Foundations and Trends® in
Communications and Information Theory

Network Coding Theory

Raymond Yeung, S.-Y. R. Li, N. Cai and Z. Zhang

now
the essence of knowledge

Network Coding Theory

Network Coding Theory

Raymond W. Yeung

The Chinese University of Hong Kong
Hong Kong, China
whyeung@ie.cuhk.edu.hk

Shuo-Yen Robert Li

The Chinese University of Hong Kong
Hong Kong, China
bob@ie.cuhk.edu.hk

Ning Cai

Xidian University
Xi'an, Shaanxi, China
caining@mail.xidian.edu.cn

Zhen Zhang

University of Southern California
Los Angeles, CA, USA
zzhang@milly.usc.edu

the essence of knowledge

Boston – Delft

Foundations and Trends® in Communications and Information Theory

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
USA
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

A Cataloging-in-Publication record is available from the Library of Congress

The preferred citation for this publication is R.W. Yeung, S.-Y.R. Li, N. Cai, and Z. Zhang, Network Coding Theory, Foundation and Trends® in Communications and Information Theory, vol 2, nos 4 and 5, pp 241–381, 2005

Printed on acid-free paper

ISBN: 1-933019-24-7

Contents

1

Introduction

1.1 A historical perspective

Consider a network consisting of point-to-point communication channels. Each channel transmits information noiselessly subject to the channel capacity. Data is to be transmitted from the source node to a prescribed set of destination nodes. Given the transmission requirements, a natural question is whether the network can fulfill these requirements and how it can be done efficiently.

In existing computer networks, information is transmitted from the source node to each destination node through a chain of intermediate nodes by a method known as *store-and-forward*. In this method, data packets received from an input link of an intermediate node are stored and a copy is forwarded to the next node via an output link. In the case when an intermediate node is on the transmission paths toward multiple destinations, it sends one copy of the data packets onto each output link that leads to at least one of the destinations. It has been a folklore in data networking that there is no need for data processing at the intermediate nodes except for data replication.

Recently, the fundamental concept of *network coding* was first introduced for satellite communication networks in [211] and then fully

developed in [158], where in the latter the term "network coding" was coined and the advantage of network coding over store-and-forward was first demonstrated, thus refuting the aforementioned folklore. Due to its generality and its vast application potential, network coding has generated much interest in information and coding theory, networking, switching, wireless communications, complexity theory, cryptography, operations research, and matrix theory.

Prior to [211] and [158], network coding problems for special networks had been studied in the context of distributed source coding [207][177][200][212][211]. The works in [158] and [211], respectively, have inspired subsequent investigations of network coding with a single information source and with multiple information sources. The theory of network coding has been developed in various directions, and new applications of network coding continue to emerge. For example, network coding technology is applied in a prototype file-sharing application [176][1]. For a short introduction of the subject, we refer the reader to [173]. For an update of the literature, we refer the reader to the *Network Coding Homepage* [157].

The present text aims to be a tutorial on the basics of the theory of network coding. The intent is a transparent presentation without necessarily presenting all results in their full generality. Part I is devoted to network coding for the transmission from a single source node to other nodes in the network. It starts with describing examples on network coding in the next section. Part II deals with the problem under the more general circumstances when there are multiple source nodes each intending to transmit to a different set of destination nodes.

Compared with the multi-source problem, the single-source network coding problem is better understood. Following [188], the best possible benefits of network coding can very much be achieved when the coding scheme is restricted to just linear transformations. Thus the tools employed in Part I are mostly algebraic. By contrast, the tools employed in Part II are mostly probabilistic.

While this text is not intended to be a survey on the subject, we nevertheless provide at <http://dx.doi.org/10.1561/0100000007>

[1] See [206] for an analysis of such applications.

a summary of the literature (see page 135) in the form of a table according to the following categorization of topics:

1. Linear coding
2. Nonlinear coding
3. Random coding
4. Static codes
5. Convolutional codes
6. Group codes
7. Alphabet size
8. Code construction
9. Algorithms/protocols
10. Cyclic networks
11. Undirected networks
12. Link failure/Network management
13. Separation theorem
14. Error correction/detection
15. Cryptography
16. Multiple sources
17. Multiple unicasts
18. Cost criteria
19. Non-uniform demand
20. Correlated sources
21. Max-flow/cutset/edge-cut bound
22. Superposition coding
23. Networking
24. Routing
25. Wireless/satellite networks
26. Ad hoc/sensor networks
27. Data storage/distribution
28. Implementation issues
29. Matrix theory
30. Complexity theory
31. Graph theory
32. Random graph
33. Tree packing

34. Multicommodity flow
35. Game theory
36. Matriod theory
37. Information inequalities
38. Noisy channels
39. Queueing analysis
40. Rate-distortion
41. Multiple descriptions
42. Latin squares
43. Reversible networks
44. Multiuser channels
45. Joint network-channel coding

1.2 Some examples

Terminology. By a *communication network* we shall refer to a *finite* directed graph, where multiple edges from one node to another are allowed. A node without any incoming edges is called a *source node*. Any other node is called a *non-source node*. Throughout this text, in the figures, a source node is represented by a square, while a non-source node is represented by a circle. An edge is also called a *channel* and represents a noiseless communication link for the transmission of a data unit per unit time. The capacity of direct transmission from a node to a neighbor is determined by the multiplicity of the channels between them. For example, the capacity of direct transmission from the node W to the node X in Figure 1.1(a) is 2. When a channel is from a node X to a node Y, it is denoted as XY.

A communication network is said to be *acyclic* if it contains no directed cycles. Both networks presented in Figures 1.1(a) and (b) are examples of acyclic networks.

A source node generates a message, which is propagated through the network in a multi-hop fashion. We are interested in how much information and how fast it can be received by the destination nodes. However, this depends on the nature of data processing at the nodes in relaying the information.

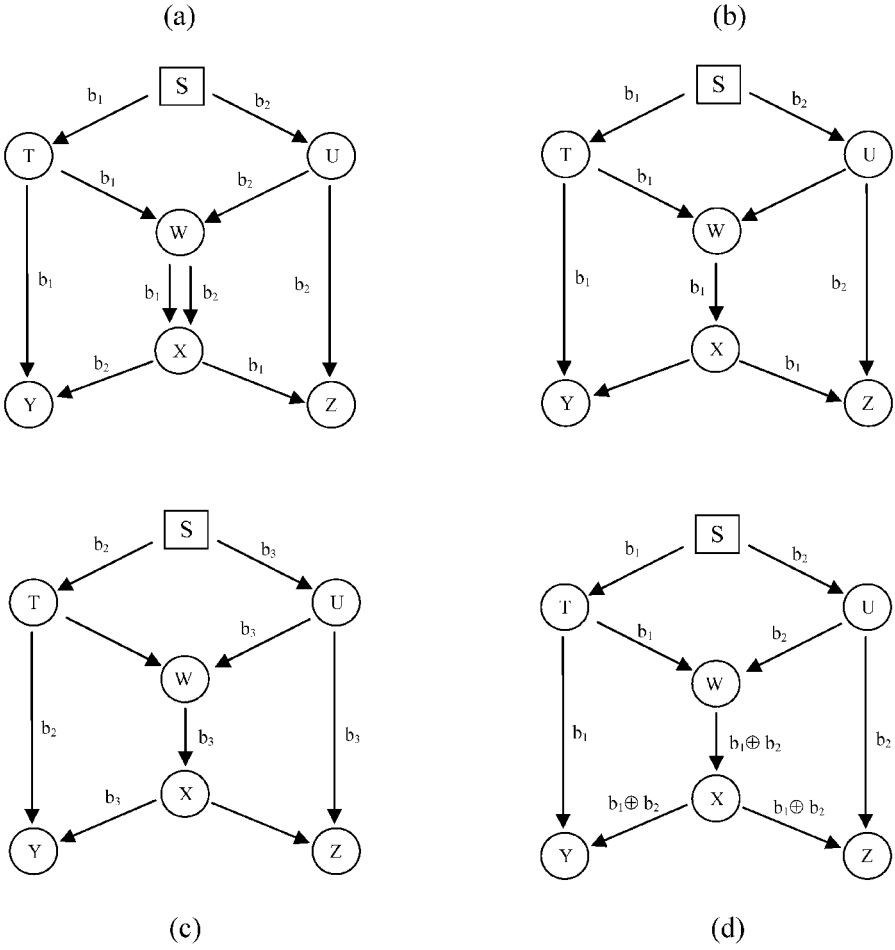

Fig. 1.1 Multicasting over a communication network.

Assume that we multicast two data bits b_1 and b_2 from the source node S to both the nodes Y and Z in the acyclic network depicted by Figure 1.1(a). Every channel carries either the bit b_1 or the bit b_2 as indicated. In this way, every intermediate node simply replicates and sends out the bit(s) received from upstream.

The same network as in Figure 1.1(a) but with one less channel appears in Figures 1.1(b) and (c), which shows a way of multicasting 3 bits b_1, b_2 and b_3 from S to the nodes Y and Z in 2 time units. This

achieves a multicast rate of 1.5 bits per unit time, which is actually the maximum possible when the intermediate nodes perform just bit replication (See [209], Ch. 11, Problem 3). The network under discussion is known as the *butterfly network*.

Example 1.1. (Network coding on the butterfly network) Figure 1.1(d) depicts a different way to multicast two bits from the source node S to Y and Z on the same network as in Figures 1.1(b) and (c). This time the node W derives from the received bits b_1 and b_2 the exclusive-OR bit $b_1 \oplus b_2$. The channel from W to X transmits $b_1 \oplus b_2$, which is then replicated at X for passing on to Y and Z. Then, the node Y receives b_1 and $b_1 \oplus b_2$, from which the bit b_2 can be decoded. Similarly, the node Z decodes the bit b_1 from the received bits b_2 and $b_1 \oplus b_2$. In this way, all the 9 channels in the network are used exactly once.

The derivation of the exclusive-OR bit is a simple form of *coding*. If the same communication objective is to be achieved simply by bit replication at the intermediate nodes without coding, at least one channel in the network must be used twice so that the total number of channel usage would be at least 10. Thus, coding offers the potential advantage of minimizing both latency and energy consumption, and at the same time maximizing the bit rate.

Example 1.2. The network in Figure 1.2(a) depicts the conversation between two parties, one represented by the node combination of S and T and the other by the combination of S' and T'. The two parties send one bit of data to each other through the network in the straightforward manner.

Example 1.3. Figure 1.2(b) shows the same network as in Figure 1.2(a) but with one less channel. The objective of Example 1.2 can no longer be achieved by straightforward data routing but is still achievable if the node U, upon receiving the bits b_1 and b_2, derives the new bit $b_1 \oplus b_2$ for the transmission over the channel UV. As in Example 1.1, the coding mechanism again enhances the bit rate. This

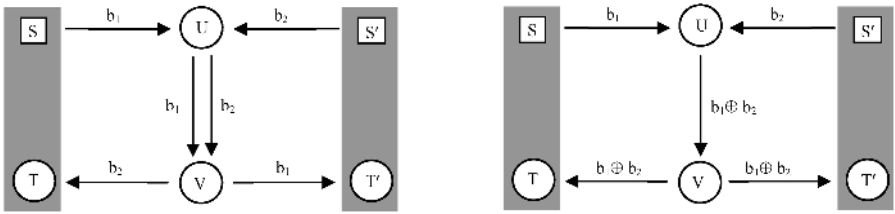

Fig. 1.2 (a) and (b) Conversation between two parties, one represented by the node combination of S and T and the other by the combination of S' and T'.

example of coding at an intermediate node reveals a fundamental fact in information theory first pointed out in [207]: When there are multiple sources transmitting information over a communication network, joint coding of information may achieve higher bit rate than separate transmission.

Example 1.4. Figure 1.3 depicts two neighboring base stations, labeled ST and $S'T'$, of a communication network at a distance twice the wireless transmission range. Installed at the middle is a relay transceiver labeled by UV, which in a unit time either receives or transmits one bit. Through UV, the two base stations transmit one bit of data to each other in three unit times: In the first two unit times, the relay transceiver receives one bit from each side. In the third unit time, it broadcasts the exclusive-OR bit to both base stations, which then can decode the bit from each other. The wireless transmission among the base stations and the relay transceiver can be symbolically represented by the network in Figure 1.2(b).

The principle of this example can readily be generalized to the situation with N-1 relay transceivers between two neighboring base stations at a distance N times the wireless transmission range.

This model can also be applied to satellite communications, with the nodes ST and $S'T'$ representing two ground stations communicating with each other through a satellite represented by the node UV. By employing very simple coding at the satellite as prescribed, the downlink bandwidth can be reduced by 50%.

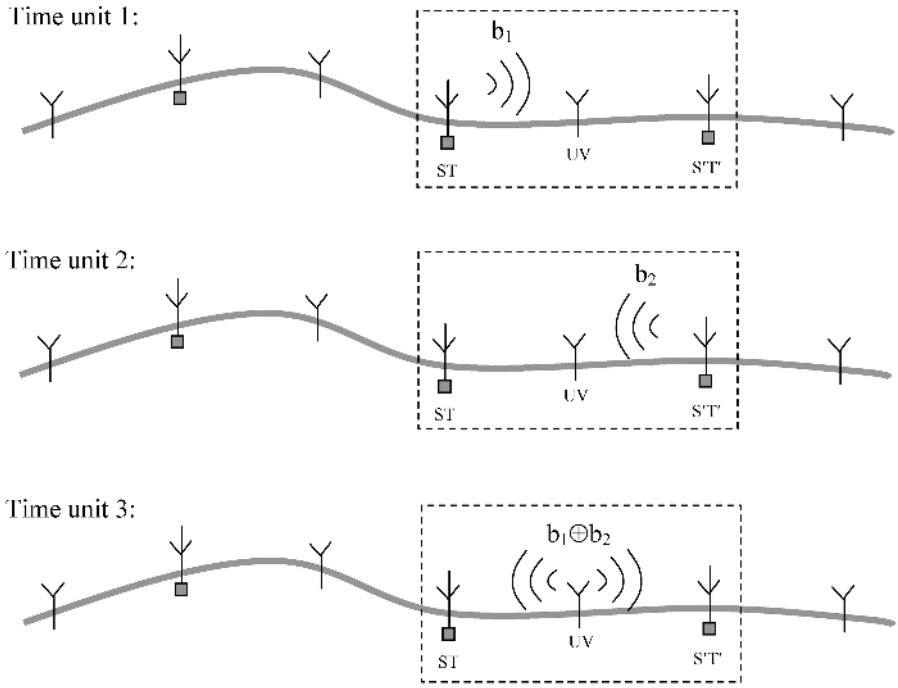

Fig. 1.3 Operation of the relay transceiver between two wireless base stations.

