Network codes as codes on graphs

Ralf Koetter Michelle Effros Tracey Ho Muriel Médard

***Abstract*—This paper points out connections between linear network coding and linear system theory. In particular, a network code is interpreted as a state space realization of a network behavior that implements a desired set of network connections. A reversibility theorem is derived for network coding that is a direct consequence of a fundamental duality theorem derived by Forney in the context of state-space realizations.**

I. Introduction

Starting from the seminal work of Ahlswede et al. [1] it is by now well understood that network coding is an essential ingredient in achieving network capacity and a significant research activity is developing around this topic. It turns out that the problem of network coding for the multicast is especially attractive and Li et al. could show that *linear* network coding is in fact sufficient to achieve capacity in this setup. Consequently, significant progress has been made for the question of multicast network coding [2], [9], [10], [6], [11]. On the other hand, the general network coding problem has remained elusive. While the multicast scenario can capitalize on coding theoretic tools — in particular MDS codes — the general problem is far more intricate. The goal of this paper is to contribute to the understanding of the foundations of linear network coding. In particular, the goal is to exhibit a close connection of network coding with the representation of linear behaviors in given graphs. This branch of linear system theory is essentially an outgrowth of both the general theory of graphical models and the theory of codes on graphs. While the parallels between these areas seem puzzling at first, it turns out that there is a surprisingly accurate way to translate insights on concepts between the different areas and we can utilize understanding and tools across the area boundaries in a natural way. Here we focus on state space realization as a prototype of a graphical model in linear systems and show connections and parallels to network coding.

II. Basics

In this section we set up the basic notations that are applicable to networks and state space realization. We begin by defining the notations for state space realizations. For a thorough treatment of state space realizations in arbitrary graph we refer to the seminal paper by G.D. Forney [20]. See also [15], [16], [17].

R. Koetter (koetter@uiuc.edu) is with the Coordinated Science Laboratory, University of Illinois at Urbana-Champain, Urbana, IL 61801. M. Effros (effros@caltech.edu) is with the Department of Electrical Engineering, California Insitute of Technology, Pasadena, CA 91125. T. Ho (trace@mit.edu) and M. Médard (medard@mit.edu) are with the Laboratory for Information and Decision Systems (LIDS), Massachusetts Institute of Technology, Cambridge, MA 02139. This research is supported in part by NSF Grants CCR-0325324 and CCR-0220039, University of Illinois subaward #02-194, Hewlett-Packard 008542-008, and Caltech's Lee Center for Advanced Networking.

II-A. State-Space Realizations

A *factor graph* $\mathcal{G} = (V, E)$ with vertex set V and edge set E is a bipartite graph consisting of a set of state/symbol nodes V_S, a set of function nodes V_f , (i.e., $V = V_S \cup V_f$,) and a set of edges $E \subseteq V_S \times V_f$. For factor graphs, we consider edges as sets of pairs of vertices and hence $\{v, u\} \in E$ implies $\{u, v\} \in E$. We will use the terms "node" and "vertex" interchangeably.

To each vertex $u \in V_S$ we associate a set of symbols $\mathscr{A}_u$ called an alphabet. In the context of this paper we will assume that all alphabets are finite.

The *configuration space* for the factor graph $\mathcal{G}$ is defined as the Cartesian product $\mathscr{A} = \bigotimes_{u \in V_S} \mathscr{A}_u$. The restriction or projection of the configuration space to a subset $V' \subset V_S$ of vertices is denoted by $\mathscr{A}_{|V'}$. Similarly, the restriction of any element $\underline{a} \in \mathscr{A}$ to V' is denoted $\underline{a}_{|V'}$.

For any $v \in V$ let $\Gamma(v)$ denote the neighborhood of v, i.e.,

$$\Gamma(v) = \Big\{ v' : \{v, v'\} \in E \Big\}.$$

We assume a fixed ordering of the elements of $\Gamma(v)$ for all $v \in V_f$, and associate to each $v \in V_f$ a *local behavior* $\mathscr{C}_v \subseteq \bigotimes_{u \in \Gamma(v)} \mathscr{A}_u$.

