound.

The graph is constructed as follows. Let[3] $i \in [m], j \in [M]$, and $k \in [2]$, be arbitrary indices. The source s is connected to nodes u_1 and u_2 via a single link to each. Both u_1 and u_2 are connected to all nodes v_i. Each node v_i is connected to a corresponding node v_i' via a single link. For each pair of nodes in $\{v_1', \ldots, v_m'\}$, there is a node w_j, with links from the nodes of the pair to w_j. Each node w_j is connected to a sink t_j via 2 links (or equivalently, via a link of capacity 2). The source is

[2]As part of an expression containing a variable x, we use $O(x)$ (respectively $o(x)$) to denote a function $f(x)$ s.t. $\lim_{x\to\infty} f(x) \lneq \infty$ ($\lim_{x\to\infty} f(x) = 0$).

[3]For any ℓ, we use $[\ell]$ to denote $\{1, \ldots, \ell\}$.

connected to each of z_j via $h-2$ links (or equivalently, via a link of capacity $h-2$). Each node z_j is connected to t_j via $h-2$ links (or equivalently, via a link of capacity $h-2$).

Clearly, this graph is cycle free, and there is a min-cut of capacity h between s and each of t_j. By the Max-Flow Min-Cut theorem, h symbols can be sent from s to any sink t_j individually. It follows from [1], [6], [8], [9] that s can multicast h symbols of information to all t_j simultaneously.

We define some random processes observed at graph nodes:

$$\begin{array}{ll} Z_j = \left\{ y(e) \mid e \in \mathsf{E}^-(z_j) \right\}, & j \in [M] \\ U_k = \left\{ y((s, u_k)) \right\}, & k = 1, 2 \\ V'_i = \left\{ y(e) \mid e \in \mathsf{E}^-(v'_i) \right\}, & i \in [m] \\ W_j = \left\{ y(e) \mid e \in \mathsf{E}^-(w_j) \right\}, & j \in [M] \\ T_j = \left\{ y(e) \mid e \in \mathsf{E}^-(t_j) \right\}, & j \in [M] \end{array} \tag{4}$$

Lemma 1: For $j \in [M]$, the entropies of the processes defined in (4), satisfy the following:

$$H(U_1, U_2 | W_j, Z_j) = 0 \tag{5}$$
$$H(W_j, Z_j) = h \cdot \log(q) \tag{6}$$
$$H(U_1, U_2, Z_j) = h \cdot \log(q) \tag{7}$$
$$H(U_1, U_2 | Z_j) = 2 \cdot \log(q). \tag{8}$$

Proof: From the Max-Flow Min-Cut theorem, we clearly have that

$$H(U_k) \leq \log(q), \tag{9}$$
$$H(Z_i) \leq (h-2) \cdot \log(q). \tag{10}$$

Also,

$$\begin{aligned} h \cdot \log(q) &\overset{(a)}{=} H(T_j) \\ &\overset{(b)}{\leq} H(Z_j, W_j) \\ &\leq H(Z_j) + H(W_j) \\ &\overset{(c)}{\leq} H(Z_j) + H(U_1, U_2) \\ &\leq H(Z_j) + H(U_1) + H(U_2) \\ &\overset{(d)}{\leq} h \cdot \log(q), \end{aligned} \tag{11}$$

where (a) follows from the fact that the process at t_j can reconstruct the information of X, (b) and (c) follow from the data-processing inequality ([3]), and (d) follows from the Max-Flow Min-Cut theorem.