Part I

SINGLE SOURCE

2

Acyclic Networks

A network code can be formulated in various ways at different levels of generality. In a general setting, a source node generates a pipeline of messages to be multicast to certain destinations. When the communication network is *acyclic*, operation at all the nodes can be so synchronized that each message is individually encoded and propagated from the upstream nodes to the downstream nodes. That is, the processing of each message is independent of the sequential messages in the pipeline. In this way, the network coding problem is independent of the propagation delay, which includes the transmission delay over the channels as well as processing delay at the nodes.

On the other hand, when a network contains cycles, the propagation and encoding of sequential messages could convolve together. Thus the amount of delay becomes part of the consideration in network coding.

The present chapter, mainly based on [187], deals with network coding of a *single message* over an acyclic network. Network coding for a whole pipeline of messages over a cyclic network will be discussed in Section 3.

2.1 Network code and linear network code

A communication network is a directed graph[1] allowing multiple edges from one node to another. Every edge in the graph represents a communication channel with the capacity of one data unit per unit time. A node without any incoming edge is a *source node* of the network. There exists at least one source node on every acyclic network. In Part I of the present text, all the source nodes of an acyclic network are combined into one so that there is a unique source node denoted by S on every acyclic network.

For every node T, let $\mathrm{In}(T)$ denote the set of incoming channels to T and $\mathrm{Out}(T)$ the set of outgoing channels from T. Meanwhile, let $\mathrm{In}(S)$ denote a set of *imaginary channels*, which terminate at the source node S but are without originating nodes. The number of these imaginary channels is context dependent and always denoted by ω. Figure 2.1 illustrates an acyclic network with $\omega = 2$ imaginary channels appended at the source node S.

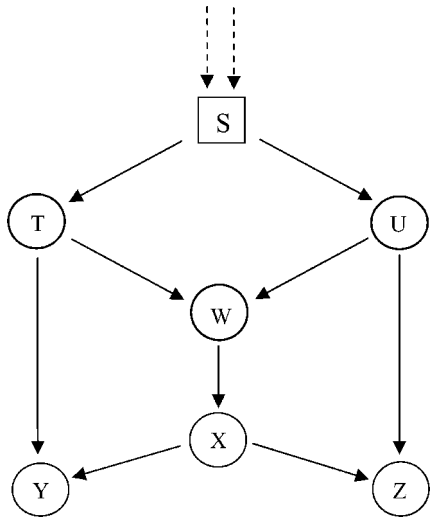

Fig. 2.1 Imaginary channels are appended to a network, which terminate at the source node S but are without originating nodes. In this case, the number of imaginary channels is $\omega = 2$.

[1] Network coding over undirected networks was introduced in [189]. Subsequent works can be found in [185][159][196].

A data unit is represented by an element of a certain *base field* F. For example, $F = GF(2)$ when the data unit is a bit. A message consists of ω data units and is therefore represented by an ω-dimensional row vector $x \in F^\omega$. The source node S generates a message x and sends it out by transmitting a symbol over every outgoing channel. Message propagation through the network is achieved by the transmission of a symbol $\tilde{f}_e(x) \in F$ over every channel e in the network.

A non-source node does not necessarily receive enough information to identify the value of the whole message x. Its encoding function simply maps the ensemble of received symbols from all the incoming channels to a symbol for each outgoing channel. A *network code* is specified by such an encoding mechanism for every channel.

Definition 2.1. (Local description of a network code on an acyclic network) Let F be a finite field and ω a positive integer. An ω-dimensional F-valued *network code* on an acyclic communication network consists of a *local encoding mapping*

$$\tilde{k}_e : F^{|\mathrm{In}(T)|} \to F$$

for each node T in the network and each channel $e \in \mathrm{Out}(T)$.

The acyclic topology of the network provides an upstream-to-downstream procedure for the local encoding mappings to accrue into the values $\tilde{f}_e(x)$ transmitted over all channels e. The above definition of a network code does not explicitly give the values of $\tilde{f}_e(x)$, of which the mathematical properties are at the focus of the present study. Therefore, we also present an equivalent definition below, which describes a network code by both the local encoding mechanisms as well as the recursively derived values $\tilde{f}_e(x)$.

Definition 2.2. (Global description of a network code on an acyclic network) Let F be a finite field and ω a positive integer. An ω-dimensional F-valued *network code* on an acyclic communication network consists of a local encoding mapping $\tilde{k}_e : F^{|\mathrm{In}(T)|} \to F$ and a *global*

encoding mapping $\tilde{f}_e : F^\omega \to F$ for each channel e in the network such that:

(2.1) For every node T and every channel $e \in \mathrm{Out}(T)$, $\tilde{f}_e(x)$ is uniquely determined by $(\tilde{f}_d(x), d \in \mathrm{In}(T))$, and $\tilde{k}_e$ is the mapping via

$$(\tilde{f}_d(x), d \in \mathrm{In}(T)) \mapsto \tilde{f}_e(x).$$

(2.2) For the ω imaginary channels e, the mappings $\tilde{f}_e$ are the projections from the space F^ω to the ω different coordinates, respectively.

Example 2.3. Let $x = (b_1, b_2)$ denote a generic vector in $[GF(2)]^2$. Figure 1.1(d) shows a 2-dimensional binary network code with the following global encoding mappings:

$$\begin{aligned}
\tilde{f}_e(x) &= \mathrm{b}_1 \quad \text{for } e = OS, ST, TW, \text{ and } TY \\
\tilde{f}_e(x) &= \mathrm{b}_2 \quad \text{for } e = OS', SU, UW, \text{ and } UZ \\
\tilde{f}_e(x) &= \mathrm{b}_1 \oplus \mathrm{b}_2 \quad \text{for } e = WX, XY, \text{ and } XZ
\end{aligned}$$

where OS and OS' denote the two imaginary channels in Figure 2.1. The corresponding local encoding mappings are

$$\begin{aligned}
&\tilde{k}_{ST}(b_1, b_2) = b_1,\ \tilde{k}_{SU}(b_1, b_2) = b_2, \\
&\tilde{k}_{TW}(b_1) = \tilde{k}_{TY}(b_1) = b_1, \\
&\tilde{k}_{UW}(b_2) = \tilde{k}_{UZ}(b_2) = b_2,\ \tilde{k}_{WX}(b_1, b_2) = b_1 \oplus b_2,
\end{aligned}$$

etc.

Physical implementation of message propagation with network coding incurs transmission delay over the channels as well as processing delay at the nodes. Nowadays node processing is likely the dominant factor of the total delay in message delivery through the network. It is therefore desirable that the coding mechanism inside a network code be implemented by simple and fast circuitry. For this reason, network codes that involve only linear mappings are of particular interest.

When a global encoding mapping $\tilde{f}_e$ is linear, it corresponds to an ω-dimensional column vector f_e such that $\tilde{f}_e(x)$ is the product $x \cdot f_e$, where the ω-dimensional row vector x represents the message generated by S. Similarly, when a local encoding mapping $\tilde{k}_e$, where $e \in \mathrm{Out}(T)$, is linear, it corresponds to an $|\mathrm{In}(T)|$-dimensional column vector k_e such that $\tilde{k}_e(y) = y \cdot k_e$, where $y \in F^{|\mathrm{In}(T)|}$ is the row vector representing the symbols received at the node T. In an ω-dimensional F-valued network code on an acyclic communication network, if all the local encoding mappings are linear, then so are the global encoding mappings since they are functional compositions of the local encoding mappings. The converse is also true and formally proved in Appendix A: If the global encoding mappings are all linear, then so are the local encoding mappings.

Let a pair of channels (d, e) be called an *adjacent pair* when there exists a node T with $d \in \mathrm{In}(T)$ and $e \in \mathrm{Out}(T)$. Below, we formulate a *linear network code* as a network code where all the local and global encoding mappings are linear. Again, both the local and global descriptions are presented even though they are equivalent. A linear network code was originally called a "linear-code multicast (LCM)" in [188].

Definition 2.4. (Local description of a linear network code on an acyclic network) Let F be a finite field and ω a positive integer. An ω-dimensional F-valued *linear network code* on an acyclic communication network consists of a scalar $k_{d,e}$, called the *local encoding kernel*, for every adjacent pair (d, e). Meanwhile, the local encoding kernel at the node T means the $|\mathrm{In}(T)| \times |\mathrm{Out}(T)|$ matrix $K_T = [k_{d,e}]_{d \in \mathrm{In}(T), e \in \mathrm{Out}(T)}$.

Note that the matrix structure of K_T implicitly assumes some ordering among the channels.

Definition 2.5. (Global description of a linear network code on an acyclic network) Let F be a finite field and ω a positive integer. An ω-dimensional F-valued *linear network code* on an acyclic communication network consists of a scalar $k_{d,e}$ for every adjacent pair

(d, e) in the network as well as an ω-dimensional column vector f_e for every channel e such that:

(2.3) $f_e = \sum_{d \in \mathrm{In}(T)} k_{d,e} f_d$, where $e \in \mathrm{Out}(T)$.
(2.4) The vectors f_e for the ω imaginary channels $e \in \mathrm{In}(S)$ form the natural basis of the vector space F^ω.

The vector f_e is called the *global encoding kernel* for the channel e.

Let the source generate a message x in the form of an ω-dimensional row vector. A node T receives the symbols $x \cdot f_d$, $d \in \mathrm{In}(T)$, from which it calculates the symbol $x \cdot f_e$ for sending onto each channel $e \in \mathrm{Out}(T)$ via the linear formula

$$x \cdot f_e = x \cdot \sum_{d \in \mathrm{In}(T)} k_{d,e} f_d = \sum_{d \in \mathrm{In}(T)} k_{d,e} (x \cdot f_d),$$

where the first equality follows from (2.3).

Given the local encoding kernels for all the channels in an acyclic network, the global encoding kernels can be calculated recursively in any upstream-to-downstream order by (2.3), while (2.4) provides the boundary conditions.

Remark 2.6. A partial analogy can be drawn between the global encoding kernels f_e for the channels in a linear network code and the columns of a generator matrix of a linear error-correcting code [161][190][162][205]. The former are indexed by the channels in the network, while the latter are indexed by "time." However, the mappings f_e must abide by the law of information conservation dictated by the network topology, i.e., (2.3), while the columns in the generator matrix of a linear error-correcting code in general are not subject to any such constraint.

Example 2.7. Example 2.3 translates the solution in Example 1.1 into a network code over the network in Figure 2.1. This network code is in fact linear. Assume the alphabetical order among the channels $OS, OS', ST, \ldots, XZ$. Then, the local encoding kernels at nodes are the

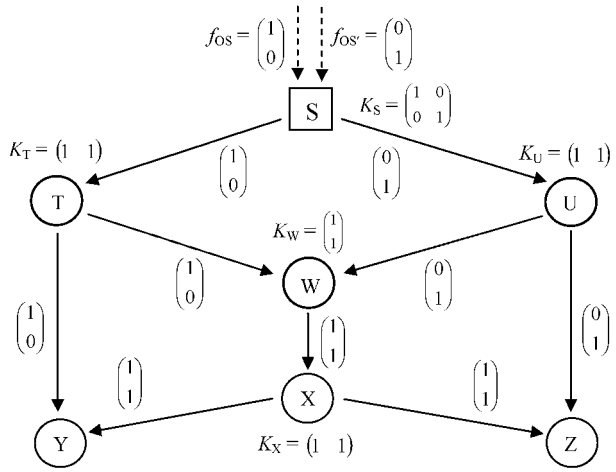

Fig. 2.2 The global and local encoding kernels in the 2-dimensional linear network code in Example 2.7.

following matrices:

$$K_S = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix},\ K_T = K_U = K_X = \begin{bmatrix}1 & 1\end{bmatrix},\ K_W = \begin{bmatrix}1\\1\end{bmatrix}.$$

The corresponding global encoding kernels are:

$$f_e = \begin{cases} \begin{bmatrix}1\\0\end{bmatrix} \text{ for } e = OS, ST, TW, \text{ and } TY \\ \\ \begin{bmatrix}0\\1\end{bmatrix} \text{ for } e = OS', SU, UW, \text{ and } UZ \\ \\ \begin{bmatrix}1\\1\end{bmatrix} \text{ for } e = WX, XY, \text{ and } XZ. \end{cases}$$

The local/global encoding kernels are summarized in Figure 2.2. In fact, they describe a 2-dimensional network code regardless of the choice of the base field.

Example 2.8. For a general 2-dimensional linear network code on the network in Figure 2.2, the local encoding kernels at the nodes can be expressed as

$$K_S = \begin{bmatrix} n & q \\ p & r \end{bmatrix},\ K_T = \begin{bmatrix} s & t \end{bmatrix},\ K_U = \begin{bmatrix} u & v \end{bmatrix},$$

$$K_W = \begin{bmatrix} w \\ x \end{bmatrix},\ K_X = \begin{bmatrix} y & z \end{bmatrix},$$

where n, p, q, ..., z are indeterminates. Starting with $f_{OS} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $f_{OS'} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, we calculate the global encoding kernels recursively as follows:

$$f_{ST} = \begin{bmatrix} n \\ p \end{bmatrix},\ f_{SU} = \begin{bmatrix} q \\ r \end{bmatrix},\ f_{TW} = \begin{bmatrix} ns \\ ps \end{bmatrix},\ f_{TY} = \begin{bmatrix} nt \\ pt \end{bmatrix},$$

$$f_{UW} = \begin{bmatrix} qu \\ ru \end{bmatrix},\ f_{UZ} = \begin{bmatrix} qv \\ rv \end{bmatrix},\ f_{WX} = \begin{bmatrix} nsw + qux \\ psw + rux \end{bmatrix},$$

$$f_{XY} = \begin{bmatrix} nswy + quxy \\ pswy + ruxy \end{bmatrix},\ f_{XZ} = \begin{bmatrix} nswz + quxz \\ pswz + ruxz \end{bmatrix}.$$

The above local/global encoding kernels are summarized in Figure 2.3.

2.2 Desirable properties of a linear network code

Data flow with any conceivable coding schemes at an intermediate node abides with the *law of information conservation*: the content of information sent out from any group of non-source nodes must be derived from the accumulated information received by the group from outside. In particular, the content of any information coming out of a non-source node must be derived from the accumulated information received by that node. Denote the maximum flow from S to a non-source node T

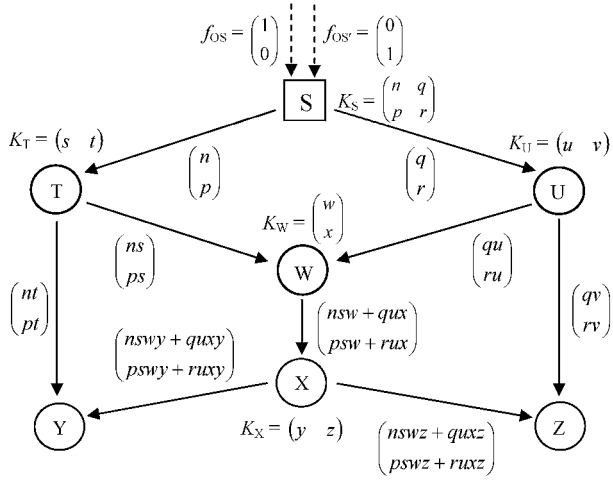

Fig. 2.3 Local/global encoding kernels of a general 2-dimensional linear network code.

as maxflow(T). From the Max-flow Min-cut Theorem, the information rate received by the node T obviously cannot exceed maxflow(T). (See for example [195] for the definition of a maximum flow and the Max-flow Min-cut Theorem.) Similarly, denote the maximum flow from S to a collection $\wp$ of non-source nodes as maxflow($\wp$). Then, the information rate from the source node to the collection $\wp$ cannot exceed maxflow($\wp$).