For a given set of local behaviors a *global behavior* $\mathscr{C}$ is obtained as a subset of $\mathscr{A}$ by requiring that the restriction of any element of $\mathscr{C}$ to $\Gamma(v)$ is an element in the local behavior $\mathscr{C}_v$ for all $v \in V_f$, i.e.,

$$\mathscr{C} = \{\underline{a} \in \mathscr{A} : \underline{a}_{|\Gamma(v)} \in \mathscr{C}_v, \forall v \in V_f\}.$$

Let $I_{\mathscr{C}}$ and $I_{\mathscr{C}_v}$ be set indicator functions for sets $\mathscr{C}$ and $\mathscr{C}_v$, respectively, that is $I_{\mathscr{C}}(\underline{a})$ evaluates to one if $\underline{a} \in \mathscr{C}$ and evaluates to zero otherwise. The notion of a factor graph reflects the factorization of the function $I_{\mathscr{C}}$ as $I_{\mathscr{C}}(\underline{a}) = \prod_{v \in V_f} I_{\mathscr{C}_v}(\underline{a}_{|\Gamma(v)})$.

Forney [20] introduced a specialization of factor graphs which will lead to a notion of a *state-space realization* that is suitable for this paper. In particular, Forney distinguishes between nodes in V_S as symbol nodes V_T and state nodes V_L, i.e., $V_S = V_L \cup V_T$. In the context of this paper, it will turn out that state nodes reflect the capacity of links in the network (hence the index L), while symbol nodes correspond to transmitted data (hence the index T).

Definition 1 *A normal graph is a factor graph with vertex set $V = V_S \cup V_f$, $V_S = V_L \cup V_T$ such that all vertices in V_L have degree exactly two and all vertices in V_T have degree exactly one.* ■

The underlying understanding is that symbol nodes V_T are *observable* while state nodes V_L are *hidden* and correspond to

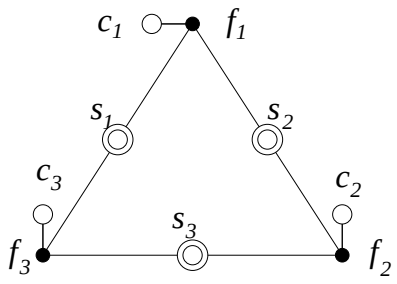

Fig. 1. A normal graph representing a linear behavior. Symbol nodes are represented by small unfilled circles, local behaviors are represented by small unfilled circles, and states are represented by double circles.

a state in the network. We are particularly interested in the restriction of $\mathscr{C}$ to the vertices V_T.

Definition 2 *Let $\mathcal{G} = (V_f \cup \{V_L \cup V_T\}, E)$ be a normal graph with global behavior $\mathscr{C}$. We say that the normal graph $\mathcal{G}$ represents a code $\mathcal{C}$ if $\mathcal{C} = \mathscr{C}_{|V_T}$.* ■

Example 1 Before we continue we give an example for a normal graph, depicted in Figure 1, that represents a simple linear behavior $\mathcal{C}$ with generator matrix

$$G = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

.

The bits in words in $\mathscr{C}$ are grouped in groups of three so that the symbol nodes c_1, c_2, c_3 in Figure 1 can assume values over an alphabet $\mathbb{F}_2^3$. The state nodes s_1, s_2, s_3 are assumed to take on values in $\mathbb{F}_2^2$. The local behaviors f_i are thus subspaces of $\mathbb{F}_2^3 \times \mathbb{F}_2^2 \times \mathbb{F}_2^2 \equiv F_2^7$ given by the three generator matrices

$$G_1 = \left(\begin{array}{ccc|cc|cc} 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{array}\right)$$

$$G_2 = \left(\begin{array}{ccc|cc|cc} 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{array}\right)$$

$$G_3 = \left(\begin{array}{ccc|cc|cc} 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{array}\right)$$

■

We will slightly extend the notion of normal graph and state-space realization by allowing arrows in such graphical models. Whenever a link direction is indicated by an arrow we implicitly assume that information flows in the direction of the arrow.