It follows from 11) that

$$H(Z_j) + H(U_1) + H(U_2) = h \cdot \log(q). \tag{12}$$

We obtain (6) from (9), (10), and (12). In a similar manner, we can obtain (7). We obtain (8) from (9), (10), and (11). To obtain (5), note that

$$\begin{aligned} & H(U_1, U_2 | W_j, Z_j) + H(Z_j, W_j) \\ & \quad = H(W_j, Z_j, U_1, U_2) \\ & \quad = H(Z_j, U_1, U_2) + H(W_j | Z_j, U_1, U_2) \\ & \quad = H(Z_j, U_1, U_2) \end{aligned} \tag{13}$$

$$\overset{(a)}{\Rightarrow}$$

$$H(U_1, U_2 | W_j, Z_j) = 0 \tag{14}$$

where (a) follows from (6), and (7). ■

We can now prove Theorem 1:

Proof: Let

$$\begin{array}{l} E^z = \{ e \mid \mathsf{v}^-(e) = s \wedge \mathsf{v}^+(e) = z_j \} \\ E^{v'} = \{ e \mid \mathsf{v}^-(e) = v_i \wedge \mathsf{v}^+(e) = v'_i \} \end{array} . \tag{15}$$

For any fixed values of $y(e)$, $e \in E^z$, it follows from (8) that $y(s,u_1), y(s,u_2)$ obtain all values from $\mathcal{F} \times \mathcal{F}$. It also follows from (5) that given any $e_1, e_2 \in E^V$, $y(e_1), y(e_2)$ are sufficient for determining $y(s,u_1), y(s,u_2)$. Thus, $y(e')$, $e' \in E^{v'}$, is effectively an *MDS* (maximal distance separable) code. By known bounds on MDS codes ([5]),

$$m = \left|E^{v'}\right| \le q \cdot (1 + o(1)), \tag{16}$$

and so

$$d = M \le \binom{q \cdot (1 + o(1))}{2}. \tag{17}$$

■

III. Alphabet-Size Upper-Bound

An upper bound as a function of terminal-set size was shown in [8]. As an extension of this work, we show in this section that for some graph topologies, the alphabet size is somewhat smaller.

Consider the three networks depicted in Figure 2. In terms of coding, network (a) requires coding, whereas networks (b) and (c) do not. In terms of flows from s to t_1 and t_2, the flow paths in (a) "clash" in some sense, whereas those in (b) and (c) do not; the flow paths in (a) are the only ones in which the edges of two flow paths merge into a single edge.

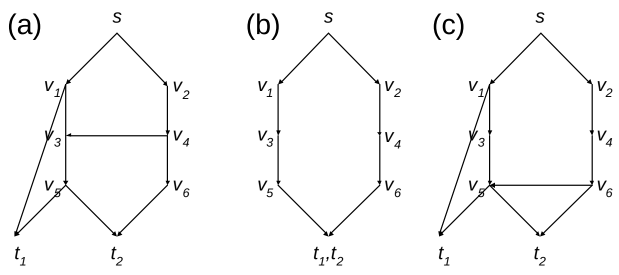

Fig. 2. Graph constructions illustrating flow clashes.

This section is organized as follows. In Subsection III-A we repeat, in brief, the determination of the auxiliary functions m_e using *global coding vectors* [8]. In Subsection III-B we define the *clash set* and *minimal clash set*, two sets which are relevant for the determination of m_e. This subsection is relevant for Section V as well. In Subsection III-C we show how the minimal clash set can be used for determining m_e, a consequence of which is an upper bound on the alphabet size.

A. Global Coding Vectors

Let $G_t^f = \left(V_t^f, E_t^f\right)$ $(t \in T)$ be subgraphs of G, each composed of h edge-disjoint paths from s' to t. A subgraph G_t^f can be constructed from G by running any appropriate flow algorithm (see [2]). For any $e \in E_t^f \setminus \{e_1^{s'}, \ldots, e_h^{s'}\}$, let $f_t^-(e)$ denote the immediate-predecessor edge of e on the path in G_t^f containing e. As an extension, for any $e \in E_t^f \setminus \{e_1^{s'}, \ldots, e_h^{s'}\}$ and any $T' \subseteq T$, let $f_{T'}^-(e)$ denote $\bigcup_{t' \in T'} f_{t'}^-(e)$.