Whether this upper bound is achievable depends on the network topology, the dimension ω, and the coding scheme. Three special classes of linear network codes are defined below by the achievement of this bound to three different extents. The conventional notation $\langle\cdot\rangle$ for the linear span of a set of vectors will be employed.

Definition 2.9. Let vectors f_e denote the global encoding kernels in an ω-dimensional F-valued linear network code on an acyclic network. Write $V_T = \langle\{f_e : e \in \mathrm{In}(T)\}\rangle$. Then, the linear network code qualifies as a *linear multicast*, a *linear broadcast*, or a *linear dispersion*, respectively, if the following statements hold:

(2.5) $\dim(V_T) = \omega$ for every non-source node T with maxflow(T) $\geq \omega$.

(2.6) $\dim(V_T) = \min\{\omega, \text{maxflow}(T)\}$ for every non-source node T.
(2.7) $\dim(\langle \cup_{T \in \wp} V_T \rangle) = \min\{\omega, \text{maxflow}(\wp)\}$ for every collection $\wp$ of non-source nodes.

In the existing literature, the terminology of a "linear network code" is often associated with a given set of "sink nodes" with $\text{maxflow}(T) \geq \omega$ and requires that $\dim(V_T) = \omega$ for every sink T. Such terminology in the strongest sense coincides with a "linear network multicast" in the above definition.

Clearly, (2.7) $\Rightarrow$ (2.6) $\Rightarrow$ (2.5). Thus, every linear dispersion is a linear broadcast, and every linear broadcast is a linear multicast. The example below shows that a linear broadcast is not necessarily a linear dispersion, a linear multicast is not necessarily a linear broadcast, and a linear network code is not necessarily a linear multicast.

Example 2.10. Figure 2.4(a) presents a 2-dimensional linear dispersion on an acyclic network by prescribing the global encoding kernels. Figure 2.4(b) presents a 2-dimensional linear broadcast on the same network that is not a linear dispersion because $\text{maxflow}(\{T, U\}) = 2 = \omega$ while the global encoding kernels for the channels in $\text{In}(T) \cup \text{In}(U)$ span only a 1-dimensional space. Figure 2.4(c) presents a 2-dimensional linear multicast that is not a linear broadcast since the node U receives no information at all. Finally, the 2-dimensional linear network code in Figure 2.4(d) is not a linear multicast.

When the source node S transmits a message of ω data units into the network, a receiving node T obtains sufficient information to decode the message if and only if $\dim(V_T) = \omega$, of which a necessary prerequisite is that $\text{maxflow}(T) \geq \omega$. Thus, an ω-dimensional linear multicast is useful in multicasting ω data units of information to all those non-source nodes T that meet this prerequisite.

A linear broadcast and a linear dispersion are useful for more elaborate network applications. When the message transmission is through a linear broadcast, every non-source node U with $\text{maxflow}(U) < \omega$ receives partial information of $\text{maxflow}(U)$ units, which may be designed to outline the message in more compressed encoding, at a

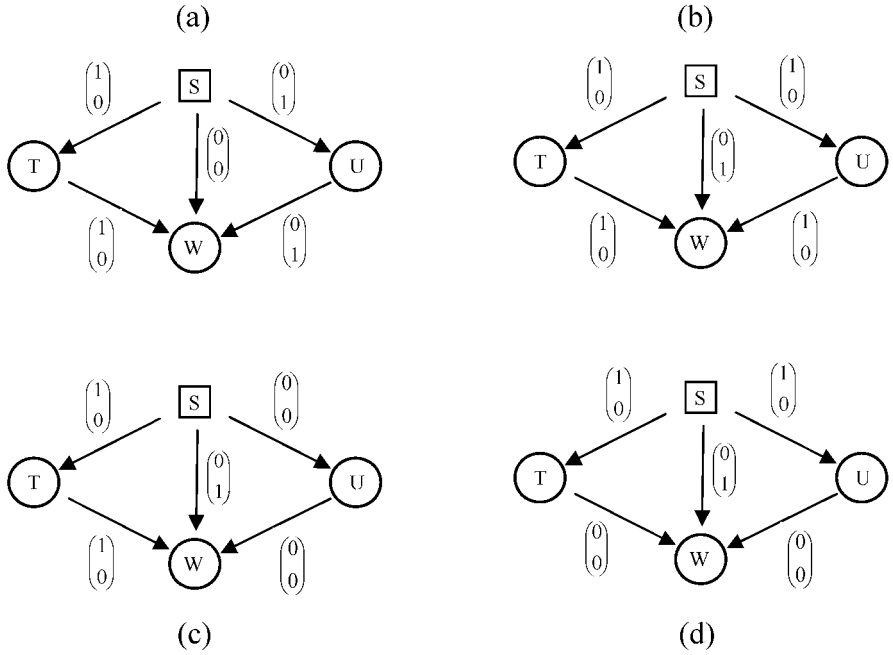

Fig. 2.4 (a) A 2-dimensional binary linear dispersion over an acyclic network, (b) a 2-dimensional linear broadcast that is not a linear dispersion, (c) a 2-dimensional linear multicast that is not a linear broadcast, and (d) a 2-dimensional linear network code that is not a linear multicast.

lower resolution, with less error-tolerance and security, etc. An example of application is when the partial information reduces a large image to the size for a mobile handset or renders a colored image in black and white. Another example is when the partial information encodes ADPCM voice while the full message attains the voice quality of PCM (see [178] for an introduction to PCM and ADPCM). Design of linear multicasts for such applications may have to be tailored to network specifics. Most recently, a combined application of linear broadcast and directed diffusion [182] in sensor networks has been proposed [204].

A potential application of a linear dispersion is in the scalability of a 2-tier broadcast system herein described. There is a backbone network and a number of local area networks (LANs) in the system. A single source presides over the backbone, and the gateway of every LAN is connected to backbone node(s). The source requires a connection to

the gateway of every LAN at the minimum data rate ω in order to ensure proper reach to LAN users. From time to time a new LAN is appended to the system. Suppose that there exists a linear broadcast over the backbone network. Then ideally the new LAN gateway should be connected to a backbone node T with $\text{maxflow}(T) \geq \omega$. However, it may so happen that no such node T is within the vicinity to make the connection economically feasible. On the other hand, if the linear broadcast is in fact a linear dispersion, then it suffices to connect the new LAN gateway to any collection $\wp$ of backbone nodes with $\text{maxflow}(\wp) \geq \omega$.

In real implementation, in order that a linear multicast, a linear broadcast, or a linear dispersion can be used as intended, the global encoding kernels $f_e, e \in \text{In}(T)$ must be available to each node T. In case this information is not available, with a small overhead in bandwidth, the global encoding kernel f_e can be sent along with the value $\tilde{f}_e(x)$ on each channel e, so that at a node T, the global encoding kernels $f_e, e \in \text{Out}(T)$ can be computed from $f_d, d \in \text{In}(T)$ via (2.3) [179].

Example 2.11. The linear network code in Example 2.7 meets all the criteria (2.5) through (2.7) in Definition 2.5. Thus it is a 2-dimensional linear dispersion, and hence also a linear broadcast and linear multicast, regardless of the choice of the base field.

Example 2.12. The more general linear network code in Example 2.8 meets the criterion (2.5) for a linear multicast when

- f_{TW} and f_{UW} are linearly independent;
- f_{TY} and f_{XY} are linearly independent;
- f_{UZ} and f_{XZ} are linearly independent.

Equivalently, the criterion says that s, t, u, v, y, z, $nr - pq$, $npsw + nrux - pnsw - pqux$, and $rnsw + rqux - qpsw - qrux$ are all nonzero. Example 2.7 has been the special case with

$$n = r = s = t = u = v = w = x = y = z = 1$$

and

$$p = q = 0.$$

The requirements (2.5), (2.6), and (2.7) that qualify a linear network code as a linear multicast, a linear broadcast, and a linear dispersion, respectively, state at three different levels of strength that the global encoding kernels f_e span the maximum possible dimensions. Imagine that if the base field F were replaced by the real field $\mathbf{R}$. Then arbitrary infinitesimal perturbation of local encoding kernels $k_{d,e}$ in any given linear network code would place the vectors f_e at "general positions" with respect to one another in the space $\mathbf{R}^\omega$. Generic positions maximize the dimensions of various linear spans by avoiding linear dependence in every conceivable way. The concepts of generic positions and infinitesimal perturbation do not apply to the vector space F^ω when F is a finite field. However, when F is almost infinitely large, we can emulate this concept of avoiding unnecessary linear dependence.

One way to construct a linear multicast/broadcast/dispersion is by considering a linear network code in which every collection of global encoding kernels that can possibly be linearly independent is linearly independent. This motivates the following concept of a *generic linear network code*.

Definition 2.13. Let F be a finite field and ω a positive integer. An ω-dimensional F-valued linear network code on an acyclic communication network is said to be *generic* if:

(2.8) Let $\{e_1, e_2, \ldots, e_m\}$ be an arbitrary set of channels, where each $e_j \in \mathrm{Out}(T_j)$. Then, the vectors $f_{e_1}, f_{e_2}, \ldots, f_{e_m}$ are linearly independent (and hence $m \le \omega$) provided that

$$\langle \{f_d : d \in \mathrm{In}(T_j)\} \rangle \not\subset \langle \{f_{e_k} : k \ne j\} \rangle \text{ for } 1 \le j \le m.$$

Linear independence among f_{e_1}, f_{e_2}, $\ldots$, f_{e_m} is equivalent to that $f_{e_j} \notin \langle \{f_{e_k} : k \ne j\} \rangle$ for all j, which implies that $\langle \{f_d : d \in \mathrm{In}(T_j)\} \rangle \not\subset \langle \{f_{e_k} : k \ne j\} \rangle$. Thus the requirement (2.8), which is the converse of

the above implication, indeed says that any collection of global encoding kernels that can possibly be linearly independent must be linearly independent.

Remark 2.14. In Definition 2.13, suppose all the nodes T_j are equal to some node T. If the linear network code is generic, then for any collection of no more than $\dim(V_T)$ outgoing channels from T, the corresponding global encoding kernels are linearly independent. In particular, if $|\mathrm{Out}(T)| \leq \dim(V_T)$, then the global encoding kernels of all the outgoing channels from T are linearly independent.

Theorem 2.21 in the next section will prove the existence of a generic linear network code when the base field F is sufficiently large. Theorem 2.29 will prove every generic linear network code to be a linear dispersion. Thus, a generic network code, a linear dispersion, a linear broadcast, and a linear multicast are notions of decreasing strength in this order with regard to linear independence among the global encoding kernels. The existence of a generic linear network code then implies the existence of the rest.

Note that the requirement (2.8) of a generic linear network code is purely in terms of linear algebra and does not involve the notion of maximum flow. Conceivably, other than (2.5), (2.6) and (2.7), new conditions about linear independence among global encoding kernels might be proposed in the future literature and might again be entailed by the purely algebraic requirement (2.8).

On the other hand, a linear dispersion on an acyclic network does not necessarily qualify for a generic linear network code. A counterexample is as follows.

Example 2.15. The 2-dimensional binary linear dispersion on the network in Figure 2.5 is a not a generic linear network code because the global encoding kernels of two of the outgoing channels from the source node S are equal to $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$, a contradiction to the remark following Definition 2.13.

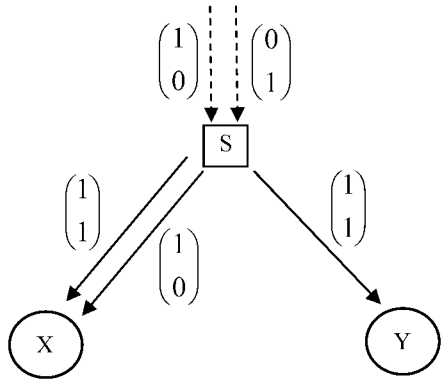

Fig. 2.5 A 2-dimensional linear dispersion that is not a generic linear network code.

2.3 Existence and construction

The following three factors dictate the existence of an ω-dimensional F-valued generic linear network code, linear dispersion, linear broadcast, and linear multicast on an acyclic network:

- the value of ω,
- the network topology,
- the choice of the base field F.

We begin with an example illustrating the third factor.

Example 2.16. On the network in Figure 2.6, a 2-dimensional ternary linear multicast can be constructed by the following local encoding kernels at the nodes:

$$K_S = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 2 \end{bmatrix} \quad \text{and} \quad K_{U_i} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

for $i = 1$ to 4. On the other hand, we can prove the nonexistence of a 2-dimensional binary linear multicast on this network as follows. Assuming to the contrary that a 2-dimensional binary linear multicast exists, we shall derive a contradiction. Let the global encoding kernel $f_{SU_i} = \begin{bmatrix} y_i \\ z_i \end{bmatrix}$ for $i = 1$ to 4. Since $\text{maxflow}(T_k) = 2$ for all $k = 1$ to 6,

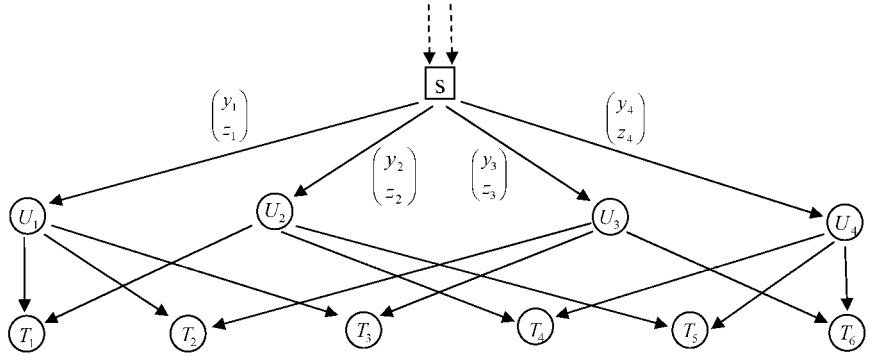

Fig. 2.6 A network with a 2-dimensional ternary linear multicast but without a 2-dimensional binary linear multicast.

the global encoding kernels for the two incoming channels to each node T_k must be linearly independent. Thus, if T_k is at the downstream of both U_i and U_j, then the two vectors $\begin{bmatrix} y_i \\ z_i \end{bmatrix}$ and $\begin{bmatrix} y_j \\ z_j \end{bmatrix}$ must be linearly independent. Each node T_k is at the downstream of a different pair of nodes among U_1, U_2, U_3, and U_4. Therefore, the four vectors $\begin{bmatrix} y_i \\ z_i \end{bmatrix}$, $i = 1$ to 4, are pairwise linearly independent, and consequently, must be four distinct vectors in $GF(2)^2$. Thus, one of them must be $\begin{bmatrix} 0 \\ 0 \end{bmatrix}$, as there are only four vectors in $GF(2)^2$. This contradicts the pairwise linear independence among the four vectors.