Before we continue with a description of basic notations for communication networks we give the following lemma which allows the construction of complicated state space realization out of considerably simpler ones. For simplicity, we restrict the lemma to the binary case but it generalizes to arbitrary group alphabets with little effort.

The reader familiar with the subject will realize that the lemma may be interpreted as a reformulation of the product construction of trellises of Kschischang and Sorokine [18].

Let a normal graph $\mathcal{G} = (V, E)$ be given. Assume two different linear state space realizations indicated by superscripts A and B are described in the normal graph with symbol alphabets $\mathbb{F}_2^{s(u)_A}$, $\mathbb{F}_2^{s(u)_B}$ for $u \in V_S$ and suitably chosen integers $s(u)_A$ and $s(u)_B$. Let a new state space realization, indicated by a superscript C, be given with symbol alphabets $\mathbb{F}_2^{s(u)_A + s(u)_B} = \mathbb{F}_2^{s(u)_C}$ for $u \in V_S$. The new local behaviors $\mathscr{C}^C$ are given by simply concatenating the original local behaviors $\mathscr{C}^A$ and $\mathscr{C}^C$. We say that the state space realization C is obtained as the product of A and B.

Lemma 1 *Let two state space realizations be defined on the same normal graph each representing a behavior $\mathscr{C}_A$ and $\mathscr{C}_B$. The product of the two state space realizations represents a behavior $\mathscr{C}_C = \{(\underline{c}^A, \underline{c}^B) : \underline{c}^A \in \mathscr{C}_A, \underline{c}^B \in \mathscr{C}_B\}$*

Example 2[Example 1 continued] It is straightforward to verify that the state space realization in Example 1 may be thought of as obtained as the product of three state space realizations each representing a linear behavior corresponding to a binary repetition code of length three.

II-B. Communication Networks

For the purpose of this paper a communication network is a collection of directed links connecting transmitters, switches, and receivers. Hence, a network may be represented by a directed graph $\mathcal{G} = (V, E)$ with a vertex set V and an edge set E. In order to model links of different capacity in a uniform fashion, we will allow multiple edges between two vertices and, hence, E is a subset of $E \subseteq V \times V \times \mathbb{Z}_+$, where the last integer enumerates edges between two vertices. Thus, edges are denoted by round brackets $(v_1, v_2, i) \in E$ and assumed to be directed. The *head* and *tail* of an edge $e = (v', v, i)$ is denoted by $v = \text{head}(e)$ and $v' = \text{tail}(e)$.

We define $\Gamma_I(v)$ as the set of edges that end at a vertex $v \in V$ and $\Gamma_O(v)$ as the set of edges originating at v. Formally, we have

$$\Gamma_I(v) = \{e \in E : \text{head}(e) = v\}$$
$$\Gamma_O(v) = \{e \in E : \text{tail}(e) = v\}.$$

The *in-degree* $\delta_I(v)$ of v is defined as $\delta_I(v) = |\Gamma_I(v)|$ while the *out-degree* $\delta_O(v)$ is defined as $\delta_O(v) = |\Gamma_O(v)|$.

A network is called *cyclic* if it contains directed cycles, i.e. if there exists a sequence of edges $(v_0, v_1), (v_1, v_2), \ldots, (v_n, v_0)$ in $\mathcal{G}$. A network is called *acyclic* if it does not contain directed cycles. To each link $e \in E$ we associate a non-negative number $C(e)$, called the capacity of e.