We associate with each edge $e \in \left(\bigcup_{t \in T} E_{t_i}^f\right)$ a *global coding vector*[4] $\mathbf{b}(e) \in \mathcal{F}^h$. We first set the global coding vectors for $e_k^{s'}$, $(k \in [h])$, as

$$\mathbf{b}(e_k^{s'}) = [0^{k-1}, 1, 0^{h-k}]. \tag{18}$$

The global coding vector of any other edge $e \in \bigcup_{i=1,\ldots,d} G_{t_i}^f$ is set so that it satisfies the invariant

$$\mathbf{b}(e) = \sum_{e' \in \mathsf{E}^-(\mathsf{v}^-(e))} m_e(e') \cdot \mathbf{b}(e'). \tag{19}$$

Assume the algorithm terminates after ℓ steps. We generate $\ell \cdot d$ edge-sets, $C_{t_1}^1, \ldots, C_{t_d}^1, \ldots\ldots, C_{t_1}^\ell, \ldots, C_{t_d}^\ell$. Let $j \in [\ell]$ be an arbitrary step of the algorithm, and let $i \in [d]$ be the index of an arbitrary terminal node. Each $C_{t_i}^j$ is composed of h edges, one from each path between s and t_i in $G_{t_i}^f$, Initially, we set $C_{t_1}^1 = \cdots = C_{t_d}^1 = \{e_1^{s'}, \ldots, e_h^{s'}\}$. At the termination

[4]We use $\underline{x}$ to denote a *row* vector, and $\underline{x}^+$ to denote its transpose.

of the algorithm, we have $C^m_{t_i} \subseteq \mathsf{E}^-(t_i)$. At step j an edge e_j is chosen so that, for some i, $e_j \in G^f_{t_i}$ and $f^-_{t_i}(e_j) \in C^{j-1}_{t_i}$. Then, m_e is set according to an invariant shown below (see Lemma 2), and through it, $\mathbf{b}(e)$. Step j is completed by assigning

$$C^j_t = \begin{cases} C^{j-1}_t, & f^-_t(e_j) \notin C^{j-1}_t \\ \left(C^{j-1}_t \setminus f^-_t(e_j)\right) \dot{\bigcup} \{e_j\}, & f^-_t(e_j) \in C^{j-1}_t \end{cases}. \tag{20}$$

The following lemma was proved in [8].

Lemma 2: Let m_e be set with the invariant that for $j \in [\ell]$, and $i \in [d]$,

$$Rank\left(\left\{\mathbf{b}(e) \mid e \in C^j_{t_i}\right\}\right) = h. \tag{21}$$

Then the code is good.

B. Clash Sets

In this subsection, we define two sets which are relevant for the determination of m_{e_j}, where e_j is the edge chosen in step j.

Definition 1: (Clash Set) The *clash set* of $e_j \in E$, is

$$T'(e_j) = T \bigcap \left\{t \mid \exists_j e_j \in G^f_t\right\}. \tag{22}$$

Since we assume that G is an acyclic graph, there is a natural order that can be imposed on any edge $e \in E$. It follows that e_j, can be chosen so that it satisifes

$$\forall_{t' \in T'(e_j)} f^-_{t'}(e_j) \in C^{j-1}_{t'}, \tag{23}$$

i.e., that ... . In the remainder of the section, we assume, without loss of generality, that (23) holds.

Definition 2: (Minimum Clash-Set) Let $e_j \in E$ and $T'(e_j) \subseteq T$ be be an edge and its clash set, respectively. The *minimum clash-set* of e_j, $T''(e_j)$, is a minimum subset $T''(e_j) \subseteq T'(e_j)$ satisfying

$$\forall_{t' \in T'(e_j) \setminus T''(e_j)} \exists_{t'' \in T''(e_j)} \forall_{e' \in C^j_{t'} \setminus f^-_{t'}(e_j)} \tag{24}$$
$$Rank\left(\{\mathbf{b}(e')\} \bigcup \left\{\mathbf{b}(e'') \mid e'' \in C^j_{t''} \setminus f^-_{t''}(e_j)\right\}\right) = h-1.$$

Figure 3 shows an example of a clash set and a minimum clash set. In the graph shown in part (a) of the figure, let $e = (w_{1,2,3}, w'_{1,2,3})$, and $e'_i = (u_i, u'_i)$, for $i = 1, 2, 3$. As is shown in part (b) of the figure, the clash set of e is $T' = \{t_1, t_2, t_3\}$. The minimum clash set of E is $T'' = \{t_1, t_2\}$, since $\mathbf{b}(e'_1)$ and $\mathbf{b}(e'_2)$ are not linearly dependent, but $\mathbf{b}(e'_3)$ is linearly dependent on $\mathbf{b}(e'_1)$.

