

# Fundamentals of Network Coding: A Review

Srinath Puducheri

## I. INTRODUCTION

Network coding broadly refers to the processing of data at the intermediate nodes of a network in addition to routing data from source to destination. The theoretical framework for network coding was first provided by Ahlswede *et al.* in [1]. In this paper, the authors demonstrate the benefits of network coding over mere routing in improving the throughput of a single-source *multicast* transmission, i.e., when the same data at a source is to be transmitted to multiple destinations in the network. Specifically, a communication network is modeled as a directed graph with error-free links, each of finite transmission capacity; by means of a random coding argument, the authors show that it is possible for a source to achieve a multicast rate equal to the minimum of the *max-flows* to the individual destinations via network coding.

This work has been followed by several results on the explicit construction of network codes. In particular, Li *et al.* [2] show that *linear network codes* suffice to achieve the max-flow rate for the single-source multicast scenario. In [4], algorithms for the construction of such linear network codes are presented, which execute in time that is polynomial in the number of edges and vertices in the network.

An algebraic framework for network coding is developed by Koetter and Médard in [3] – they address the more general case of multiple sources communicating with multiple destinations; further, necessary and sufficient conditions for a given set of connections to be feasible under linear network coding are derived. In addition, [3] and [4] also address the problem of designing multicast network codes that are robust to link failures.

In this report, we shall focus primarily on the results for the single-source multicast scenario obtained in [1] and [2].

## II. NOTATION AND GRAPH-THEORETIC BACKGROUND

The networks we shall deal with consist of communicating nodes that are interconnected by non-interfering point-to-point links. The links are assumed to be ideal, i.e., error-free, and of finite information carrying capacity (bits per unit time). The network as a whole is represented by a *directed graph*  $G =$

$(V, E)$ , where  $V = \{1, 2, \dots, |V|\}$  is the set of vertices and  $E \subseteq V \times V$  is the set of edges<sup>1</sup> (ordered pairs of vertices). With edge  $(i, j)$  we associate an *edge capacity*  $R_{ij} \geq 0$  corresponding to its information carrying capacity.

A directed graph is said to be *cyclic* if it contains a directed cycle, i.e., there exists a sequence of edges  $(i_0, i_1), \dots, (i_m, i_0)$  for some  $m$ . Otherwise, the graph is *acyclic*.

A *cut* between a source node  $s$  and a sink node  $t$  is a set of vertices  $B$  such that  $s \in B$  and  $t \notin B$ . Define the set of edges *in* the cut to be  $E_B = \{(i, j) \in E : i \in B, j \notin B\}$ . Then the *value* of a cut is given by  $\sum_{(i,j) \in E_B} R_{ij}$ .

A *flow* from a source  $s$  to sink  $t$  ( $s, t \in V$ ) is an assignment of real numbers  $F_{ij}$  to edges  $(i, j)$  such that  $0 \leq F_{ij} \leq R_{ij}$ , and for all  $i \in V$  not equal to  $s$  or  $t$ , we have

$$\sum_{i':(i',i) \in E} F_{i'i} = \sum_{j:(i,j) \in E} F_{ij}, \quad (1)$$

i.e., the total flow into node  $i$  = total flow out of node  $i$ . The *value* of the flow is defined as the total flow *out of*  $s$  which is equal to the total flow *into*  $t$ . The *max-flow* from  $s$  to  $t$  is a flow whose value is greater than any other flow from  $s$  to  $t$ . We shall denote the value of the max-flow between  $s$  and  $t$  as  $\text{max-flow}(s \rightarrow t)$ .

The *Max-flow Min-cut theorem* [5] says that the value of the max-flow from  $s$  to  $t$  is given by the minimum value of a cut between  $s$  and  $t$ , i.e., if  $B$  is a cut between  $s$  and  $t$  and  $E_B$  is the set of edges in the cut, then:

$$\text{max-flow}(s \rightarrow t) = \min_B \sum_{(i,j) \in E_B} R_{ij} \quad (2)$$

### III. THE NETWORK MULTICAST PROBLEM

Consider a network given by the directed graph  $G = (V, E)$ . Let  $\mathbf{R} = [R_{ij}, (i, j) \in E]$  denote the vector of edge capacities associated with this graph, i.e., the average transmission rate over edge  $(i, j)$  cannot exceed  $R_{ij}$  bits per unit time. Let  $s$  be the source node and  $T = \{t_i : 1 \leq i \leq L\}$  be a set of  $L$  sink (destination) nodes. The node  $s$  is associated with an information source  $X$  that produces  $h$  bits per unit time. The *multicast problem* is to design a coding strategy using which these bits can be communicated at the same rate  $h$  to the set of sinks  $T$ .