Let $\mathscr{X}(v) = \{X(v,1), X(v,2), \ldots, X(v,\mu(v))\}$ be a collection of $\mu(v)$ discrete random processes that are observable at node v. We want to allow communication between selected nodes in the network, i.e. we want to replicate, by means of the network, a subset of the random processes in $\mathscr{X}(v)$ at some different node v'. We define a *connection* c as a triple $(v, v', \mathscr{X}(v, v')) \in V \times V \times \mathscr{P}_{\mathscr{X}(v)}$, where $\mathscr{X}(v, v')$ denotes a subset of $\mathscr{X}(v)$ The rate $R(c)$ of a connection $c = (v, v', \mathscr{X}(v, v'))$ is defined as $R(c) =$

$\sum_{i:X(v,i)\in\mathscr{X}(v,v')} H(X(v,i))$, where $H(X)$ is the entropy rate of a random process X.

Given a connection $c = (v, v', \mathscr{X}(v, v'))$, we call v a *source* and v' a *sink* of c and write $v = \text{source}(c)$ and $v' = \text{sink}(c)$. For notational convenience we will always assume that $\text{source}(c) \neq \text{sink}(c)$.

A node v can send information through a link $e = (v, u)$ originating at v at a rate of at most $C(e)$ bits per time unit. The random process transmitted through link e is denoted by $Y(e)$. In addition to the random processes in $\mathscr{X}(v)$, node v can observe random processes $Y(e')$ for all e' in $\Gamma_I(v)$. In general the random process $Y(e)$ transmitted through link $e = (v, u) \in \Gamma_O(v)$ will be a function of both $\mathscr{X}(v)$ and $Y(e')$ if e' is in $\Gamma_I(v)$.

If v is the sink of any connection c, the collection of $\nu(v)$ random processes $\mathscr{Z}(v) = \{Z(v,1), Z(v,2), \ldots, Z(v,\nu(v))\}$ denotes the output at $v = \text{sink}(c)$. A connection $c = (v, v', \mathscr{X}(v, v'))$ is established successfully if a (possibly delayed) copy of $\mathscr{X}(v, v')$ is a subset of $\mathscr{Z}(v')$.

We will make a number of simplifying assumptions:

1) *The capacity of any link in $\mathcal{G}$ is a constant, e.g. m bits per time unit.*
2) *Each link in the communication network has the same delay.*
3) *Random processes $X(v,l)$, $l \in \{1, 2, \ldots, \mu(v)\}$ are independent and have a constant and integral entropy rate of, e.g., m bits per unit time.*
4) *The random processes $X(v,l)$ are independent for different v.*

In addition to the above constraints, we assume that communication in the network is performed by transmission of vectors (symbols) of bits. The length of the vectors is equal in all transmissions and we assume that all links are synchronized with respect to the symbol timing.

Any binary vector of length m can be interpreted as an element in $\mathbb{F}_2^m$, the vector space of binary sequences of length m. The random processes $X(v,l)$, $Y(e)$ and $Z(v,l)$ can hence be modeled as discrete processes $X(v,l) = \{X_0(v,l), X_1(v,l), \ldots\}$, $Y(e) = \{Y_0(e), Y_1(e), \ldots\}$ and $Z(v,l) = \{Z_0(v,l), Z_1(v,l), \ldots\}$, that consist of a sequence of symbols from $\mathbb{F}_2^m$.

A network code is a collection of input output relationships at nodes in the network. In contrast to classical, "routing" communication networks, where random processes that are observed at sources and incoming links are simply forwarded to appropriately chosen outgoing links, in a coded network the random processes on outgoing links can be formed in an arbitrary way from the incoming messages. It is clear that this added freedom cannot yield any solutions that are inferior to the routing solutions. In fact, it is by now well understood that network coding is necessary in order to achieve capacity for the case of arbitrary network problems.