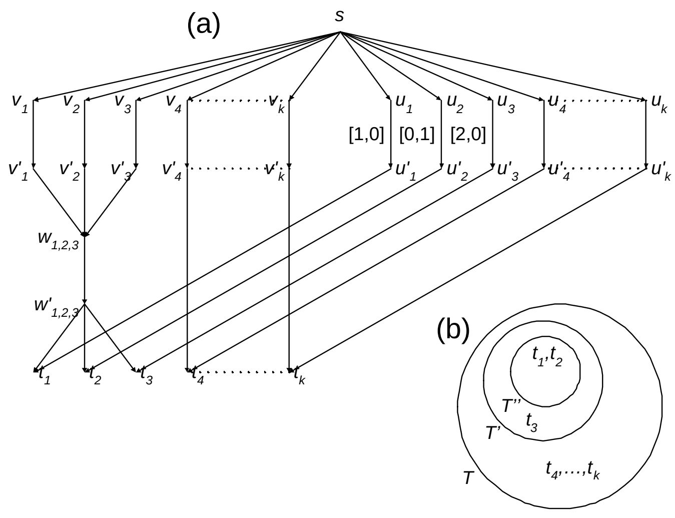

Fig. 3. Example of a clash set and a minimum clash set.

The following lemma follows from a straightforward counting argument.

Lemma 3: At step j, the size of any minimum clash set is at most

$$\binom{\left|\left\{\mathbf{b}(e) \mid \exists_{t \in T} e \in C_t^j\right\}\right|}{h-1}. \tag{25}$$

C. Setting Global Coding Vectors Using Minimal Clash Sets

In this subsection, we show how the minimum clash set determines the alphabet size needed for setting m_{e_j}, where e_j is the edge chosen in step j.

Lemma 4: Let $T''(e_j)$ be the minimum clash set of e_j. If m_{e_j} is set so that

$$\forall_{t'' \in T''(e_j)} \quad Rank\left(\{\mathbf{b}(e_j)\} \bigcup \left\{\mathbf{b}(e'') \mid e'' \in C_{t''}^j \setminus f_{t''}^-(e_j)\right\}\right) = h, \tag{26}$$

then the code is good.

Proof: We prove, by induction on $j = 1, \ldots, \ell$, that for $i \in [i]$, (21) holds. By Lemma 2, it will follow that the code is good.

The induction base, *i.e.*, $j = 1$, holds trivially by the determination of the global coding vectors between s' and s. Assume that (21) holds for $j-1$, and that $\mathbf{b}(e_j)$ is set so that (26) is satisfied. Clearly then, $\mathbf{b}(e_j) \neq \underline{0}$. For any $t'' \in T''(e_j)$, (21) holds by definition. For any $t' \in T'(e_j)$, there is a $t'' \in T''(e_j)$, and a matrix $B_{e''}^{e'}$, s.t.

$$[\mathbf{b}(e_1'), \ldots, \mathbf{b}(e_{h-1}')] = [\mathbf{b}(e_1''), \ldots, \mathbf{b}(e_{h-1}'')] \cdot B_{e''}^{e'}, \tag{27}$$

for $e_1', \ldots, e_{h-1}' = C_{t'}^j \setminus f_{t'}^-(e_j)$, and $e_1'', \ldots, e_{h-1}'' = C_{t''}^j \setminus f_{t''}^-(e_j)$. It follows that $\mathbf{b}(e_j) \neq \underline{0}$ cannot be a linear combination of $\mathbf{b}(e_1'), \ldots, \mathbf{b}(e_{h-1}')$, since, by (27) it would be then also be a linear combination of $\mathbf{b}(e_1''), \ldots, \mathbf{b}(e_{h-1}'')$, which would contradict (26). ■

The following is a slight variation of a lemma in [8]:
Lemma 5: If at step j of the algorithm, $|T''(e_j)| \leq q$, then m_{e_j} can be set so that (26) is satisfied.