In order for the linear network code to qualify as a linear multicast, a linear broadcast, or a linear dispersion, it is required that certain collections of global encoding kernels span the maximum possible dimensions. This is equivalent to certain polynomial functions taking nonzero values, where the indeterminates of these polynomials are the local encoding kernels. To fix ideas, take $\omega = 3$ and consider a node T with two incoming channels. Put the global encoding kernels for these two channels in juxtaposition to form a 3×2 matrix. Then, this matrix attains the maximum possible rank of 2 if and only if there exists a 2×2 submatrix with nonzero determinant.

According to the local description, a linear network code is specified by the local encoding kernels and the global encoding kernels can be derived recursively in the upstream-to-downstream order. From Example 2.11, it is not hard to see that every component in a global encoding kernel is a polynomial function whose indeterminates are the local encoding kernels.

When a nonzero value of such a polynomial function is required, it does not merely mean that at least one coefficient in the polynomial is nonzero. Rather, it means a way to choose scalar values for the indeterminates so that the polynomial function assumes a nonzero scalar value.

When the base field is small, certain polynomial equations may be unavoidable. For instance, for any prime number p, the polynomial equation $z^p - z = 0$ is satisfied for any $z \in GF(p)$. The nonexistence of a binary linear multicast in Example 2.16 can also trace its root to a set of polynomial equations that cannot be avoided simultaneously over $GF(2)$.

However, when the base field is sufficiently large, every nonzero polynomial function can indeed assume a nonzero value with a proper choice of the values taken by the set of indeterminates involved. This is asserted by the following elementary proposition, which will be instrumental in the alternative proof of Corollary 2.24 asserting the existence of a linear multicast on an acyclic network when the base field is sufficiently large.

Lemma 2.17. Let $g(z_1, z_2, \ldots, z_n)$ be a nonzero polynomial with coefficients in a field F. If $|F|$ is greater than the degree of g in every z_j, then there exist $a_1, a_2, \ldots, a_n \in F$ such that $g(a_1, a_2, \ldots, a_n) \neq 0$.

Proof. The proof is by induction on n. For $n = 0$, the proposition is obviously true, and assume that it is true for $n - 1$ for some $n \geq 1$. Express $g(z_1, z_2, \ldots, z_n)$ as a polynomial in z_n with coefficients in the polynomial ring $F[z_1, z_2, \ldots, z_{n-1}]$, i.e.,

$$g(z_1, z_2, \ldots, z_n) = h(z_1, z_2, \ldots, z_{n-1}){z_n}^k + \ldots,$$

where k is the degree of g in z_n and the leading coefficient $h(z_1, z_2, \ldots, z_{n-1})$ is a nonzero polynomial in $F[z_1, z_2, \ldots, z_{n-1}]$.

By the induction hypothesis, there exist $a_1, a_2, \ldots, a_{n-1} \in E$ such that $h(a_1, a_2, \ldots, a_{n-1}) \neq 0$. Thus $g(a_1, a_2, \ldots, a_{n-1}, z)$ is a nonzero polynomial in z with degree $k < |F|$. Since this polynomial cannot have more than k roots in F and $|F| > k$, there exists $a_n \in F$ such that

$$g(a_1, a_2, \ldots, a_{n-1}, a_n) \neq 0. \qquad \square$$

Example 2.18. Recall the 2-dimensional linear network code in Example 2.8 that is expressed in the 12 indeterminates $n, p, q, \ldots, z$. Place the vectors f_{TW} and f_{UW} in juxtaposition into the 2×2 matrix

$$L_W = \begin{bmatrix} ns & qu \\ ps & ru \end{bmatrix},$$

the vectors f_{TY} and f_{XY} into the 2×2 matrix

$$L_Y = \begin{bmatrix} nt & nswy + quxy \\ pt & pswy + ruxy \end{bmatrix},$$

and the vectors f_{UZ} and f_{XZ} into the 2×2 matrix

$$L_Z = \begin{bmatrix} nswz + quxz & qv \\ pswz + ruxz & rv \end{bmatrix}.$$

Clearly,

$$\det(L_W) \cdot \det(L_Y) \cdot \det(L_Z) \neq 0$$

in $F[n, p, q, \ldots, z]$. Applying Lemma 2.17 to $F[n, p, q, \ldots, z]$, we can set scalar values for the 12 indeterminates so that

$$\det(L_W) \cdot \det(L_Y) \cdot \det(L_Z) \neq 0$$

when the field F is sufficiently large. These scalar values then yield a 2-dimensional F-valued linear multicast. In fact,

$$\det(L_W) \cdot \det(L_Y) \cdot \det(L_Z) = 1$$

when

$$p = q = 0$$

and

$$n = r = s = t = \cdots = z = 1.$$

Therefore, the 2-dimensional linear network code depicted in Figure 2.2 is a linear multicast, and this fact is regardless of the choice of the base field F.

Algorithm 2.19. (Construction of a generic linear network code) Let a positive integer ω and an acyclic network with N channels be given. This algorithm constructs an ω-dimensional F-valued linear network code when the field F contains more than $\binom{N+\omega-1}{\omega-1}$ elements. The following procedure prescribes global encoding kernels that form a generic linear network code.

{
  // By definition, the global encoding kernels for the ω
  // imaginary channels form the standard basis of F^ω.
  for (every channel e in the network except for the imaginary channels)
    f_e = the zero vector;
    // This is just initialization.
    // f_e will be updated in an upstream-to-downstream order.
  for (every node T, following an upstream-to-downstream order)
  {
    for (every channel $e \in \mathrm{Out}(T)$)
    {
      // Adopt the abbreviation $V_T = \langle \{ f_d : d \in \mathrm{In}(T) \} \rangle$ as before.
      Choose a vector w in the space V_T such that $w \notin \langle \{ f_d : d \in \xi \} \rangle$, where ξ is any collection of $\omega - 1$ channels, including possibly imaginary channels in $\mathrm{In}(S)$ but excluding e, with $V_T \not\subset \langle \{ f_d : d \in \xi \} \rangle$;
      // To see the existence of such a vector w, denote $\dim(V_T)$
      // by k. If ξ is any collection of $\omega - 1$ channels with $V_T \not\subset$
      // $\langle \{ f_d : d \in \xi \} \rangle$, then $\dim(V_T) \cap \langle \{ f_d : d \in \xi \} \rangle \leq k - 1$.
      // There are at most $\binom{N+\omega-1}{\omega-1}$ such collections ξ. Thus,
      // $|V_T \cap (\cup_\xi \langle \{ f_d : d \in \xi \} \rangle)| \leq \binom{N+\omega-1}{\omega-1} |F|^{k-1} < |F|^k = |V_T|$.

```
            f_e = w;
              // This is equivalent to choosing scalar values for local
              // encoding kernels k_{d,e} for all d such that Σ_{d∈In(T)} k_{d,e} f_d ∉
              // ⟨{f_d : d ∈ ξ}⟩ for every collection ξ of channels with
              // V_T ⊄ ⟨{f_d : d ∈ ξ}⟩.
          }
      }
}
```

Justification. We need to show that the linear network code constructed by Algorithm 2.19 is indeed generic. Let $\{e_1, e_2, \ldots, e_m\}$ be an arbitrary set of channels, excluding the imaginary channels in $\mathrm{In}(S)$, where $e_j \in \mathrm{Out}(T_j)$ for all j. Assuming that $V_{T_j} \not\subset \langle\{f_{e_k} : k \neq j\}\rangle$ for all j, we need to prove the linear independence among the vectors $f_{e_1}, f_{e_2}, \ldots, f_{e_m}$.

Without loss of generality, we may assume that f_{e_m} is the last updated global encoding kernel among $f_{e_1}, f_{e_2}, \ldots, f_{e_m}$ in the algorithm, i.e., e_m is last handled by the inner "for loop" among the channels $e_1, e_2, \ldots, e_m$. Our task is to prove (2.8) by induction on m, which is obviously true for $m = 1$. To prove (2.8) for $m \geq 2$, observe that if

$$\langle\{f_d : d \in \mathrm{In}(T_j)\}\rangle \not\subset \langle\{f_{e_k} : k \neq j,\ 1 \leq k \leq m\}\rangle \quad \text{for } 1 \leq j \leq m,$$

then

$$\langle\{f_d : d \in \mathrm{In}(T_j)\}\rangle \not\subset \langle\{f_{e_k} : k \neq j,\ 1 \leq k \leq m-1\}\rangle \quad \text{for } 1 \leq j \leq m-1.$$

By the induction hypothesis, the global encoding kernels $f_{e_1}, f_{e_2}, \ldots, f_{e_{m-1}}$ are linearly independent. Thus it suffices to show that f_{e_m} is linearly independent of $f_{e_1}, f_{e_2}, \ldots, f_{e_{m-1}}$.

Since

$$V_{T_m} \not\subset \{f_{e_k} : 1 \leq k \leq m-1\}$$

and $f_{e_1}, f_{e_2}, \ldots, f_{e_{m-1}}$ are assumed to be linearly independent, we have $m - 1 < \omega$, or $m \leq \omega$. If $m = \omega$, $\{e_1, e_2, \ldots, e_{m-1}\}$ is one of the collections ξ of $\omega - 1$ channels considered in the inner loop of the algorithm. Then f_{e_m} is chosen such that

$$f_{e_m} \notin \langle \{f_{e_1}, f_{e_2}, \ldots, f_{e_{m-1}}\} \rangle,$$

and hence f_{e_m} is linearly independent of f_{e_1}, f_{e_2}, $\ldots$, $f_{e_{m-1}}$.

If $m \leq \omega - 1$, let $\zeta = \{e_1, e_2, \ldots, e_{m-1}\}$, so that $|\zeta| \leq \omega - 2$. Subsequently, we shall expand ζ iteratively so that it eventually contains $\omega - 1$ channels. Initially, ζ satisfies the following conditions:

1. $\{f_d : d \in \zeta\}$ is a linearly independent set;
2. $|\zeta| \leq \omega - 1$;
3. $V_{T_m} \not\subset \langle \{f_d : d \in \zeta\} \rangle$.

Since $|\zeta| \leq \omega - 2$, there exists two imaginary channels b and c in $\mathrm{In}(S)$ such that $\{f_d : d \in \zeta\} \cup \{f_b, f_c\}$ is a linearly independent set. To see the existence of the channels b and c, recall that the global encoding kernels for the imaginary channels in $\mathrm{In}(S)$ form the natural basis for F^ω. If for all imaginary channels b, $\{f_d : d \in \zeta\} \cup \{f_b\}$ is a dependence set, then $f_b \in \langle \{f_d : d \in \zeta\} \rangle$, which implies $F^\omega \subset \langle \{f_d : d \in \zeta\} \rangle$, a contradiction because $|\zeta| \leq \omega - 2 < \omega$. Therefore, such an imaginary channel b exists. To see the existence of the channel c, we only need to replace ζ in the above argument by $\zeta \cup \{b\}$ and to note that $|\zeta| \leq \omega - 1 < \omega$.

Since $\{f_d : d \in \zeta\} \cup \{f_b, f_c\}$ is a linearly independent set,

$$\langle \{f_d : d \in \zeta\} \cup \{f_b\} \rangle \cap \langle \{f_d : d \in \zeta\} \cup \{f_c\} \rangle = \langle \{f_d : d \in \zeta\} \rangle.$$

Then either

$$V_{T_m} \not\subset \langle \{f_d : d \in \zeta\} \cup \{f_b\} \rangle$$

or

$$V_{T_m} \not\subset \langle \{f_d : d \in \zeta\} \cup \{f_c\} \rangle,$$

otherwise

$$V_{T_m} \subset \langle \{f_d : d \in \zeta\} \rangle,$$

a contradiction to our assumption. Now update ζ by replacing it with $\zeta \cup \{b\}$ or $\zeta \cup \{c\}$ accordingly. Then the resulting ζ contains one more channel than before, while it continues to satisfy the three properties it satisfies initially. Repeat this process until $|\zeta| = \omega - 1$, so that ζ is

one of the collections ξ of $\omega - 1$ channels considered in the inner loop of the algorithm. For this collection ξ, the global encoding kernel f_{e_m} is chosen such that

$$f_{e_m} \notin \langle \{ f_d : d \in \xi \} \rangle.$$

As

$$\{ f_{e_1}, f_{e_2}, \ldots, f_{e_{m-1}} \} \subset \xi,$$

we conclude that $\{ f_{e_1}, f_{e_2}, \ldots, f_{e_m} \}$ is an independent set. This complete the justification.

Analysis of complexity. For each channel e, the "for loop" in Algorithm 2.19 processes $\binom{N+\omega-1}{\omega-1}$ collections of $\omega - 1$ channels. The processing includes the detection of those collections ξ with $V_T \not\subset \langle \{ f_d : d \in \xi \} \rangle$ and the calculation of the set $V_T \setminus \cup_\xi \langle \{ f_d : d \in \xi \} \rangle$. This can be done by, for instance, Gaussian elimination. Throughout the algorithm, the total number of collections of $\omega - 1$ channels processed is $N\binom{N+\omega-1}{\omega-1}$, a polynomial in N of degree ω. Thus, for a fixed ω, it is not hard to implement Algorithm 2.19 within a polynomial time in N. This is similar to the polynomial-time implementation of Algorithm 2.31 in the sequel for refined construction of a linear multicast.

Remark 2.20. In [158], nonlinear network codes for multicasting were considered, and it was shown that they can be constructed by a random procedure with high probability for large block lengths. The size of the base field of a linear network code corresponds to the block length of a nonlinear network code. It is not difficult to see from the lower bound on the required field size in Algorithm 2.19 that if a field much larger than sufficient is used, then a generic linear network code can be constructed with high probability by randomly choosing the global encoding kernels. See [179] for a similar result for the special case of linear multicast. The random coding scheme proposed therein has the advantage that code construction can be done independent of the network topology, making it potentially very useful when the network topology is unknown.

While random coding offers simple construction and more flexibility, a much larger base field is usually needed. In some applications, it is

necessary to verify that the code randomly constructed indeed possesses the desired properties. Such a task can be computationally non-trivial.

Algorithm 2.19 constitutes a constructive proof for the following theorem.

Theorem 2.21. Given a positive integer ω and an acyclic network, there exists an ω-dimensional F-valued generic linear network code for sufficiently large base field F.