Given a vector  $\mathbf{R}$  of edge capacities, we are interested in the maximum value of  $h$  that can be supported, i.e., for which the multicast problem has a solution. Equivalently, we can attempt to characterize the set  $R$

<sup>1</sup>Later we shall make use of a slightly different and more general definition of  $E$  which allows for multiple parallel edges corresponding to the same ordered pair of vertices.

of all  $\mathbf{R}$  that support a given source information rate  $h$ . Every such rate-tuple  $\mathbf{R}$  is said to be *admissible*. In order to obtain bounds on the admissible rate-tuples  $\mathbf{R}$ , it is necessary to obtain a complete and general characterization of any coding scheme that may be applied to an arbitrary network to solve the corresponding multicast problem. This is an extremely hard task. Instead the authors in [1] propose a fairly general subclass of network coding schemes called  $\alpha$ -codes.

**Definition 1** ( $\alpha$ -code, [1]). An  $(n, (\eta_{ij}, (i, j) \in E), h)$   $\alpha$ -code on a graph  $G$  consists of a sequence of  $K$  transmissions that take place in chronological order. The message  $x$  to be communicated to the sinks  $T$  by the source node  $s$  is chosen from a set  $\Omega$  with a uniform distribution, where

$$\Omega = \{1, \dots, \lceil 2^{nh} \rceil\}. \quad (3)$$

In the  $k$ th transmission, node  $u(k)$  transmits a message  $m_k$  to node  $v(k)$ . The message  $m_k$  is chosen from a set  $A_k = \{1, \dots, |A_k|\}$  according to an encoding function  $f_k$ , which takes into account messages already received by node  $u(k)$  from previous transmissions. If  $u(k) = s$ , then

$$f_k : \Omega \rightarrow A_k. \quad (4)$$

If  $u(k) \neq s$ , then

$$f_k : \prod_{k' \in Q_k} A_{k'} \rightarrow A_k \quad (5)$$

where  $\prod$  denotes the Cartesian product of sets, and  $Q_k$  is the set of all prior transmissions to node  $u(k)$ , i.e.,

$$Q_k = \{1 \leq k' < k : v(k') = u(k)\}. \quad (6)$$

At the end of  $K$  transmissions, each sink  $t_l$  ( $1 \leq l \leq L$ ) applies a decoding function  $g_l$  to the messages it has received, as follows:

$$g_l : \prod_{k' \in W_l} A_{k'} \rightarrow \Omega, \quad 1 \leq l \leq L \quad (7)$$

where  $W_l$  is the set of all transmissions to node  $t_l$ , i.e.,

$$W_l = \{1 \leq k \leq K : v(k) = t_l\} \quad (8)$$

such that the application of  $g_l$  results in the original message  $x$  for any  $x \in \Omega$  chosen at the source.

The total number of bits transmitted over edge  $(i, j)$  is given by  $\log_2 \eta_{ij}$ , i.e.,

$$\eta_{ij} = \prod_{k \in T_{ij}} |A_k| \quad (9)$$

where

$$T_{ij} = \{1 \leq k \leq K : (u(k), v(k)) = (i, j)\} \quad (10)$$

**Definition 2** ( $\alpha$ -admissibility, [1]). A tuple  $(\mathbf{R}, h, G)$  is  $\alpha$ -admissible if for any  $\epsilon > 0$  there exists, for sufficiently large  $n$ , an  $(n, (\eta_{ij}, (i, j) \in E), h - \epsilon)$   $\alpha$ -code on  $G$  such that

$$\frac{1}{n} \cdot \log_2 \eta_{ij} \leq R_{ij} + \epsilon \quad (11)$$

for all  $(i, j) \in E$ .