Using Lemma 5, we find an upper bound on the alphabet size as a function of the number of "flow-path clashes":
Theorem 2: Let m be the number of "clashes", once the flows are found, *i.e.*,

$$m = \left|[\ell] \bigcap \{j \mid |T'(e_j)| \geq 2\}\right|. \tag{28}$$

Then the sufficient alphabet-size is bounded by

$$q_{\max} = \begin{cases} 2, & h = 1 \\ \min\{d, \binom{h+m}{h-1}\}, & h \geq 2 \end{cases} \tag{29}$$

Proof: The case of $h = 1$ is trivial, and so we consider only $h \geq 2$. the upper bound of d was proven in [8]. Prior to the first clash, $\left|\left\{\mathbf{b}(e) \mid \exists_{t \in T} e \in C_t^1\right\}\right| = h$, and each clash increments the size of this set by at most 1. The theorem now follows by Lemmas 3 and 5. ■

IV. Static Broadcast

The previous work, mentioned in Section I, has focused on the sender transmitting at rate equal to the minimum of the set of min-cuts separating s from all $t \in T$. In this section we discuss *static broadcast* [4]. Let $i \in [d]$ be an arbitrary index of a sink. Assume that the minimum of min cuts from s to $t_1, \ldots, t_d$, is $h_1, \ldots, h_d$, respectively. Regardless of the difference in rates, s wishes to broadcast a static (*i.e.*, fixed, identical) set of data to all $t \in T$.

Intuitively, if h_j, h_k are s.t. $h_j \gneqq h_k$, then the sink t_j should be able to receive and decode the information before the sink t_k. This might necessitate encoding the source, in some cases. Consider cases (a) and (b) in Figure 4. In case (a), each sink can indeed receive the same number of symbols as the min cut from s to it. In case (b), if the same symbols are sent to t_1 and t_3, then t_2 does not receive information at the rate indicated by the min-cut from s to it.

For the case of a sequential transmission, we show that there is an asymptotically optimal encoding. Assume that the values that X attains between times 1 and $N \gg h_{\max}$ are to be transmitted to all $t \in T$. Let $h_{\max} = \max_i h_i$ be the maximum of min cuts between s and any t_i, $c \gneqq 1$ be any number, and

$$\alpha(N, c) = N + c \cdot \log_q(N)\left(\ln\left(c \cdot \log_q(N)\right) + 1\right). \tag{30}$$

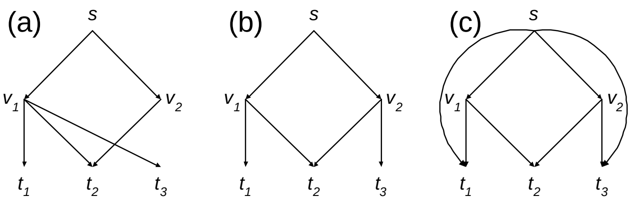

Fig. 4. Static broadcast possibilities.

Theorem 3: There is a transmission scheme in which the source transmits between times 1 and $\frac{\alpha(N,c)}{h}$, s.t. with probability at least $1-\frac{d}{N^{c-1}}$, each t_i can reconstruct the N values of X at rate

$$\frac{N}{\frac{\alpha(N,c)}{h_i}} \overset{N\to\infty}{\longrightarrow} h_i. \tag{31}$$

For the proof of Theorem 3, we first compose a graph $G'=(V',E')$ from $G=(V,E)$, by adding edges from s to $t\in T$, so that the minimum of the set of min-cuts separating s from any $t\in T$ is h. That is,

$$V' = V \tag{32}$$

$$E' = E \dot{\bigcup} \left(\dot{\bigcup_i} \left\{e_1^{t_i},\dots,e_{h_{\max}-h_i}^{t_i}\right\}\right), \tag{33}$$

where for any i, and $j\in[h-h_{t_i}]$, $\mathsf{v}^-\left(e_j^{t_i}\right)=s$ and $\mathsf{v}^+\left(e_j^{t_i}\right)=t_i$ (*e.g.*, G,G' are the graphs in cases (b),(c) in Figure 4, respectively). We then run the algorithm from [8] (as described in Section III) on G'.