Corollary 2.22. Given a positive integer ω and an acyclic network, there exists an ω-dimensional F-valued linear dispersion for sufficiently large base field F.

Proof. Theorem 2.29 in the sequel will assert that every generic linear network code is a linear dispersion. □

Corollary 2.23. Given a positive integer ω and an acyclic network, there exists an ω-dimensional F-valued linear broadcast for sufficiently large base field F.

Proof. (2.7) $\Rightarrow$ (2.6). □

Corollary 2.24. Given a positive integer ω and an acyclic network, there exists an ω-dimensional F-valued linear multicast for sufficiently large base field F.

Proof. (2.6) $\Rightarrow$ (2.5). □

Actually, Corollary 2.23 also implies Corollary 2.22 by the following argument. Let a positive integer ω and an acyclic network be given. For every nonempty collection $\wp$ of non-source nodes, install a new node $T_\wp$ and $|\wp|$ channels from every node $T \in \wp$ to this new node. This constructs a new acyclic network. A linear broadcast on the new network incorporates a linear dispersion on the original network.

Similarly, Corollary 2.24 implies Corollary 2.23 by the following argument. Let a positive integer ω and an acyclic network be given. For every non-source node T, install a new node T' and ω incoming channels to this new node, $\min\{\omega, \text{maxflow}(T)\}$ of them from T and the remaining $\omega - \min\{\omega, \text{maxflow}(T)\}$ from S. This constructs a new acyclic network. A linear multicast on the new network then incorporates a linear broadcast on the original network.

The paper [188] gives a computationally less efficient version of Algorithm 2.19, Theorem 2.21, and also proves that every generic linear network code (therein called a "generic LCM") is a linear broadcast. The following alternative proof for Corollary 2.24 is adapted from the approach in [184].

Alternative proof of Corollary 2.24. Let a sequence of channels $e_1, e_2, \ldots, e_m$, where $e_1 \in \text{In}(S)$ and (e_j, e_{j+1}) is an adjacent pair for all j, be called a *path from* e_1 to e_m. For a path $P = (e_1, e_2, \ldots, e_m)$, define

$$K_P = \prod_{1 \le j < m} k_{e_j, e_{j+1}}. \tag{2.9}$$

Calculating by (2.3) recursively from the upstream channels to the downstream channels, it is not hard to find that

$$f_e = \Sigma_{d \in \text{In}(S)} \left(\Sigma_{P\text{: a path from } d \text{ to } e} K_P\right) f_d \tag{2.10}$$

for every channel e (see Example 2.25 below). Thus, every component of every global encoding kernel belongs to $F[*]$. The subsequent arguments in this proof actually depend only on this fact alone but not on the exact form of (2.10). Denote by $F[*]$ the polynomial ring over the field F with all the $k_{d,e}$ as indeterminates, where the total number of such indeterminates is equal to $\Sigma_T \, |\text{In}(T)| \cdot |\text{Out}(T)|$.

Let T be a non-source node with $\text{maxflow}(T) \ge \omega$. Then, there exists ω disjoint paths from the ω imaginary channels to ω distinct channels in $\text{In}(T)$. Putting the global encoding kernels for these ω channels of $\text{In}(T)$ in juxtaposition to form an $\omega \times \omega$ matrix L_T. Claim that

$$\det(L_T) = 1 \text{ for properly set scalar values of the indeterminates.} \tag{2.11}$$

To prove the claim, we set $k_{d,e} = 1$ when both d and e belong to one of the ω channel-disjoint paths with d immediately preceding e, and set $k_{d,e} = 0$ otherwise. With such local encoding kernels, the symbols sent on the ω imaginary channels at S are routed to the node T via the channel-disjoint paths. Thus the columns in L_T are simply global encoding kernels for the imaginary channels, which form the standard basis of the space F^ω. Therefore, $\det(L_T) = 1$, verifying the claim (2.11).

Consequently, $\det(L_T) \neq 0$ in $F[*]$, i.e., $\det(L_T)$ is a nonzero polynomial in the indeterminates $k_{d,e}$. This conclusion applies to every non-source node T with $\text{maxflow}(T) \geq \omega$. Thus

$$\prod_{T:\text{maxflow}(T)\geq\omega} \det(L_T) \neq 0$$

in $F[*]$. Applying Lemma 2.17 to $F[*]$, we can set scalar values for the indeterminates so that

$$\prod_{T:\text{maxflow}(T)\geq\omega} \det(L_T) \neq 0$$

when the field F is sufficiently large, which in turns implies that $\det(L_T) \neq 0$ for all T such that $\text{maxflow}(T) \geq \omega$. These scalar values then yield a linear network code that meets the requirement (2.5) for a linear multicast.

This proof provides an alternative way to construct a linear multicast, using Lemma 2.17 as a subroutine to search for scalars $a_1, a_2, \ldots, a_n \in F$ such that $g(a_1, a_2, \ldots, a_n) \neq 0$ whenever $g(z_1, z_2, \ldots, z_n)$ is a nonzero polynomial over a sufficiently large field F. The straightforward implementation of this subroutine is exhaustive search.

We note that it is straightforward to strengthen this alternative proof for Corollary 2.23 and thereby extend the alternative construction to a linear broadcast.

Example 2.25. We now illustrate (2.10) in the above alternative proof of Corollary 2.24 with the 2-dimensional linear network code in

Example 2.8 that is expressed in the 12 indeterminates $n, p, q, \ldots, z$. The local encoding kernels at the nodes are

$$K_S = \begin{bmatrix} n & q \\ p & r \end{bmatrix},\ K_T = \begin{bmatrix} s & t \end{bmatrix},\ K_U = \begin{bmatrix} u & v \end{bmatrix},$$

$$K_W = \begin{bmatrix} w \\ x \end{bmatrix},\ K_X = \begin{bmatrix} y & z \end{bmatrix}.$$

Starting with $f_{OS} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $f_{OS'} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, we can calculate the global encoding kernels by the formula (2.10). Take f_{XY} as the example. There are two paths from OS to XY and two from OS' to XY. For these paths,

$$K_p = \begin{cases} nswy \\ pswy \\ quxy \\ ruxy \end{cases} \text{ when P is the path } \begin{cases} OS, ST, TW, WX, XY \\ OS', ST, TW, WX, XY \\ OS, SU, UW, WX, XY \\ OS', SU, UW, WX, XY. \end{cases}$$

Thus

$$\begin{aligned} f_{XY} &= (nswy) f_{OS} + (pswy) f_{OS'} + (quxy) f_{OS} + (ruxy) f_{OS'} \\ &= \begin{bmatrix} nswy + quxy \\ pswy + ruxy \end{bmatrix}, \end{aligned}$$

which is consistent with Example 2.8.

The existence of an ω-dimensional F-valued generic linear network code for sufficiently large base field F has been proved in Theorem 2.21 by a construction algorithm, but the proof of the existence of a linear dispersion still hinges on Theorem 2.29 in the sequel, which asserts that every generic linear network code is a linear dispersion. The remainder of the section is dedicated to Theorem 2.29 and its proof. A weaker version of this theorem, namely that a generic linear network code is a linear multicast, was proved in [188].

Notation. Consider a network with ω imaginary channels in $\mathrm{In}(S)$. For every set $\wp$ of nodes in the network, denote by $\mathrm{cut}(\wp)$

the collection of channels that terminates at the nodes in $\wp$ but do not originate from nodes in $\wp$. In particular, $\text{cut}(\wp)$ includes all the imaginary channels when $S \in \wp$.

Example 2.26. For the network in Figure 2.3, $\text{cut}(\{U,X\}) = \{SU, WX\}$ and $\text{cut}(\{S,U,X,Y,Z\}) = \{OS, OS', WX, TY\}$, where OS and OS' stand for the two imaginary channels.

Lemma 2.27. Let f_e denote the global encoding kernel for a channel e in an ω-dimensional linear network code on an acyclic network. Then,

$$\langle\{f_e : e \in \text{cut}(\wp)\}\rangle = \langle \cup_{T\in\wp} V_T\rangle$$

for every set $\wp$ of non-source nodes, where $V_T = \langle f_e : e \in \text{In}(T)\rangle$.

Proof. First, note that

$$\langle \cup_{T\in\wp} V_T\rangle = \langle\{f_e : e \text{ terminates at a node in } \wp\}\rangle.$$

We need to show the emptiness of the set

$$\Psi = \{c : f_c \notin \langle\{f_e : e \in \text{cut}(\wp)\}\rangle \text{ and } c \text{ terminates at a node in } \wp\}.$$

Assuming the contrary that Ψ is nonempty, we shall derive a contradiction. Choose c to be a channel in Ψ that it is not at the downstream of any other channel in Ψ. Let $c \in \text{Out}(U)$. From the definition of a linear network code, f_c is a linear combination of vectors $f_d, d \in \text{In}(U)$. As $f_c \notin \langle\{f_e : e \in \text{cut}(\wp)\}\rangle$, there exists a channel $d \in \text{In}(U)$ with $f_d \notin \langle\{f_e : e \in \text{cut}(\wp)\}\rangle$. As d is at the upstream of c, it cannot belong to the set Ψ. Thus d terminates at a node outside $\wp$. The terminal end U of d is the originating end of c. This makes c a channel in $\text{cut}(\wp)$, a contradiction to that $f_c \notin \langle\{f_e : e \in \text{cut}(\wp)\}\rangle$. □

Lemma 2.28. Let $\wp$ be a collection of non-source nodes on an acyclic network with ω imaginary channels. Then

$$\min\{\omega, \text{maxflow}(\wp)\} = \min_{\mathfrak{I}\supset\wp} |\text{cut}(\mathfrak{I})|.$$

Proof. The proof is by the standard version of the Max-flow Min-cut Theorem in the theory of network flow (see, e.g., [195]), which applies to a network with a source and a sink. Collapse the whole collection $\wp$ into a sink, and install an imaginary source at the upstream of S. Then the max-flow between this pair of source and sink is precisely $\min\{\omega, \text{maxflow}(\wp)\}$ and the min-cut between this pair is precisely $\min_{\mathfrak{I} \supset \wp} |\text{cut}(\mathfrak{I})|$. □

The above lemma equates $\min\{\omega, \text{maxflow}(\wp)\}$ with $\min_{\mathfrak{I} \supset \wp} |\text{cut}(\mathfrak{I})|$ by identifying them as the *max-flow* and *min-cut*, respectively, in a network flow problem. The requirement (2.5) of a linear dispersion is to achieve the natural bound $\min\{\omega, \text{maxflow}(\wp)\}$ on the information transmission rate from S to every group $\wp$ of non-source nodes. The following theorem verifies this qualification for a generic linear network code.

Theorem 2.29. Every generic linear network code is a linear dispersion.

Proof. Let f_e denote the global encoding kernel for each channel e in an ω-dimensional generic linear network code on an acyclic network. In view of Lemma 2.27, we adopt the abbreviation

$$\text{span}(\wp) = \langle f_e : e \in \text{cut}(\wp) \rangle = \langle \cup_{T \in \wp} V_T \rangle$$

for every set $\wp$ of non-source nodes. Thus, for any set $\mathfrak{I} \supset \wp$ ($\mathfrak{I}$ may possibly contain S), we find

$$\text{span}(\mathfrak{I}) \supset \text{span}(\wp),$$

and therefore

$$\dim(\text{span}(\wp)) \leq \dim(\text{span}(\mathfrak{I})) \leq |\text{cut}(\mathfrak{I})|.$$

In conclusion,

$$\dim(\text{span}(\wp)) \leq \min_{\mathfrak{I} \supset \wp} |\text{cut}(\mathfrak{I})|.$$

Hence, according to Lemma 2.28,

$$\dim(\text{span}(\wp)) \leq \min_{\mathfrak{I} \supset \wp} |\text{cut}(\mathfrak{I})| = \min\{\omega, \text{maxflow}(\wp)\} \leq \omega. \quad (2.10)$$

In order for the given generic linear network code to be a linear dispersion, we need

$$\dim(\mathrm{span}(\wp)) = \min\{\omega, \mathrm{maxflow}(\wp)\} \tag{2.11}$$

for every set $\wp$ of non-source nodes. From (2.10), this is true if either

$$\dim(\mathrm{span}(\wp)) = \omega \tag{2.12}$$

or

(2.13) There exists a set $\mathfrak{I} \supset \wp$ such that $\dim(\mathrm{span}(\wp)) = |\mathrm{cut}(\mathfrak{I})|$.

(Again, $\mathfrak{I}$ may contain S.) Thus, it remains to verify (2.13) under the assumption that

$$\dim(\mathrm{span}(\wp)) < \omega. \tag{2.14}$$

This is by induction on the number of non-source nodes outside $\wp$. First, assume that this number is 0, i.e., $\wp$ contains all non-source nodes. Then, because the linear network code is generic, from the remark following Definition 2.13, we see that $\dim(\mathrm{span}(\wp))$ is equal to either $|\mathrm{cut}(\mathfrak{I})|$ or $|\mathrm{cut}(\wp \cup \{S\})|$ depending on whether $|\mathrm{cut}(\wp)| \leq \omega$ or not. This establishes (2.13) by taking $\mathfrak{I}$ to be $\wp$ or $\wp \cup \{S\}$.

Next, suppose the number of non-source nodes outside $\wp$ is nonzero. Consider any such node T and write

$$\wp' = \wp \cup \{T\}.$$

Then there exists a set $\mathfrak{I}' \supset \wp'$ such that

$$\dim(\mathrm{span}(\wp')) = |\mathrm{cut}(\mathfrak{I}')|,$$

which can be seen as follows. If $\dim(\mathrm{span}(\wp')) = \omega$, take $\mathfrak{I}'$ to be the set of all nodes, otherwise the existence of such a set $\mathfrak{I}'$ follows from the induction hypothesis. Now if

$$\dim(\mathrm{span}(\wp')) = \dim(\mathrm{span}(\wp)),$$

then (2.13) is verified by taking $\mathfrak{I}$ to be $\mathfrak{I}'$. So, we shall assume that

$$\dim(\mathrm{span}(\wp')) > \dim(\mathrm{span}(\wp))$$

and hence

(2.15) There exists a channel $d \in \mathrm{In}(T)$ such that $f_d \notin \mathrm{span}(\wp)$.