II-C. Linear network codes as state space realizations

An especially useful restriction in network coding is the restriction to only use linear operations in the network. It has been shown by Li et al. [2] that this does not constitute a restriction in the context of multicast communication, i.e. a setup where a single source wants to communicate the same information to a collection of receivers. In other words the multicast scenario is characterized by a set of connections $(u, v', \mathscr{X}(u))$ where v' can range over a set of receiver nodes. In a general setup it is not known if linearity is sufficient to solve any given network coding problem. For a treatment of these issues we refer to [12], [13] for a discussion of related questions. For completeness we give a definition of linearity which is suitable for out purpose. The definition goes beyond the definition of linearity originally given in [3] by allowing arbitrary vector spaces (rather than finite fields) as alphabets for communication. In particular this includes the possibility of time sharing as discussed in [12], [14], [13].

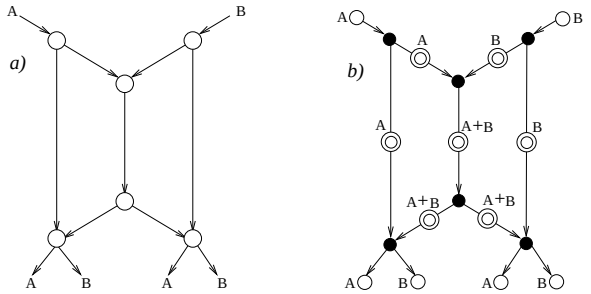

Fig. 2. A network coding problem and its equivalent state space realization. a) Bits A and B have to be communicated to two sink nodes. The transmission is possible only if the information is transmitted in a coded fashion through the bottleneck link in the middle of the network. b) An equivalent state space realization. In particular state space sizes correspond to the rates of transmission on particular links.

Definition 3 *Let a network code be given on an arbitrary communication network. We call a network code linear if*

1) *All messages communicated on links are equipped with the structure of a vector space.*
2) *The set of vectors of local input/output symbols that is allowed by the local input/output relationships at any node in the network constitutes a linear space.* ■

The above definition is very general and encompasses a number of interesting scenarios. In particular, continuous operation of networks is covered by the setup where the communicated symbols are simply elements of the field of formal power series. Other scenarios covered include time sharing between different solutions and the simple case where the communicated symbols are just elements of an arbitrary finite or non-finite field.

Definition 3 clearly resembles the definition if a linear state space realization where we required that local behaviors $\mathscr{C}_u$ should carry the algebraic structure of a linear space. In fact, it is straight forward to make this connection precise. We will do so in the remainder of this section using a (by now) standard example of network coding depicted in Figure 2.

A slightly more general form of network coding would be obtained for "group network coding" where the communicated alphabets carry a group structure and the restriction at a intermediate node allows subgroups of the direct product of the adjacent groups. However, this technicality adds little insight to the here presented results

It should be clear from Figure 2 how the separate elements of a communications network correspond to the elements of a state space realization. I particular, sink and source nodes correspond to visible or symbol nodes. The transmission links in the communication network may be interpreted as communicating a state value from one node in the network to another. Finally, the encoding and decoding that is achieved in a network at intermediate nodes is subsumed by the notion of a local behavior in state space realizations. While state space realizations usually are considered as undirected any sense of direction that may be inherited from a network is included naturally in its description. Thus the overall behavior or code that is implemented in the graph of Figure 2 has a generator matrix of the form

$$G = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

where the first three positions of codewords in $\mathscr{C}$ correspond to the source and two sinks for bit A and the last three positions correspond to the source and two sinks for bit B.

The question arises which insights can be obtained from the connection between state communication networks and state space realization. This will be the topic of the next section where we leverage a beautiful theorem for state space realizations due to Forney [20] in the context of communication networks.

III. A DUALITY RELATION

Linear state space realizations describe indicator functions for linear spaces. One of the most successful principles for the understanding of linear spaces is the notion of duality. Given a linear space and an inner product $\langle \cdot, \cdot \rangle$ defined as $\langle \underline{a}, \underline{b} \rangle = \sum_i a_i b_i$, the dual space of a linear space $\mathscr{C}$ is defined as $\mathscr{C}^\perp = \{\underline{c} : \langle \underline{c}, \underline{a} \rangle = 0 \; \forall \; \underline{a} \in \mathscr{C}\}$. A natural question arising for state space realizations is how the dual space might be representable by a normal graph. The following theorem gives an answer to this problem.