Let

$$\underline{x} = x_{1,1},\dots,x_{1,h_{\max}},\dots\dots,x_{N,1},\dots,x_{N,h_{\max}} \tag{34}$$

be an $(N\cdot h_{\max})$-dimensional vector, consisting of N sub-vectors of h elements, s.t. each sub-vector describes a value of X. For some m which we will specify later, we generate $m\cdot h_{\max}$ vectors, each of dimension $N\cdot h_{\max}$,

$$\underline{a}_{1,1},\dots,\underline{a}_{1,h_{\max}},\dots\dots,\underline{a}_{m,1},\dots,\underline{a}_{m,h_{\max}}, \tag{35}$$

whose each symbol is distributed i.i.d. from[5] $\mathcal{U}_{\mathcal{F}}^{N\cdot h_{\max}}$. At time j, s transmits the $h_{\max}$ symbols

$$\begin{cases} x_{j,1},\dots,x_{j,h_{\max}}, & j\le N \\ \underline{a}_{j-N+1,1}\cdot\underline{x}^+,\dots,\underline{a}_{j-N+1,h_{\max}}\cdot\underline{x}^+, & N<j\le N+m \end{cases}.$$

Let t_i be a sink node, $G_{t_i}^f$ be the flow graph from s to t_i described in Section III, and

$$\left\{\underline{b}_1,\dots,\underline{b}_{h_i}\right\} = \left\{\mathbf{b}(e) \mid e\in G_{t_i}^f \setminus \left\{e_1^{t_i},\dots,e_{h_{\max}-h_i}^{t_i}\right\}\right\}. \tag{36}$$

Lemma 6: For $N<j\le N+m$, $j'=j-N+1$, and $k\in[h_i]$, the distribution of

$$[\underline{b}_{k+1}^+]\cdot[\underline{a}_{j,1}^+,\dots,\underline{a}_{j',h_{\max}}^+]^+ \tag{37}$$

given

$$[\underline{b}_1^+,\dots,\underline{b}_k^+]\cdot[\underline{a}_{j',1}^+,\dots,\underline{a}_{j',h_{\max}}^+]^+, \tag{38}$$

is $\mathcal{U}_{\mathcal{F}}^{N\cdot h_{\max}}$.

Proof: The proof is by the following series of inequalities:

$$\begin{aligned} N\cdot h_{\max}\cdot\log(q) &= H\left([\underline{a}_{j',1},\dots,\underline{a}_{j',h_{\max}}]\right) \\ &\overset{(a)}{=} H\left([\underline{b}_1^+,\dots,\underline{b}_{h_{\max}}^+]\cdot[\underline{a}_{j',1}^+,\dots,\underline{a}_{j',h_{\max}}^+]^+\right) \\ &\overset{(b)}{\le} H\left([\underline{b}_1^+,\dots,\underline{b}_k^+]\cdot[\underline{a}_{j',1}^+,\dots,\underline{a}_{j',h_{\max}}^+]^+\right) \\ &+H\left([\underline{b}_{k+1}^+]\cdot[\underline{a}_{j',1}^+,\dots,\underline{a}_{j',h_{\max}}^+]^+\right) \\ &+H\left([\underline{b}_{k+2}^+,\dots,\underline{b}_{h_{\max}}^+]\cdot[\underline{a}_{j',1}^+,\dots,\underline{a}_{j',h_{\max}}^+]^+\right) \\ &\le N\cdot h_{\max}\cdot\log(q). \end{aligned} \tag{39}$$

[5]For any m, we signify that the distribution of an m-dimension vector with elements from $\mathcal{F}$ is uniform over all q^m possibilities, by $\mathcal{U}_{\mathcal{F}}^m$.