Thus,  $\alpha$ -admissibility implies the existence of a solution to the multicast problem. In [1], the authors characterize the region of all rate-tuples for which a particular multicast problem can be solved using  $\alpha$ -codes. Their main result is that a tuple  $(\mathbf{R}, h, G)$  is  $\alpha$ -admissible if and only if the max-flows from  $s$  to  $t_l, 1 \leq l \leq L$ , resulting due to  $\mathbf{R}$ , are all at least  $h$  in value. This can be formally stated as follows.

**Theorem 1** ([1]). If

$$\mathcal{R}_{h,G} = \{\mathbf{R} : (\mathbf{R}, h, G) \text{ is } \alpha\text{-admissible}\} \quad (12)$$

and

$$\mathcal{R}_{h,G}^* = \{\mathbf{R} : \text{max-flow}(s \rightarrow t_l) \geq h, \quad 1 \leq l \leq L\} \quad (13)$$

then

$$\mathcal{R}_{h,G} = \mathcal{R}_{h,G}^* \quad (14)$$

A. *The Converse:*  $\mathcal{R}_{h,G} \subseteq \mathcal{R}_{h,G}^*$

This part of the argument relies primarily on the max-flow min-cut theorem and standard information theoretic inequalities.

Let  $X$  denote the message generated at node  $s$ . Thus, given an  $(n, (\eta_{ij}, (i, j) \in E), h)$   $\alpha$ -code, we have

$$h - \epsilon \leq \frac{1}{n} \cdot H(X). \quad (15)$$

For any sink  $t_l$ , consider a cut  $B \subset V$  with  $s \in B$  and  $t_l \notin B$ . Define

$$E_B = \{(i, j) \in E : i \in B \text{ and } j \notin B\}. \quad (16)$$

Then, using information theoretic arguments, it can be shown that the total number of bits carried on the edges in  $E_B$  must exceed the source entropy, if sink  $t_l$  is to be able to decode the message correctly.

Thus,

$$H(X) \leq \sum_{(i,j) \in E_B} \log_2 \eta_{ij} \quad (17)$$

However, by assumption

$$\frac{1}{n} \cdot \log_2 \eta_{ij} \leq R_{ij} + \epsilon \quad (18)$$

Putting all of this together, we obtain

$$h - \epsilon \leq \sum_{(i,j) \in E_B} R_{ij} + |E|\epsilon \quad (19)$$

Minimizing over all  $B$  and letting  $\epsilon \rightarrow 0$ , we have

$$h \leq \min_B \sum_{(i,j) \in E_B} R_{ij} \quad (20)$$

$$= \text{max-flow}(s \rightarrow t_l) \quad (21)$$

by the max-flow min-cut theorem. Thus, we obtain the desired result.

### B. Admissibility: $\mathcal{R}_{h,G}^* \subseteq \mathcal{R}_{h,G}$

In [1], the authors prove this part in two steps – first it is shown to be true for acyclic graphs, and then the proof is extended to cyclic graphs.

The proof for acyclic graphs relies on the fact that in acyclic graphs, the vertices can be numbered such that any edge originates from a lower-numbered vertex and terminates in a higher-numbered vertex. This presents a natural ordering of transmissions for an  $\alpha$ -code. Specifically, the transmissions can be ordered such that encoding function  $f_{ij}$  is applied before  $f_{i'j'}$  if  $i < i'$ , and  $f_{ij}$  is applied before  $f_{i'j'}$  if  $j < j'$ . This simplifies the analysis significantly, and it is shown by means of a random coding argument that such an  $\alpha$ -code that is admissible can be constructed if the max-flows from  $s$  to each sink exceed  $h$ . The details of the proof are omitted here and can be found in [1].

For cyclic graphs, there is no natural ordering of edges which allows encoding functions to be applied sequentially. Instead, we can map a cyclic graph  $G$  to an acyclic graph  $G^*$  by “expanding” it out as follows: the graph  $G^*$  consists of  $\Lambda$  layers of nodes, each layer containing a replica of the nodes in  $G$ ; however, edges are allowed only from a layer ‘ $i$ ’ to the next successive layer ‘ $i + 1$ ’ (no edges within a layer) and only connections corresponding to edges in  $G$  are valid. The edge capacities in  $G^*$  are also as dictated by those in  $G$ . The exact details can be found in [1]. As an example, if the graph  $G$  corresponds to the state transition diagram of a finite state machine (say a convolutional encoder), then  $G^*$  is similar to the trellis diagram that represents the time-evolution of possible state sequences.