Theorem 2 (Forney [20]) *Let a normal graph $\mathcal{G}$ be given together with a collection of local behaviors $\mathfrak{C} = \{\mathscr{C}_v : v \in V_f\}$ representing the code $\mathcal{C}$. The same normal graph $\mathcal{G}$ together with the collection of local behaviors $\mathfrak{C}^\perp = \{\mathscr{C}_v^\perp : v \in V_f\}$ represents the dual code $\mathcal{C}^\perp$.* ■

In some situations it may be of interest to reverse a solutions to a network problem. An example would be a scenario where a network supports a number of N individual connections of type $(u_i, v_i, \mathscr{X}(u_i))\; i = 1, 2, \ldots, N$ where $u_i \neq u_j$ and $v_i \neq v_j$ for $i \neq j$.

In the reversed network we assume that all links have changed direction and the set of connections has been replaced by $(v_i, u_i, \mathscr{X}(v_i))\; i = 1, 2, \ldots, N$. Figure 3 give an example of a network and its reversed form.

By the symmetry of the problem it is clear that a solution to the network problem in Figure 3 a) implies a solution to the network problem in Figure 3 b). It is interesting to now that the local behaviors f,f' as well as g and g' satisfy a duality relationship. While the behavior at node f in Figure 3a) is a

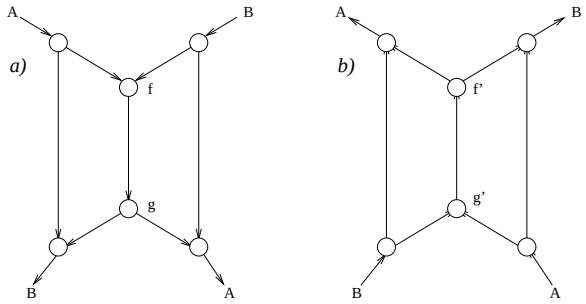

Fig. 3. A network coding problem and its reverse problem.

parity check code the behavior at node f' is a repetition code. A similar ralationship holds for the local behaviors g and g'. Before we proceed to prove the main duality/reversibility claim we will need the following lemma.

Lemma 3 *Let $\mathcal{C}$ be a linear code of dimension k and length n over a field $\mathbb{F}$ and let $\mathcal{I} = \{i_1, i_2, \ldots, i_k\}$ be an information set so that any codeword position c_ℓ may be computed as $c_\ell = \sum_{j=1}^k \alpha_j^{(\ell)} c_{i_j}$, $\alpha_j^{(\ell)} \in \mathbb{F}$. The set $\mathcal{J} = \{1, 2, \ldots, n\} \setminus \mathcal{I}$ is an information set of the dual code $\mathcal{C}^\perp$.*

Proof. This is a straightforward consequence of the fact that a generator matrix for $\mathcal{C}$ can be written (potentially after a permutation of symbols) as $G = (I_k : P)$ and a parity check matrix is then given as $H = (-P^T : I_{n-k})$ for a suitably chosen $k \times (n-k)$ matrix P. ■

Assume we have a network code that locally implements a local behavior $\mathscr{C}_u$. The incoming links on a node u must correspond to an information set of $\mathscr{C}_u$. Indeed if the incoming links do not correspond to an information set we have two possible situations. Either the incoming links carry only a restricted set of symbol combinations in order to satisfy the local behavior $\mathscr{C}_u$ and thus we can reduce the set of locally allowed behaviors further, or the network code allows for random choices at node u which cannot be beneficial in a network coding context and again leads to the possibility to further constrain $\mathscr{C}_u$

Thus we immediately see that dualizing a local code $\mathscr{C}_u$ is naturally accompanied with a reversal of all link directions of links that are incident with a node u. We thus have the following theorem:

Theorem 4 *Let a communication network $\mathcal{G}$ be given together with a desired set of connections $\mathcal{C} = (u_i, v_i, \mathscr{X}(u_i))\; i = 1, 2, \ldots, N$ where $u_i \neq u_j$ and $v_i \neq v_j$ for $i \neq j$. Assume there exists a linear network code that solves the resulting network coding problem. The network is reversible, i.e. after reversing the direction of all links in the network there exists a solution to the network problem posed by the new reversed network and the set of connections $\mathcal{C} = (v_i, u_i, \mathscr{X}(v_i))\; i = 1, 2, \ldots, N$ where $u_i \neq u_j$ and $v_i \neq v_j$ for $i \neq j$. This network coding solution is obtained by dualizing all local behaviors $\mathscr{C}_u$ in the associated state space realization.*

Proof. Most of the theorem is a direct consequence of Forney's duality theorem. Lemma 3 guarantees that reversing link directions and dualizing local behaviors is unproblematic. The theorem then follows from observing that the network code $\mathcal{C}$

that needs to be implemented to solve the original problem has a generator matrix $G = (I_N : I_N)$ where the first N columns correspond to vertices u_i and the last N columns correspond to vertices v_i. Thus, $\mathcal{C}$ is self dual and all that reversing and dualizing the network does, is change the direction of data transmission. ■

Remark 1 The above theorem is fairly general with respect to the alphabet used for transmission of information. In fact, the algebraic framework of [3] may also be used to give an alternative proof of Theorem 4 if the underlying alphabets used for transmission of information carry the algebraic structure of a field. The generality of Forney's duality theorem allows us to cover less restrictive situations. If the alphabet is itself only a vector space, i.e. vectors of length m, the embedded behavior may be interpreted as generated again by a generator matrix $G = (I_N : I_N)$, where however each element in G is itself an $m \times m$ matrix, either an all zero matrix or an identity matrix I_m. The theorem even holds essentially unchanged in the case that the network code is only equipped with merely a group structure. However, in this case the notion of duality has to be extended to the notion of Pontryagin duality, a technicality outside the scope of this paper. ■

It is an intriguing question about the effect of reversing and dualizing a network code which implements a behavior that is different form a collection of disjoint point-to-point connections. In order to keep track of the demands we introduce a $K \times |V|$ binary demand matrix D describing which (if any) of K possible sources node $v \in V$ requires. Let the K sources be given as $u_1, u_2, \ldots, u_K$. The entry $D_{i,v}$ equals one if the network problem contains the connection $(u_i, v, \mathscr{X}(u_i))$ and is zero otherwise. The demand matrix can be easily interpreted in terms of a state-space realization. In fact the behavior $\mathcal{C}$ that has to be imprinted into the network is described by a generator matrix $G = (I_K : D)$ where the identity matrix corresponds to the sources and the matrix D corresponds to the demands. For simplicity we have assumed that any source that also acts as sink has been modeled as two nodes. Dualizing and reversing the network now implements a behavior with generator matrix $H = (D^T : I_N)$ where each previous sink acts as independent source but the original sources now receive the modulo two sum of all the reversed connections. While the utility of this situation at first seems somewhat unclear, we believe that it might have interesting applications. In particular, the reversed multicast scenario might be of independent interest. We formulate the situation in the following theorem:

Theorem 5 *Let a communication network $\mathcal{G}$ be given together with a desired set of (multicast) connections $\mathcal{C} = (u, v_i, \mathscr{X}(u))$ $i = 1, 2, \ldots, N$. Assume there exists a linear network code that solves the resulting network coding problem. The network is reversible, i.e. after reversing the direction of all links in the network and dualizing all local behaviors in the network the communication network implements a situation where N independent sources at the nodes v_i transmit their modulo two sum to the node u.*

A possible application of the scenario of Theorem 5 could be a sensor network where only one of many sensors might observe a certain event. Setting a communication network up as a "reverse multicast" network will efficiently solve the task of communication the occurrence of the event to a central node. A particular intriguing possibility is to set up the multicast network code in a randomized fashion [5], [8] before all operations are *locally* reversed.