In the above, (a) follows from the fact that the algorithm in [8] constructs the set $\left\{\mathbf{b}(e) \mid e \in G_{t_i}^{f}\right\}$, so that its rank is $h_{\max}$, and so there is a bijection

$$\begin{aligned}[\underline{a}_{j',1}^{+}, \ldots, \underline{a}_{j',h_{\max}}^{+}] \longleftrightarrow \\ [\underline{b}_1^{+}, \ldots, \underline{b}_{h_{\max}}^{+}] \cdot [\underline{a}_{j',1}^{+}, \ldots, \underline{a}_{j',h_{\max}}^{+}]^{+}.\end{aligned} \tag{40}$$

Inequality (b) follows by the independence bound on entropy [3]. ■

To prove Theorem 3, we use the above, the following theorem (whose proof is in the appendix), and the union bound.

Lemma 7: Let $N_i' = N\frac{h_i}{h_{\max}}$, $\widetilde{N}_i = N_i' - h_i + 1$, and A_i be the matrix

$$\left|\begin{array}{cccccc}
s_{1,1} & \cdots & s_{1,h_{\max}} & & & \\
\vdots & \vdots & \vdots & & & \\
s_{h_i,1} & \cdots & s_{h_i,h_{\max}} & & & \\
 & & \ddots & \ddots & & \\
 & & & s_{\widetilde{N}_i,N-h_{\max}+1} & \cdots & s_{\widetilde{N}_i,N} \\
 & & & \vdots & \vdots & \vdots \\
 & & & s_{N_i',N-h_{\max}+1} & \cdots & s_{N_i',N} \\
s_{N_i'+1,1} & \cdots & \cdots & \cdots & \cdots & s_{N_i'+1,N} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
s_{\alpha(N,c),1} & \cdots & \cdots & \cdots & \cdots & s_{\alpha(N,c),N}
\end{array}\right|,$$

where each empty entry represents 0, the entries in the first N_i' rows are set so that each batch of h_i rows are linearly independent, and the entries in the remaining rows are set randomly i.i.d. over $GF(q)$.

Then with probability at least $1 - \frac{1}{N^{c-1}}$, A_i contains a non-singular $N \times N$ sub-matrix.

V. Increasing the Coding Alphabet-Size by Using Node Memory

As shown in Sections II, III, and IV, the use of linear codes for multicast requires an underlying field with (non-tight) lower and upper bounds on its size. It is possible that the capacity of the links dictates a field $\mathcal{F}$ whose size is lower than the upper bound or even the lower bound.

Consider a network whose links transmit symbols over a field $\mathcal{F}'$, of size $q' = |\mathcal{F}'|$, which is large enough for linear multicast of $h \cdot \log(q')$ symbols per time unit. Ideally, a network over an isomorphic graph, whose links transmit symbols from $\mathcal{F}$, could transmit

$$\frac{h \cdot \log(q)}{\left\lceil \log_q(q') \right\rceil} \tag{41}$$

symbols per time unit. However, q might not be sufficiently large for linear multicast. We now show that the addition of node memory can suffice for the case of sequential multicast.

In this section, let

$$k = k(q, q') = \left\lceil \log_q(q') \right\rceil. \tag{42}$$

Assume that there is a multicast code (over $\mathcal{F}'$), s.t. at every k-th time unit, s transmits $H(x)$ information, and each link carries a symbol from $\mathcal{F}'$ (*i.e.*, the transmission rate is $\frac{N \cdot H(X)}{N \cdot k} = \frac{H(X)}{k}$).

It is clear that by accumulating every k consecutive symbols reaching a node by an edge, a code over $\mathcal{F}'$ can be simulated:

Theorem 4: By adding in each node an additional

$$k \cdot \left(\max_{v \in V} \left| \mathsf{E}^{-}(v) \dot{\bigcup} \mathsf{E}^{+}(v) \right| + 1 \right) \tag{43}$$

units of memory over $\mathcal{F}$, a good multicast code can be built, s.t. at every time unit each link carries a symbol from $\mathcal{F}$, and which can transmit $N\frac{H(X)}{k}$ information in $N + k \cdot n$ time units.