It can be easily seen that the graph  $G^*$  is acyclic. It is shown in [1] that the set of multicast requirements and max-flow conditions on  $G$  translate to a corresponding set of multicast requirements and max-flow conditions on  $G^*$ , and also that  $\alpha$ -admissibility for  $G^*$  under these conditions implies  $\alpha$ -admissibility for  $G$ . But since  $G^*$  is acyclic, it is  $\alpha$ -admissible under the translated conditions (according to the first part of the proof). Consequently, the desired result is proved for cyclic graphs as well.

#### IV. LINEAR NETWORK CODING

It has been shown in [2] that the “minimum max-flow” rate for multicast (discussed in the last section) can be achieved using *linear* network codes, which significantly simplify encoding and decoding operations at the nodes. By “linear” we mean that the encoding and the decoding consist of taking linear combinations of received symbols over some finite field. This will be elaborated upon in the following.

We shall assume for the sake of simplicity that the edge capacities  $R_{ij}$  are non-negative integers. Consequently, in our graph  $G$ , we can view an edge  $(i, j)$  as being made up of  $R_{ij}$  parallel edges each of unit capacity. We denote this new transformed graph as  $G' = (V, E')$ . Also, we use the notation  $(i, j)_k$  to denote the  $k$ th edge from node  $i$  to node  $j$  in  $G'$  ( $1 \leq k \leq R_{ij}$ ). It is easily seen that the max-flow between any two nodes is the same in  $G$  and  $G'$ . Another assumption we make is that the graphs  $G$  and hence  $G'$  are acyclic, although this is not assumed for most of the development in [2]. This is not a problem as any cyclic graph can be translated to an acyclic graph (along the lines described in Section III-B), and the code constructed for the acyclic graph can then be translated back into a code for the cyclic graph.

A linear network code is first constructed for the graph  $G'$  with the same multicast requirements, and this code can easily be translated back into a linear code for the graph  $G$ . The main features of this code are:

- 1) The symbols transmitted over the network belong to a (sufficiently large) finite field  $\mathcal{F}$ . By assumption, over each edge we can transmit one such symbol per unit time.
- 2) Let  $\omega$  be a  $d$ -dimensional vector space over the base field  $\mathcal{F}$  (the choice of  $d$  will be given shortly).
- 3) At the source, the information to be transmitted is mapped to an *information vector*  $\mathbf{u} \in \omega$ . Associated with each edge  $e$  in the graph is a *code vector*  $\mathbf{v}_e \in \omega$  such that the symbol transmitted over  $e$  is  $\mathbf{u}^T \mathbf{v}_e$  (i.e., the inner product).

The value of  $d$  is chosen as

$$d = \max_{v \in V, v \neq s} \text{max-flow}(s \rightarrow v) \quad (22)$$

The desired properties of the network code are formally stated in the following definitions.

**Definition 3** (Linear-code multicast, [2]). *A linear-code multicast (LCM)  $\mathcal{L}$  on the network  $G' = (V, E')$  is an assignment of a vector space  $\mathcal{L}(v)$  to every  $v \in V$  and a vector  $\mathcal{L}((i, j)_k)$  to every edge  $(i, j)_k \in E'$  such that*

- 1)  $\mathcal{L}(s) = \omega$ ;
- 2)  $\mathcal{L}((i, j)_k) \in \mathcal{L}(i)$ ;
- 3) for any collection  $X$  of non-source nodes,

$$\langle \{ \mathcal{L}(v) : v \in X \} \rangle = \langle \{ \mathcal{L}((i, j)_k) : i \notin X, j \in X \} \rangle \quad (23)$$

(‘ $\langle \cdot \rangle$ ’ denotes linear span).