The assumption (2.15) applies to every non-source node T outside $\wp$. Because of (2.14), it applies as well to the case $T = S$. Thus (2.15) applies to every node T outside $\wp$. With this, we shall show that

$$\dim(\mathrm{span}(\wp)) = |\mathrm{cut}(\wp)| \tag{2.16}$$

which would imply (2.13) by taking $\mathfrak{I}$ to be $\wp$. Write

$$\mathrm{cut}(\wp) = \{e_1, e_2, \cdots, e_m\}$$

with each $e_j \in \mathrm{Out}(T_j)$. Taking $T = T_j$ in (2.15), there exists a channel $d \in \mathrm{In}(T)$ such that $f_d \notin \mathrm{span}(\wp)$. Thus

$$\langle f_d : d \in \mathrm{In}(T_j)\rangle \not\subset \mathrm{span}(\wp) = \langle f_{e_k} : 1 \le k \le m\rangle$$

for $1 \le j \le m$. Therefore,

$$\langle f_d : d \in \mathrm{In}(T_j)\rangle \not\subset \langle f_{e_k} : k \ne j\rangle$$

since $\{e_k : k \ne j\}$ is a subset of $\{e_1, e_2, \cdots, e_m\}$. According to the requirement (2.8) for a generic linear network code, the vectors $f_{e_1}, f_{e_2}, \cdots, f_{e_m}$ are linearly independent. This verifies (2.13). □

2.4 Algorithm refinement for linear multicast

When the base field is sufficiently large, Theorem 2.21 asserts the existence of a generic linear network code and the ensuing corollaries assert the existence of a linear dispersion, a linear broadcast, and a linear multicast. The root of all these existence results traces to Algorithm 2.19, which offers the threshold $\binom{N+\omega-1}{\omega-1}$ on the sufficient size of the base field, where N is the number of channels in the network. It applies to the existence of a generic linear network code as well as the existence of a linear multicast. The lower the threshold, the stronger are the existence statements.

Generally speaking, the weaker the requirement on a class of special linear network codes, the smaller is the required size of the base field. The following is an example of an acyclic network where the requirement on the base field for a generic linear network code is more stringent than it is for a linear multicast.

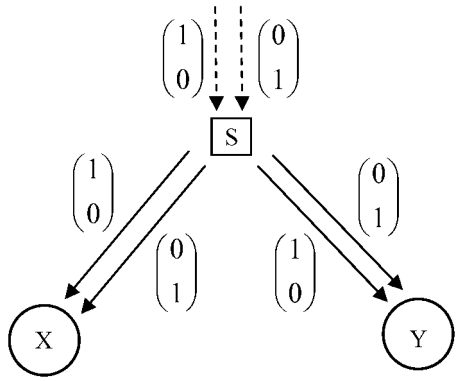

Fig. 2.7 A network on which a 2-dimensional binary linear multicast and a ternary generic linear network code exist, but not a binary generic linear network code.

Example 2.30. Figure 2.7 presents a 2-dimensional linear multicast on an acyclic network regardless of the choice of the base field. The linear multicast becomes a 2-dimensional ternary generic linear network code when the global encoding kernels for the two channels from S to Y are replaced by $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 2 \end{pmatrix}$. On the other hand, it is not hard to prove the nonexistence of a 2-dimensional binary generic linear network code on the same network.

The aforementioned threshold on the sufficient size of the base field is only a sufficient condition for existence but not a necessary one. Sometimes the existence is independent of the choice of the base field. For instance, Example 2.7 constructs a 2-dimensional linear multicast on the network in Figure 2.2 regardless of the choice of the base field. However, the choice of the base field and more generally the alphabet size plays an intriguing role. For instance, a multicast may exist on a network for a certain alphabet but not necessarily for some larger alphabets [168].

With respect to Algorithm 2.19, it is plausible that one can devise a computationally more efficient algorithm for constructing a code that is weaker than a generic linear network code. The following algorithm exemplifies a fine tuning of Algorithm 2.19 with an aim to lower the

computational complexity as well as the threshold on the sufficient size of the base field. This algorithm as presented is only for the construction of a linear multicast, but it can be adapted for the construction of a linear broadcast in a straightforward manner.

Algorithm 2.31. (Construction of a linear multicast) [183] The objective is to modify Algorithm 2.19 for efficient construction of a linear multicast. This algorithm constructs an ω-dimensional F-valued linear multicast on an acyclic network when $|F| > \eta$, the number of non-source nodes T in the network with $\text{maxflow}(T) \geq \omega$. Denote these η non-source nodes by $T_1, T_2, \ldots, T_\eta$.

A sequence of channels $e_1, e_2, \ldots, e_l$ is called a *path* leading to a node T_q when $e_1 \in \text{In}(S)$, $e_l \in \text{In}(T_q)$, and (e_j, e_{j+1}) is an adjacent pair for all j. For each q, $1 \leq q \leq \eta$, there exist channel-disjoint paths $P_{q,1}, P_{q,2}, \ldots, P_{q,\omega}$ leading to T_q. Altogether there are $\eta\omega$ paths. Adopt the notation $V_T = \langle \{f_d : d \in \text{In}(T)\} \rangle$ as before. The following procedure prescribes a global encoding kernel f_e for every channel e in the network such that $\dim(V_{T_q}) = \omega$ for $1 \leq q \leq \eta$.

```
{
            // By definition, the global encoding kernels for the ω
            // imaginary channels form the standard basis of F^ω.
    for (every channel e in the network)
        f_e = the zero vector;
            // This is just initialization. f_e will be updated in an
            // upstream-to-downstream order.
    for (q = 1; q ≤ η; q++)
        for (i = 1; i ≤ ω; i++)
            e_{q,i} = the imaginary channel initiating the path P_{q,i};
            // This is just initialization. Later e_{q,i} will be
            // dynamically updated by moving down along the path
            // P_{q,i} until finally e_{q,i} becomes a channel in In(T_q).
    for (every node T, in any upstream-to-downstream order)
    {
        for (every channel e ∈ Out(T))
        {
            // With respect to this channel e, define a "pair" as a
```

```
            // pair (q,i) of indices such that the channel e is on the
            // path P_{q,i}. Note that for each q, there exists at most
            // one pair (q,i). Thus, the number of pairs is at least 0
            // and at most η. Since the nodes T are chosen in
            // an upstream-to-downstream manner, if (q,i) is a pair,
            // then e_{q,i} ∈ In(T) by induction, so that f_{e_{q,i}} ∈ V_T. For
            // reasons to be explained in the justification below,
            // f_{e_{q,i}} ∉ ⟨{f_{e_{q,j}} : j ≠ i}⟩, and therefore
            // f_{e_{q,i}} ∈ V_T\⟨{f_{e_{q,j}} : j ≠ i}⟩.
        Choose a vector w in V_T such that w ∉ ⟨{f_{e_{q,j}} : j ≠ i}⟩ for
          every pair (q,i);
            // To see the existence of such a vector w, denote
            // dim(V_T) = k. Then, dim(V_T ∩ ⟨{f_{e_{q,j}} : j ≠ i}⟩) ≤
            // k − 1 for every pair (q,i) since
            // f_{e_{q,i}} ∈ V_T\⟨{f_{e_{q,j}} : j ≠ i}⟩. Thus
            // |V_T ∩ (∪_{(q,i): a pair}⟨{f_{e_{q,j}} : j ≠ i}⟩)|
            // ≤ η|F|^{k−1} < |F|^k = |V_T|.
        f_e = w;
            // This is equivalent to choosing scalar values for local
            // encoding kernels k_{d,e} for all d ∈ In(T) such that
            // Σ_{d∈In(T)} k_{d,e} f_d ∉ ⟨{f_{e_{q,j}} : j ≠ i}⟩ for every pair (q,i).
        for (every pair (q,i))
            e_{q,i} = e;
    }
  }
}
```

Justification. For $1 \le q \le \eta$ and $1 \le i \le \omega$, the channel $e_{q,i}$ is on the path $P_{q,i}$. Initially $e_{q,i}$ is an imaginary channel at S. Through dynamic updating it moves downstream along the path until finally reaching a channel in $\mathrm{In}(T_q)$.

Fix an index q, where $1 \le q \le \eta$. Initially, the vectors $f_{e_{q,1}}, f_{e_{q,2}}, \ldots,$ $f_{e_{q,\omega}}$ are linearly independent because they form the standard basis of F^ω. At the end, they need to span the vector space V_{T_q}. Therefore, in order for the eventually constructed linear network code to qualify

as a linear multicast, it suffices to show the preservation of the linear independence among $f_{e_{q,1}}, f_{e_{q,2}}, \ldots, f_{e_{q,\omega}}$ throughout the algorithm.

Fix a node X_j and a channel $e \in \mathrm{Out}(X_j)$. We need to show the preservation in the generic step of the algorithm for each channel e in the "for loop." The algorithm defines a "pair" as a pair (q,i) of indices such that the channel e is on the path $P_{q,i}$. When no (q,i) is a pair for $1 \leq i \leq \omega$, the channels $e_{q,1}, e_{q,2}, \ldots, e_{q,\omega}$ are not changed in the generic step; neither are the vectors $f_{e_{q,1}}, f_{e_{q,2}}, \ldots, f_{e_{q,\omega}}$. So we may assume the existence of a pair (q,i) for some i. The only change among the channels $e_{q,1}, e_{q,2}, \ldots, e_{q,\omega}$ is that $e_{q,i}$ becomes e. Meanwhile, the only change among the vectors $f_{e_{q,1}}, f_{e_{q,2}}, \ldots, f_{e_{q,\omega}}$ is that $f_{e_{q,i}}$ becomes a vector $w \notin \langle \{f_{e_{q,j}} : j \neq i\} \rangle$. This preserves the linear independence among $f_{e_{q,1}}, f_{e_{q,2}}, \ldots, f_{e_{q,\omega}}$ as desired.

Analysis of complexity. Let N be the number of channels in the network as in Algorithm 2.19. In Algorithm 2.31, the generic step for each channel e in the "for loop" processes at most η pairs, where the processing of a pair is analogous to the processing of a collection ξ of channels in Algorithm 2.19. Throughout Algorithm 2.31, at most $N\eta$ such collections of channels are processed. From this, it is not hard to implement Algorithm 2.31 within a polynomial time in N for a fixed ω. The computational details can be found in [183]. It is straightforward to extend Algorithm 2.31 for the construction of a linear broadcast in similar polynomial time.

2.5 Static network codes

So far, a linear network code has been defined on a network with a fixed network topology. In some applications, the configuration of a communication network may vary from time to time due to traffic congestion, link failure, etc. The problem of a linear multicast under such circumstances was first considered in [184].

Convention. A *configuration* ε of a network is a mapping from the set of channels in the network to the set $\{0,1\}$. Channels in $\varepsilon^{-1}(0)$ are *idle* channels with respect to this configuration, and the subnetwork resulting from the deletion of idle channels will be called the

ε-subnetwork. The maximum flow from S to a non-source node T over the ε-subnetwork is denoted as $\text{maxflow}_\varepsilon(T)$. Similarly, the maximum flow from S to a collection $\wp$ of non-source nodes over the ε-subnetwork is denoted as $\text{maxflow}_\varepsilon(\wp)$.

Definition 2.32. Let F be a finite field and ω a positive integer. Let $k_{d,e}$ be the local encoding kernel for each adjacent pair (d, e) in an ω-dimensional F-valued linear network code on an acyclic communication network. The *ε-global encoding kernel* for the channel e, denoted by $f_{e,\varepsilon}$, is the ω-dimensional column vector calculated recursively in an upstream-to-downstream order by:

(2.17) $f_{e,\varepsilon} = \varepsilon(e)\Sigma_{d\in \text{In}(T)} k_{d,e} f_{d,\varepsilon}$ for $e \in \text{Out}(T)$.
(2.18) The ε-global encoding kernels for the ω imaginary channels are independent of ε and form the natural basis of the space F^ω.

Note that in the above definition, the local encoding kernels $k_{d,e}$ are not changed with ε. Given the local encoding kernels, the ε-global encoding kernels can be calculated recursively by (2.17), while (2.18) serves as the boundary conditions. Let the source generate a message x in the form of an ω-dimensional row vector when the prevailing configuration is ε. A node T receives the symbols $x \cdot f_{d,\varepsilon}$, $d \in \text{In}(T)$, from which it calculates the symbol $x \cdot f_{e,\varepsilon}$ to be sent on each channel $e \in \text{Out}(T)$ via the linear formula

$$x \cdot f_{e,\varepsilon} = \varepsilon(e)\Sigma_{d\in \text{In}(T)} k_{d,e}(x \cdot f_{d,\varepsilon}).$$

In particular, a channel e with $\varepsilon(e) = 0$ has $f_{e,\varepsilon} = 0$ according to (2.17) and transmits the symbol $x \cdot f_{e,\varepsilon} = 0$. In a real network, a failed channel does not transmit the symbol 0. Rather, whenever a symbol is not received on an input channel, the symbol is regarded as being 0.

Definition 2.33. Following the notation of Definition 2.32 and adopting the abbreviation $V_{T,\varepsilon} = \langle \{f_{d,\varepsilon} : d \in \text{In}(T)\}\rangle$, the ω-dimensional F-valued linear network code qualifies as an *static linear multicast*, *static linear broadcast*, *static linear dispersion*, and *static generic linear*

network code, respectively, if the following statements hold:

(2.19) $\dim(V_{T,\varepsilon}) = \omega$ for every configuration ε and every non-source node T with $\text{maxflow}_\varepsilon(T) \geq \omega$.

(2.20) $\dim(V_{T,\varepsilon}) = \min\{\omega, \text{maxflow}_\varepsilon(T)\}$ for every configuration ε and every non-source node T.

(2.21) $\dim(\langle \cup_{T\in\wp} V_{T,\varepsilon}\rangle) = \min\{\omega, \text{maxflow}_\varepsilon(\wp)\}$ for every configuration ε and every collection $\wp$ of non-source nodes.

(2.22) Let ε be a configuration and $\{e_1, e_2, \ldots, e_m\}$ a set of channels, where each $e_j \in \text{Out}(T_j) \cap \varepsilon^{-1}(1)$. Then, the vectors $f_{e_1,\varepsilon}, f_{e_2,\varepsilon}, \ldots, f_{e_m,\varepsilon}$ are linearly independent (and hence $m \leq \omega$) provided that $V_{T_j,\varepsilon} \not\subset \langle\{f_{e_k,\varepsilon} : k \neq j\}\rangle$ for all j.

The adjective "static" in the terms above stresses the fact that, while the configuration ε varies, the local encoding kernels remain unchanged. The advantage of using a static linear dispersion, broadcast, or multicast in case of link failure is that the local operation at any node in the network is affected only at the minimum level. Each receiving node in the network, however, needs to know the configuration ε before decoding can be done correctly. In real implementation, this information can be provided by a separate signaling network. In the absence of such a network, training methods for conveying this information to the receiving nodes have been proposed in [170].

Example 2.34. A 2-dimensional $GF(5)$-valued linear network code on the network in Figure 2.8 is prescribed by the following local encoding kernels at the nodes:

$$K_S = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \text{ and } K_X = \begin{bmatrix} 1 & 3 \\ 3 & 2 \\ 1 & 1 \end{bmatrix}$$

Claim that this is a static generic linear network code. Denote the three channels in $\text{In}(X)$ by c, d and e and the two in $\text{Out}(X)$ by g and h. The vectors $f_{g,\varepsilon}$ and $f_{h,\varepsilon}$ for all possible configurations ε are tabulated in Table 2.1, from which it is straightforward to verify the condition (2.22).