VI. Conclusions and Future Work

In this work we have shown lower and upper alphabet-size bounds for linear codes for multicast. As can be seen from Sections II and III, there is a gap between the lower and upper bound. We do not know whether any of these bounds is tight for general graphs, nor do we know any meaningful specification of graph topology which is relevant to the tightness of these bounds.

Another issue we are considering is that of the rate in cyclic networks with small alphabet-size. Previous work [1], [6], [8], [9] has dealt with cyclic graphs by means very similar to a *TTL* (time to live) packet field (see [7]), affecting the rate negligibly when the alphabet size is large. It is unclear whether the existence of cycles affects the rate negligibly as well, when the alphabet size is small.

VII. Acknowledgements

Thanks to Prof. S. Litsyn, of the department of EE - Systems in Tel-Aviv University, for his help throughout the work.

Appendix

In this section we prove Theorem 7 from Section IV.

Proof: Let $S_{N'_i+1}, \ldots, S_N$, be a partition of the rows of A_i's rectangular part, s.t.

$$|S_j| = \begin{cases} 1, & j = N'_i + 1, \ldots, N - c \cdot \log_q(N) + 1 \\ \frac{c \cdot \log_q(N)}{N-j+1}, & j = N - c \cdot \log_q(N) + 2, \ldots, N \end{cases} . \tag{44}$$

Let $s_1, \ldots, s_{N'_i}$ be the vectors in the first rows of the above matrix. We choose $N - N'_i$ vectors $s_j \in S_j$, so that s_j maximizes $Rank\left(\{s_1, \ldots, s_{j-1}, s_j\}\right)$. Let V_j be the event

$$\forall_{s \in S_j} Rank\left(\{s_1, \ldots, s_{j-1}, s\}\right) = Rank\left(\{s_1, \ldots, s_{j-1}\}\right). \tag{45}$$

It follows that

$$\mathbf{P}\left(\bigcup_{j=N'_i+1,\ldots,N} V_j\right) \le \sum_{j=N'_i+1}^{N} \mathbf{P}\left(V_j\right) \le \sum_{j=N'_i+1}^{N} \left(\frac{q^{j-1}}{q^N}\right)^{|S_j|} \le \sum_{j=N'_i+1}^{N} q^{-c \cdot \log_q(N)} \le \frac{1}{N^{c-1}}. \tag{46}$$

The total number of vectors generated is

$$\alpha(N,c) = N'_i + \sum_{j=N'_i+1,\ldots,N} N_j = N - c \cdot \log_q(N) + 1 + c \cdot \log_q(N) \sum_{k=1}^{c \cdot \log_q(N)-1} \frac{1}{k} \le N + c \cdot \log_q(N) \left(\ln\left(c \cdot \log_q(N)\right) + 1\right). \tag{47}$$

■

References

[1] R. Ahlswede, N. Cai, S.Y. R. Li, R. W. Yeung. "Network Information Flow", IEEE Trans. Inform. Theory, 46(4):1204-1216, 2000.
[2] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, The MIT Press and McGraw-Hill BookCompany 1989.
[3] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, New York: Wiley, 1991.
[4] M. Feder and N. Shulman, "The Static Broadcast Channel", proceedings of the International Symposium on Information Theory, Italy, June, 2000.
[5] F. J. MacWilliams and N. J. A. Sloane The Theory of Error-Correcting Codes North Holland Mathematical Library, 1977.
[6] R. Koetter and M. Medard, "An Algebraic Approach to Network Coding", Submitted to IEEE Trans. Networking.
[7] "RFC 791 Internet Protocol", DARPA Internet Program, Sept., 1981, http://www.ietf.org/rfc/rfc0791.txt
[8] P. Sanders, S. Egner, and L. Tolhuizen "Polynomial Time Algorithms for Network Information Flow"
[9] S.Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding", to appear in IEEE Trans. Inform. Theory.