The definition of an LCM  $\mathcal{L}$  implies the following:

- 1) The vector space  $\mathcal{L}(v)$  associated with a vertex  $v$  is spanned by the vectors associated with its incoming edges.
- 2) The vector assigned to an outgoing edge from  $v$  must be a linear combination of the vectors assigned to the incoming edges of  $v$ .

Thus, an LCM  $\mathcal{L}$  can be viewed as a network code with coding vectors  $\mathcal{L}((i, j)_k)$  associated with each edge  $(i, j)_k$ , such that the outgoing symbols from any node are derived from appropriate linear combinations of the (received) incoming symbols.

Clearly then, if an LCM  $\mathcal{L}$  is to be used as a network code, the amount of information that can be collected by a node  $v$  from the symbols it receives depends on the dimensionality of the vector space  $\mathcal{L}(v)$  associated with it. Specifically, assume that the source transmits  $d$  symbols of information, i.e., it utilizes *all* of the available dimensions of  $\omega$  in constructing the information vector; then *only* those nodes  $v$  in the network for which  $\dim(\mathcal{L}(v)) = d$  can recover all the information. Such nodes may or may not exist, depending on the LCM chosen.

Thus, we need to impose suitable constraints on an LCM  $\mathcal{L}$  which ensure that  $\dim(\mathcal{L}(v))$  is the maximum possible for each  $v \in V$ .

**Definition 4** (Generic LCM, [2]). *An LCM  $\mathcal{L}$  is said to be generic if for any collection of  $m(\leq d)$  edges  $(i_1, j_1)_{k_1}, \dots, (i_m, j_m)_{k_m}$ , we have the following equivalence condition:*

$\mathcal{L}(i_1, j_1)_{k_1}, \dots, \mathcal{L}(i_m, j_m)_{k_m}$  are linearly independent  $\iff \mathcal{L}(i_p) \not\subseteq \langle \{ \mathcal{L}(i_q, j_q)_{k_q} : q \neq p \} \rangle$  for  $1 \leq p \leq m$ .

**Theorem 2** ([2]). *Given any LCM  $\mathcal{L}$ , for every vertex  $v \neq s$ ,*

$$\dim(\mathcal{L}(v)) \leq \text{max-flow}(s \rightarrow v). \quad (24)$$

*If  $\mathcal{L}$  is generic, then for all  $v \neq s$ ,*

$$\dim(\mathcal{L}(v)) = \text{max-flow}(s \rightarrow v). \quad (25)$$

Thus, a generic LCM, if it exists, is the “best” possible LCM. In particular, given a generic LCM  $\mathcal{L}$ , for any node  $v$  such that  $\text{max-flow}(s \rightarrow v) = d$ , we have  $\dim(\mathcal{L}(v)) = d$ . Hence, it is possible for the source to *multicast*  $d$  symbols of information to *all* nodes with a max-flow equal to  $d$ . Note that  $d$  is the maximum possible max-flow (from  $s$ ) in the network. Thus in short, using a generic LCM, it is possible to meet the multicast demand for the special case when the source information rate  $h$  and the max-flows to the intended sinks *are all equal* to the maximum possible max-flow in the network.

However, as pointed out in [2], using a generic LCM it is also possible to tackle the following general problem – the source  $s$  transmits  $d' (\leq d)$  symbols of information and it is required that *any* node  $v$  with  $\text{max-flow}(s \rightarrow v) \geq d'$  be able to recover all the information. This relies on the fact that, in  $G'$ , removing an incoming edge to any node with a higher max-flow cannot reduce lower max-flows to other nodes. Thus, by removing the incoming edge(s) to the node with the maximum max-flow at any stage and proceeding in a recursive manner, the maximum max-flow of the graph can be brought down to any desired value  $d'$ . If a generic LCM then exists for this new graph, then the multicast requirements on the old graph can be met. The reader is referred to [2] for a detailed description of this procedure.

It remains to be seen if a generic LCM can be constructed for any  $G'$ . This is answered by the following theorem.