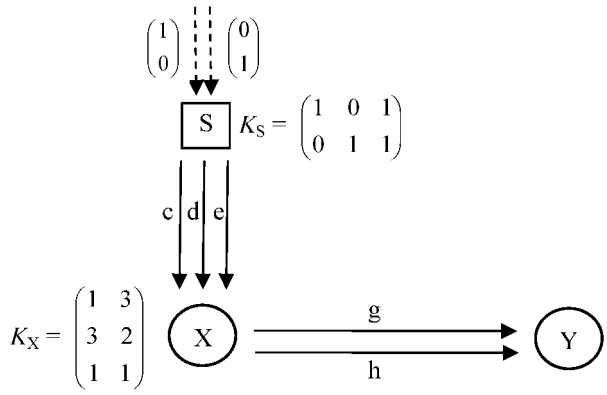

Fig. 2.8 A 2-dimensional $GF(5)$-valued static generic linear network code.

Table 2.1 The vectors $f_{g,\varepsilon}$ and $f_{h,\varepsilon}$ for all possible configurations ε in Example 2.34.

$\varepsilon(c)$	0	0	0	1	1	1	1
$\varepsilon(d)$	0	1	1	0	0	1	1
$\varepsilon(e)$	1	0	1	0	1	0	1
$f_{g,\varepsilon}$	$\varepsilon(g)\begin{bmatrix}1\\1\end{bmatrix}$	$\varepsilon(g)\begin{bmatrix}0\\3\end{bmatrix}$	$\varepsilon(g)\begin{bmatrix}1\\4\end{bmatrix}$	$\varepsilon(g)\begin{bmatrix}1\\0\end{bmatrix}$	$\varepsilon(g)\begin{bmatrix}2\\1\end{bmatrix}$	$\varepsilon(g)\begin{bmatrix}1\\3\end{bmatrix}$	$\varepsilon(g)\begin{bmatrix}2\\4\end{bmatrix}$
$f_{h,\varepsilon}$	$\varepsilon(h)\begin{bmatrix}1\\1\end{bmatrix}$	$\varepsilon(h)\begin{bmatrix}0\\2\end{bmatrix}$	$\varepsilon(h)\begin{bmatrix}1\\3\end{bmatrix}$	$\varepsilon(h)\begin{bmatrix}3\\0\end{bmatrix}$	$\varepsilon(h)\begin{bmatrix}4\\1\end{bmatrix}$	$\varepsilon(h)\begin{bmatrix}3\\2\end{bmatrix}$	$\varepsilon(h)\begin{bmatrix}4\\3\end{bmatrix}$

The following is an example of a generic linear network code that does not qualify for a static linear multicast.

Example 2.35. On the network in Figure 2.8, a 2-dimensional $GF(5)$-valued generic linear network is prescribed by the following local encoding kernels at the nodes:

$$K_S = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \quad \text{and} \quad K_X = \begin{bmatrix} 2 & 1 \\ 1 & 2 \\ 0 & 0 \end{bmatrix}$$

For the configuration ε such that

$$\varepsilon(c) = 0 \quad \text{and} \quad \varepsilon(d) = \varepsilon(e) = 1,$$

we have the ε-global encoding kernels $f_{g,\varepsilon} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ and $f_{h,\varepsilon} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$ and hence $\dim(V_{Y,\varepsilon}) = 1$. On the other hand $\text{maxflow}_\varepsilon(Y) = 2$, and hence this generic linear network code is not a static linear multicast.

Recall that in Algorithm 2.19 for the construction of a generic linear network code, the key step chooses for a channel $e \in \text{Out}(T)$ a vector in $V_T = \langle\{f_d : d \in \text{In}(T)\}\rangle$ to be the global encoding kernel f_e such that $f_e \notin \langle\{f_c : c \in \xi\}\rangle$, where ξ is any collection of $\omega - 1$ channels as prescribed with $V_T \not\subset \langle\{f_c : c \in \xi\}\rangle$. This is equivalent to choosing scalar values for local encoding kernels $k_{d,e}$ for all $d \in \text{In}(T)$ such that $\Sigma_{d\in\text{In}(T)} k_{d,e} f_d \notin \langle\{f_c : c \in \xi\}\rangle$. Algorithm 2.19 is adapted below for the construction of a static generic linear network code.

Algorithm 2.36. (Construction of a static generic linear network code) Given a positive integer ω and an acyclic network with N channels, the following procedure constructs an ω-dimensional F-valued static generic linear network code when the field F contains more than $2^N \binom{N+\omega-1}{\omega-1}$ elements. Write $V_{T,\varepsilon} = \langle\{f_{d,\varepsilon} : d \in \text{In}(T)\}\rangle$. The key step in the construction will be to choose scalar values for the local encoding kernels $k_{d,e}$ such that $\Sigma_{d\in\text{In}(T)} k_{d,e} f_{d,\varepsilon} \notin \langle\{f_{c,\varepsilon} : c \in \xi\}\rangle$ for every configuration ε and every collection ξ of $\omega - 1$ channels, including possibly the imaginary channels in $\text{In}(S)$, with $V_{T,\varepsilon} \not\subset \langle\{f_{c,\varepsilon} : c \in \xi\}\rangle$. Then, $f_{e,\varepsilon}$ will be set as $f_{e,\varepsilon} = \varepsilon(e)\Sigma_{d\in\text{In}(T)} k_{d,e} f_{d,\varepsilon}$.

```
{
        // By definition, the global encoding kernels for the ω
        // imaginary channels form the standard basis of F^ω.
    for (every channel e)
        for (every configuration ε)
            f_{e,ε} = the zero vector;
                // Initialization.
    for (every node T, following an upstream-to-downstream order)
    {
        for (every channel e ∈ Out(T))
        {
            Choose scalar values for k_{d,e} for all d ∈ T such that
```

$\Sigma_{d\in\mathrm{In}(T)}k_{d,e}f_d \notin \langle\{f_{c,\varepsilon} : c \in \xi\}\rangle$ for every configuration ε and every collection ξ of channels with $V_{T,\varepsilon} \not\subset \langle\{f_{c,\varepsilon} : c \in \xi\}\rangle$;

// To see the existence of such values $k_{d,e}$*, let* $\dim(V_{T,\varepsilon}) = m$*. For any collection* ξ *of channels with* $V_{T,\varepsilon} \not\subset \langle\{f_{c,\varepsilon} : c \in \xi\}\rangle$*, the space* $V_{T,\varepsilon} \cap \langle\{f_{c,\varepsilon} : c \in \xi\}\rangle$ *is less than* m*-dimensional. Consider the linear mapping from* $F^{|\mathrm{In}(T)|}$ *onto* F^{ω} *via* $[k_{d,e}]_{d\in\mathrm{In}(T)} \mapsto \Sigma_{d\in\mathrm{In}(T)}k_{d,e}f_{d,\varepsilon}$*. The nullity of this linear mapping is* $|\mathrm{In}(T)| - m$*. Hence the pre-image of the space* $V_{T,\varepsilon} \cap \langle\{f_{c,\varepsilon} : c \in \xi\}\rangle$ *is less than* $|\mathrm{In}(T)|$*-dimensional. Thus the pre-image of* $\cup_{\varepsilon,\xi}(V_{T,\varepsilon} \cap \langle\{f_{c,\varepsilon} : c \in \xi\}\rangle)$ *contains at most* $2^N\binom{N+\omega-1}{\omega-1}|F|^{|\mathrm{In}(T)|-1}$ *elements, which are fewer than* $|F|^{|\mathrm{In}(T)|}$ *if* $|F| > 2^N\binom{N+\omega-1}{\omega-1}$.

for (every configuration ε)

$f_{e,\varepsilon} = \varepsilon(e)\Sigma_{d\in\mathrm{In}(T)}k_{d,e}f_{d,\varepsilon}$;

}

}

}

Justification. The explanation for the code constructed by Algorithm 2.36 being a static generic network code is exactly the same as that given in the justification of Algorithm 2.19. The details are omitted.

Algorithm 2.36 constitutes a constructive proof for the following theorem.

Theorem 2.37. Given a positive integer ω and an acyclic network, there exists an ω-dimensional F-valued static generic linear network code when the field F is sufficiently large.

Corollary 2.38. Given a positive integer ω and an acyclic network, there exists an ω-dimensional F-valued static linear dispersion when the field F is sufficiently large.

Corollary 2.39. Given a positive integer ω and an acyclic network, there exists an ω-dimensional F-valued static linear broadcast when the field F is sufficiently large.

Corollary 2.40. Given a positive integer ω and an acyclic network, there exists an ω-dimensional F-valued static linear multicast when the field F is sufficiently large.

The original proof of Corollary 2.40, given in [184], was by extending the alternative proof of Corollary 2.24 in the preceding section. This, together with Lemma 2.17, provides another construction algorithm for a static linear multicast when the base field is sufficiently large. In fact, this algorithm can be extended to the construction of a static linear broadcast.

The requirements (2.19) through (2.21) in Definition 2.32 refer to all the 2^n possible configurations. Conceivably, a practical application may deal with only a certain collection $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_\kappa\}$ of configurations in order to provide link contingency, network security, network expandability, transmission redundancy, alternate routing upon congestion, etc. We may define, for instance, an $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_\kappa\}$-*static linear multicast* and an $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_\kappa\}$-*static linear broadcast* by replacing the conditions (2.19) and (2.20) respectively by

(2.23) $\dim(V_{T,\varepsilon}) = \omega$ for every configuration $\varepsilon \in \{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_\kappa\}$ and every non-source node T with $\text{maxflow}_\varepsilon(T) \geq \omega$.

(2.24) $\dim(V_{T,\varepsilon}) = \min\{\omega, \text{maxflow}_\varepsilon(T)\}$ for every configuration $\varepsilon \in \{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_\kappa\}$ and every non-source node T.

Recall that Algorithm 2.19 is converted into Algorithm 2.36 by modifying the key step in the former. In a similar fashion, Algorithm 2.31 can be adapted for the construction of an $\{\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_\kappa\}$-static linear multicast or linear broadcast. This will lower the threshold on the sufficient size of the base field as well as the computational complexity. In fact, the computation can be in polynomial time with respect to κN, where N is the number of channels in the network.

3

Cyclic Networks

A communication network is said to be *cyclic* when it contains at least one directed cycle. The present section, mainly based on [186], deals with network coding for a whole pipeline of messages over a cyclic network.

One problem with applying the local description of a linear network code (Definition 2.4) and the global description (Definition 2.5) to a cyclic network is in their different treatments of each individual message in the pipeline generated by the source node. When the communication network is acyclic, operation at all nodes can be synchronized so that each message is individually encoded and propagated from the upstream nodes to the downstream nodes. That is, the processing of each message is independent of the sequential messages in the pipeline. In this way, the network coding problem is independent of the propagation delay, which may include transmission delay over the channels as well as processing delay at the nodes. Over a cyclic network, however, the global encoding kernels for all channels could be simultaneously implemented only under the ideal assumption of delay-free communications, which is of course unrealistic. The propagation and encoding of sequential messages can potentially convolve

together. Thus the amount of delay incurred in transmission and processing becomes part of the consideration in network coding. That is, the time dimension is an essential part of the transmission medium over a cyclic network. Another problem is the non-equivalence between Definition 2.4 and Definition 2.5 over a cyclic network, as we shall see in the next section.

3.1 Non-equivalence between local and global descriptions of a linear network code over a delay-free cyclic network

Definition 2.4 for the local description and Definition 2.5 for the global description of a linear network code are equivalent over an acyclic network, because given the local encoding kernels, the global encoding kernels can be calculated recursively in any upstream-to-downstream order. In other words, the equation (2.3) has a unique solution for the global encoding kernels in terms of the local encoding kernels, while (2.4) serves as the boundary conditions. If these descriptions are applied to a cyclic network, certain logical problems are expected to arise.

First, let f_d denote the global encoding kernel for a channel d. Then for every collection $\wp$ of non-source nodes in the network, it is only natural that

$$\langle \{ f_d : d \in \mathrm{In}(T) \text{ for some } T \in \wp \} \rangle = \langle \{ f_e : e \in \mathrm{cut}(\wp) \} \rangle.$$

However, Definition 2.5 does not always imply this equality over a cyclic network. Second, given the local encoding kernels, there may exist none or one or more solutions for the global encoding kernels. Below we give one example with a unique solution, one with no solution, and one with multiple solutions.

Example 3.1. Recall the network in Figure 1.2(b) which depicts the conversation between two sources over a communication network. An equivalent representation of this network obtained by creating a single source node that generates both b_1 and b_2 and appending two imaginary incoming channels to the source node is shown in Figure 3.1. Let ST precede VT in the ordering among the channels. Similarly, let ST'

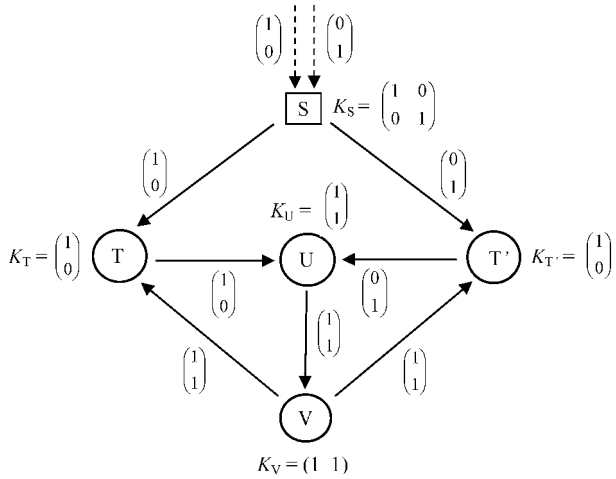

Fig. 3.1 A 2-dimensional linear broadcast on a cyclic network.

precede VT'. Given the local encoding kernels

$$K_S = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, K_T = K_{T'} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, K_U = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, K_V = \begin{bmatrix} 1 & 1 \end{bmatrix},$$

the equation (2.3) yields the following unique solution for the global encoding kernels:

$$f_{ST} = f_{TU} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \ f_{ST'} = f_{T'U} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$f_{UV} = f_{VT} = f_{VT'} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

These encoding kernels are shown in Figure 3.1 and in fact, define a 2-dimensional linear broadcast regardless of the choice of the base field.