IV. Conclusions

Network coding, and in particular *linear* network coding has deep connections to linear system theory and the theory of state space realizations. In this paper we have begun to focus on the connections between these areas and to point out synergistic benefits of considering linear network coding as part of linear system theory. As an example of such benefits we have derived a reversibility theorem for network codes based on a fundamental duality theorem for state space realizations. We believe that many more fruitful connections can be made, in particular, in the context of minimal state space realizations of given behaviors and the question of finding the most efficient network code for a given setup.

References

[1] R. Ahlswede, N. Cai, S.-Y. R. Li and R. W. Yeung, "Network information flow," IEEE Trans. on Information Theory, vol. 46, pp. 1024-1016, 2000.

[2] S.-Y. R. Li, R. W. Yeung, and N. Cai. "Linear network coding". IEEE Trans. on Information Theory , vol 49 pp.371-381, Februray, 2003

[3] R. Koetter, M. Medard, "An Algebraic Approach to Network Coding", Transactions on Networking, October 2003

[4] T. Ho, M. Medard, R. Koetter, "A coding view of network recovery and managment for single receiver communication", CISS 2002

[5] T. Ho, R. Koetter, M. Medard, D. Karger and M. Effros, "The Benefits of Coding over Routing in a Randomized Setting", ISIT 2003

[6] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," IEEE Transactions on Information Theory. Submitted July 2003.

[7] E.Erez and M. Feder, "On Codes for Network Multicast," 41st Annual Allerton Conference on Communication Control and Computing, Oct. 2003.

[8] T. Ho, M. Médard, J. Shi, M. Effros and D. R. Karger, "On Randomized Network Coding", 41st Annual Allerton Conference on Communication Control and Computing, Oct. 2003.

[9] Meir Feder, Dana Ron, Ami Tavory, "Bounds on Linear Codes for Network Multicast". Electronic Colloquium on Computational Complexity (ECCC) 10(033), 2003

[10] R. Dougherty, C. Freiling, and K. Zeger, "Linearity and Solvability in Multicast Networks," submitted to the IEEE Transactions on Information Theory

[11] P. Sanders, S. Egner, L. Tolhuizen, "Polynomial time algorithms for the construction of multicast network codes," 41st Annual Allerton Conference on Communication Control and Computing, Oct. 2003.

[12] A. Rasala-Lehman and E. Lehman, "Complexity Classification of Network Information Flow Problems," 41st Annual Allerton Conference on Communication Control and Computing, Oct. 2003.

[13] S. Riis, "Linear versus non-linear Boolean functions in Network Flow," preprint, November, 2003

[14] M. Medard, M. Effros, T. Ho, D. Karger, "On coding for non-multicast networks," 41st Annual Allerton Conference on Communication Control and Computing, Oct. 2003.

[15] R. Koetter and A. Vardy, "Factor Graphs: Construction, Classification, and Bounds," in *Proceedings of the 1998 IEEE International Symposium on Information Theory*, Cambridge, MA, 1998.

[16] R. Koetter, "On the Representation of Codes in Forney Graphs", Festschrift to the 60th birthday of G.D. Forney, 2001.

[17] R. Koetter and A. Vardy, "Minimality of Factor Graphs," in Proceedings of MTNS 1998, Padua, Italy, 1998.

[18] F. R. Kschischang and V. Sorokine, “On the Trellis Structure of Block Codes,” *IEEE Transactions on Information Theory*, vol. 41, pp. 1924–1937, 1995.
[19] R. Koetter and A. Vardy, “Construction of Minimal Tail-Biting Trellises,” in *Proceedings of the 1998 Information Theory Workshop*, (Killarney), Ireland, pp. 72–74, June 1998.
[20] G. D. Forney, “Codes on Graphs: Normal Realization,” *IEEE Transactions on Information Theory*, vol. IT-47, pp.520–548, Feb. 2001.