**Theorem 3** ([2]). *A generic LCM exists on every acyclic  $G'$ , provided that the base field  $\mathcal{F}$  of  $\omega$  is an infinite field or a large enough finite field.*

The proof of the above is constructive and exploits the fact that in an acyclic directed graph, there exists an ordering of vertices such that any edge always originates in a lower indexed vertex and terminates in a higher indexed vertex. The process of choosing a coding vector for each edge can then be done by proceeding from the lowest indexed vertex to the highest. Consequently, when edge  $(i, j)_k$  is being processed, all edges terminating in  $i$  have already been assigned coding vectors. Let the span of the incoming vectors to  $i$  be  $\mathcal{L}(i)$ . Then, the vector for edge  $(i, j)_k$  is chosen such that it lies in  $\mathcal{L}(i)$ , but *does not lie* in the span of *any* collection of at most  $d - 1$  assigned coding vectors that do not span

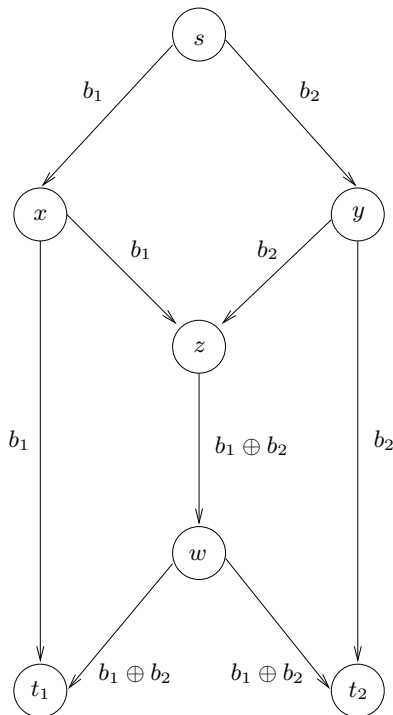


Fig. 1. A single-source two-sink network

$\mathcal{L}(i)$ . Such a vector is guaranteed to exist if the base field  $\mathcal{F}$  is large enough. The resulting allocation of vectors by this algorithm results in a generic LCM [2].

(Note: Among all the results presented in this section, this is the only theorem that actually requires  $G'$  to be acyclic. In other words, all the previous results can be proved to be true regardless of the acyclic assumption [2].)

#### A. An example

Consider the network represented by the graph in Fig. 1. The source node is given by  $s$ , and there are two sinks  $t_1$  and  $t_2$ . All edges in this graph are of *unit* capacity. Information is to be multicast from  $s$  to  $t_1$  and  $t_2$ . We shall construct a network code for this example.

At the outset, note that  $\max\text{-flow}(s \rightarrow t_1) = \max\text{-flow}(s \rightarrow t_2) = 2$ . Consequently, according to our discussion so far, *two bits* of information can be transmitted from  $s$  simultaneously to  $t_1$  and  $t_2$ . Also, the maximum max-flow  $d = 2$  in this network.

The finite field we consider is  $\mathcal{F}_2$ , i.e., the binary field  $\{0, 1\}$  under addition and multiplication modulo 2 (i.e., bit-wise addition and multiplication). It turns out that a field size of two is sufficient for this network.

With  $d = 2$ , the vector space  $\omega$  for this example is  $\mathcal{F}_2^2$  – the set of 2-tuples of elements in  $\mathcal{F}_2$ , which can be enumerated as  $[0\ 0]$ ,  $[0\ 1]$ ,  $[1\ 0]$  and  $[1\ 1]$ .

A generic LCM  $\mathcal{L}$  for this network can be constructed by systematically allocating vectors to edges in the following manner:

- 1) Since the graph is acyclic, we order the vertices as  $s, x, y, z, w$  for encoding. Note that the sinks  $t_1$  and  $t_2$  do not need to encode in this particular example as there are no outgoing edges from them.
- 2) **Node  $s$ :**  $\mathcal{L}(s) = \omega = \mathcal{F}_2^2$ .
  - a) Let  $\mathcal{L}((s, x)) = [1\ 0]$ .
  - b) Since  $\mathcal{L}((s, y))$  should not lie in the linear span of  $\mathcal{L}((s, x))$ , it can be either  $[0\ 1]$  or  $[1\ 1]$ . We choose  $\mathcal{L}((s, y)) = [0\ 1]$ .
- 3) **Node  $x$ :**  $\mathcal{L}(x) = \{[0\ 0], [1\ 0]\}$ . Note that the only possible allocations are  $\mathcal{L}((x, t_1)) = [1\ 0]$  and  $\mathcal{L}((x, z)) = [1\ 0]$ . These allocations still satisfy the desired conditions, i.e., they do not lie in the span of  $\{[0\ 1]\}$ .
- 4) **Node  $y$ :** Similarly,  $\mathcal{L}(y) = \{[0\ 0], [0\ 1]\}$  and  $\mathcal{L}((y, t_2)) = \mathcal{L}((y, z)) = [0\ 1]$ .
- 5) **Node  $z$ :** Note that  $\mathcal{L}(z) = \omega$ , the entire vector space. Thus, there is seemingly more freedom in picking  $\mathcal{L}((z, w))$ . However, recall that  $\mathcal{L}((z, w))$  should be linearly independent (l.i.) with every assigned vector that does not span  $\mathcal{L}(z)$  (note that  $d - 1 = 1$  for this case). Hence  $\mathcal{L}((z, w))$  should be l.i. with  $\{[0\ 1]\}$  and with  $\{[1\ 0]\}$ . Thus, the only possibility is  $\mathcal{L}((z, w)) = [1\ 1]$ .
- 6) **Node  $w$ :**  $\mathcal{L}(w) = \{[0\ 0], [1\ 1]\}$ . Thus, again the only possibility is  $\mathcal{L}((w, t_1)) = \mathcal{L}((w, t_2)) = [1\ 1]$ , and this does not violate the conditions for the construction of a generic LCM.

Consequently, if the information vector at the source  $[b_1\ b_2]$  consists of bits  $b_1$  and  $b_2$ , then the bits transmitted along the edges of the graph are as indicated in Fig. 1. It is easily seen that both sinks can decode  $b_1$  and  $b_2$  from the bits they receive.

## V. OTHER RELATED WORK

We have reviewed the results of [1] and [2] regarding the single-source multicast problem in communication networks. These papers deal with ideal links that never fail. An interesting problem is – what happens if the links in a network are error-free but can fail (i.e., can be permanently removed)? It has been shown by Koetter and Médard [3] that there in fact exists a *static* solution to the network coding problem that is robust to the occurrence of any failure pattern which does not render the multicast problem infeasible. The code is static in the sense that it is designed prior to the failure of any edges and the coding vectors at any edge need not be changed as a result of the failures (i.e., the network code

need not be redesigned). The assumptions are that the failed edges transmit the zero symbol, and that the sinks are informed of the failure pattern. The same problem is also dealt with by Jaggi *et al.* in [4], and they improve on the field size requirement given by Koetter and Médard [3].

Interestingly, it has been shown by Dougherty *et al.* [6] that linear network coding techniques are, in general, *insufficient* for general network information flow problems involving multiple sources. In fact, the class of linear codes that they consider is larger than what is discussed here.

#### REFERENCES

- [1] R. Ahlswede, N. Cai, S.-Y. R. Li, R. W. Yeung, "Network Information Flow," *IEEE Trans. on Information Theory*, vol. 46, no. 4, pp. 1204-1216, July 2000
- [2] S.-Y. R. Li, R. W. Yeung, N. Cai, "Linear Network Coding," *IEEE Trans. on Information Theory*, vol. 49, no. 2, pp. 371-381, February 2003
- [3] R. Koetter, M. Médard, "An Algebraic Approach to Network Coding," *IEEE/ACM Trans. on Networking*, vol. 11, no. 5, pp. 782-795, October 2003
- [4] S. Jaggi, P. Sanders, P.A. Chou, M. Effros, S. Egner, K. Jain, L. Tolhuizen, "Polynomial Time Algorithms for Multicast Network Code Construction," *IEEE Trans. on Information Theory*, vol. 51, no. 6, pp. 1973-1982, June 2005
- [5] B. Bollobas, *Graph Theory, An Introductory Course*. New York: Springer-Verlag, 1979
- [6] R. Dougherty, C. Freiling, and K. Zeger, "Insufficiency of linear coding in network information flow," *IEEE Trans. on Information Theory*, vol. 51, no. 8, pp. 2745-2759, August 2005.