Example 3.2. A randomly prescribed set of local encoding kernels at the nodes on a cyclic network is unlikely to be compatible with any global encoding kernels. In Figure 3.2(a), a local encoding kernel K_T is prescribed at each node T in a cyclic network. Had there existed a global encoding kernel f_e for each channel e, the requirement (2.3)

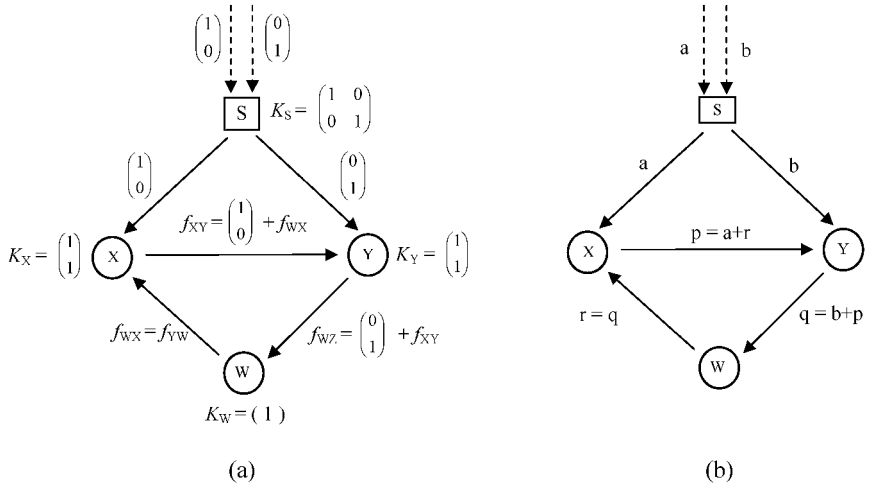

Fig. 3.2 (a) The requirement on the global encoding kernels that are compatible with the prescribed local encoding kernels and (b) the scalar value $x \cdot f_e$ that would be carried by each channel e had the global encoding kernels existed.

would imply the equations

$$f_{XY} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} + f_{WX},\ f_{YW} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} + f_{XY},\ f_{WX} = f_{YW},$$

which sum up to a contradiction.

The nonexistence of compatible global encoding kernels can also be interpreted in terms of message transmission. Let S generate the message $x = (a, b) \in F^2$. The intended symbol for the transmission over each channel e is $x \cdot f_e$ as shown in Figure 3.2(b). In particular, the symbols $p = x \cdot f_{XY}$, $q = x \cdot f_{YW}$, and $r = x \cdot f_{WX}$ are correlated by

$$\begin{aligned} p &= a + r \\ q &= b + p, \\ r &= q. \end{aligned}$$

These equalities imply that $a + b = 0$, a contradiction to the independence between the two components a and b of a generic message.

Example 3.3. Let F be a field extension of GF(2). Consider the same prescription of local encoding kernels at nodes as in Example 3.2 except that $K_S = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}$. The following three sets of global encoding kernels meet the requirement (2.3) in the definition of a linear network code:

$$f_{SX} = f_{SY} = \begin{bmatrix} 1 \\ 0 \end{bmatrix},\ f_{XY} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},\ f_{YW} = f_{WX} = \begin{bmatrix} 1 \\ 0 \end{bmatrix};$$

$$f_{SX} = f_{SY} = \begin{bmatrix} 1 \\ 0 \end{bmatrix},\ f_{XY} = \begin{bmatrix} 1 \\ 0 \end{bmatrix},\ f_{YW} = f_{WX} = \begin{bmatrix} 0 \\ 0 \end{bmatrix};$$

$$f_{SX} = f_{SY} = \begin{bmatrix} 1 \\ 0 \end{bmatrix},\ f_{XY} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},\ f_{YW} = f_{WX} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

3.2 Convolutional network code

Let every channel in a network carry a scalar value in every time slot. For both physical feasibility and mathematical logic, we need a certain assumption on the transmission/processing delay to ensure a nonzero delay when a message is propagated around any cycle in the network. Both [188] and [184] simply assume a negligible transmission delay and a unit-time delay in the node processing, and a communication network under this assumption can be called a *unit-delay network*. In this expository text, we shall again consider only unit-delay networks in order to simplify the notation in mathematical formulation and proofs. The results to be developed in this section, although discussed in the context of cyclic networks, apply equally well to acyclic networks.

As a time-multiplexed network in the combined time-space domain, a unit-delay network can be unfolded with respect to the time dimension into an indefinitely long network called a *trellis network*. Corresponding to a physical node X, there is a sequence of nodes $X_0, X_1, X_2, \ldots$ in the trellis network. A channel in the trellis network represents a physical channel e only for a particular time slot $t \geq 0$, and is thereby identified by the pair (e,t). When e is from the node X to the node Y, the channel (e,t) is then from the node X_t to the node Y_{t+1}. The trellis network is acyclic regardless of the topology of the

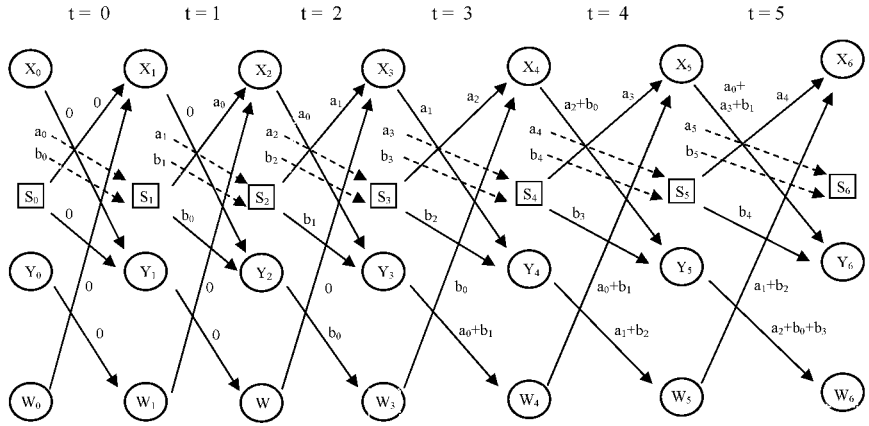

Fig. 3.3 Message transmission via a convolutional network code on a cyclic network means the pipelining of sequential symbols through every channel. The transmission media in the time-space domain can be unfolded with respect to the time dimension into an indefinitely long "trellis network."

physical network, and the upstream-to-downstream order in the trellis network is along the forward direction of time.

Example 3.4. Based on the local encoding kernels on the network in Figure 3.2, every channel (e,t), $t = 0,1,2,\ldots$ in the corresponding trellis network in Figure 3.3 carries a scalar value. For instance, the channels (XY,t), $t \geq 0$ carry the successive scalar values $0,0,a_0,a_1,a_2+b_0,a_0+a_3+b_1,a_1+a_4+b_2,a_2+a_5+b_0+b_3,\ldots$ Such a code is called a convolutional code (over the network) to be formally defined in Definition 3.5.

Given a field F, functions of the form $p(z)/(1+zq(z))$, where $p(z)$ and $q(z)$ are polynomials, are expandable into power series at $z = 0$. Rational functions of this form may be called "rational power series." They constitute an *integral domain*[1], which will be denoted by $F\langle z\rangle$. The integral domain of all power series over F is conventionally denoted by $F[[z]]$. Thus $F\langle z\rangle$ is a subdomain of $F[[z]]$.

[1] An integral domain is a commutative ring with unity $1 \neq 0$ and containing no divisors of 0. See for example [175].

Let the channel e carry the scalar value $c_t \in F$ for each $t \geq 0$. A succinct mathematical expression for a scalar sequence $(c_0, c_1, \ldots, c_t, \ldots)$ is the z-transform $\sum_{t\geq 0} c_t z^t \in F[[z]]$, where the power t of the dummy variable z represents discrete time. The pipelining of scalars over a time-multiplexed channel can thus be regarded as the transmission of a power series over the channel. For example, the transmission of a scalar value on the channel (XY, t) for each $t \geq 0$ in the trellis network of Figure 3.3 translates into the transmission of the power series

$$a_0 z^2 + a_1 z^3 + (a_2 + b_0) z^4 + (a_0 + a_3 + b_1) z^5 + (a_1 + a_4 + b_2) z^6 + (a_2 + a_5 + b_0 + b_3) z^7 + \cdots \tag{3.1}$$

over the channel XY in the network in Figure 3.2.

Definition 3.5. Let F be a finite field and ω a positive integer. An ω-dimensional F-valued *convolutional network code* on a unit-delay network consists of an element $k_{d,e}(z) \in F\langle z \rangle$ for every adjacent pair (d, e) in the network as well as an ω-dimensional column vector $f_e(\mathrm{z})$ over $F\langle z \rangle$ for every channel e such that:

(3.1) $f_e(z) = z \sum_{d \in In(T)} k_{d,e}(z) f_d(z)$ for $e \in \mathrm{Out}(T)$.

(3.2) The vectors $f_e(z)$ for the imaginary channels e, i.e., those ω channels in $\mathrm{In}(S)$, consist of scalar components that form the natural basis of the vector space F^ω.

The vector $f_e(z)$ is called the *global encoding kernel* for the channel e and $k_e(z)$ is called the *local encoding kernel* for the adjacent pair (d, e). The local encoding kernel at the node T refers to the $|\mathrm{In}(T)| \times |\mathrm{Out}(T)|$ matrix $K_T(z) = [k_{d,e}(z)]_{d \in In(T), e \in Out(T)}$.

This notion of a *convolutional network code* is a refinement of a "time-invariant linear-code multicast (TILCM)" in [LYC03]. The equation in (3.1) is the time-multiplexed version of (2.3), and the factor z in it indicates a unit-time delay in node processing. In other words, the filters in data processing for the calculation of $f_e(z)$ are $z k_{d,e}(z)$ for all channels $d \in \mathrm{In(T)}$. Write

$$f_e(z) = \sum_{t \geq 0} f_{e,t} z^t$$

and

$$k_{d,e}(z) = \sum_{t \geq 0} k_{d,e,t} z^t,$$

where each $f_{e,t}$ and $k_{d,e,t}$ are ω-dimensional column vectors in F^ω. The convolutional equation (3.1) can be further rewritten as

$$f_{e,t} = \sum_{d \in \mathrm{In}(T)} \left(\sum_{0 \leq u < t} k_{d,e,u} f_{d,t-1-u} \right) \quad \text{for all } t \geq 0, \tag{3.3}$$

with the boundary conditions provided by (3.2):

- The vectors $f_{e,0}$ for the imaginary channels e form the natural basis of the vector space F^ω over F.
- $f_{e,t}$ is the zero vector for all t > 0 when e is one of the imaginary channels.

Note that for $t = 0$, the summation in (3.3) is empty, and $f_{e,0}$ is taken to be zero by convention. With these boundary conditions, the global encoding kernels can be recursively calculated from the local encoding kernels through (3.3), where the recursive procedure follows the forward direction of time. This is equivalent to a linear network code on the indefinitely long trellis network, which is an acyclic network.

Example 3.6. In Figure 3.2, let the $\omega = 2$ imaginary channels be denoted as OS and OS'. Let SX precede WX in the ordering among the channels, and similarly let SY precede XY. A convolutional network code is specified by the prescription of a local encoding kernel at every node:

$$K_S(z) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \; K_X(z) = K_Y(z) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \; K_W(z) = \begin{bmatrix} 1 \end{bmatrix},$$

and a global encoding kernel for every channel:

$$f_{OS}(z) = \begin{bmatrix} 1 \\ 0 \end{bmatrix},\ f_{OS'}(z) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$f_{SX}(z) = z \begin{bmatrix} 1\ 0 \\ 0\ 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} z \\ 0 \end{bmatrix}$$

$$f_{SY}(z) = z \begin{bmatrix} 1\ 0 \\ 0\ 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ z \end{bmatrix}$$

$$f_{XY}(z) = \begin{bmatrix} z^2/(1-z^3) \\ z^4/(1-z^3) \end{bmatrix},\ f_{YW}(z) = \begin{bmatrix} z^3/(1-z^3) \\ z^2/(1-z^3) \end{bmatrix}$$

$$f_{WX}(z) = \begin{bmatrix} z^4/(1-z^3) \\ z^3/(1-z^3) \end{bmatrix},$$

where the last three global encoding kernels have been solved from the following equations:

$$f_{XY}(z) = z \begin{bmatrix} f_{SX}(z)\ f_{WX}(z) \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix} = z^2 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + z f_{WX}(z)$$

$$f_{YW}(z) = z \begin{bmatrix} f_{SY}(z)\ f_{XY}(z) \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix} = z^2 \begin{bmatrix} 0 \\ 1 \end{bmatrix} + z f_{XY}(z)$$

$$f_{WX}(z) = z(f_{YW}(z)) \cdot \begin{bmatrix} 1 \end{bmatrix} = z f_{YW}(z).$$

These local and global encoding kernels of a 2-dimensional convolutional network code are summarized in Figure 3.4. They correspond to the encoding kernels of a 2-dimensional linear network code over the trellis network.

Represent the message generated at the source node S at the time slot t, where $t \geq 0$, by the ω-dimensional row vector $x_t \in F^\omega$. Equivalently, S generates the message pipeline represented by the z-transform

$$x(z) = \sum_{t \geq 0} x_t z^t,$$

which is an ω-dimensional row vector over $F[[z]]$. In real applications, $x(z)$ is always a polynomial because of the finite length of the message pipeline. Through a convolutional network code, each channel e carries

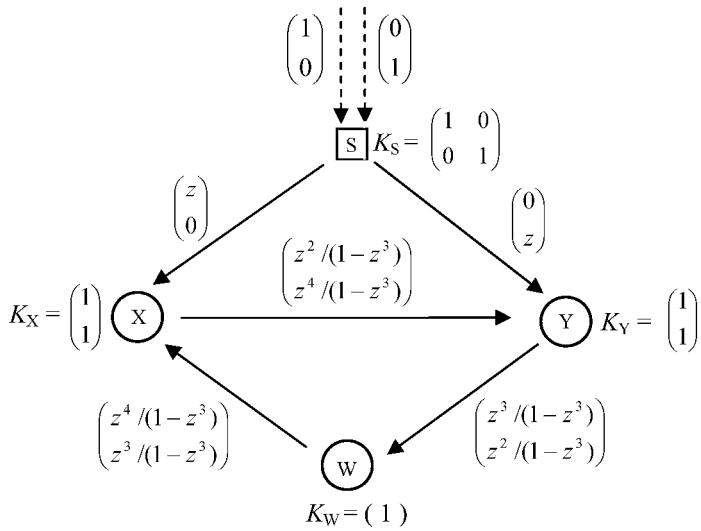

Fig. 3.4 A convolutional network code on a cyclic network that can be unfolded with respect to the time dimension into the linear network code on the trellis.

the power series $x(z) \cdot f_e(z)$. Write

$$m_{e,t} = \sum_{0 \le u \le t} x_u f_{e,t-u},$$

so that

$$x(z) \cdot f_e(z) = \sum_{t \ge 0} m_{e,t} z^t.$$

For $e \in \mathrm{Out}(T)$, the equation (3.1) yields

$$x(z) \cdot f_e(z) = z \sum_{d \in \mathrm{In}(T)} k_{d,e}(z)(x(z) \cdot f_d(z)), \tag{3.4}$$

or equivalently, in time domain,

$$m_{e,t} = \sum_{d \in \mathrm{In}(T)} \left(\sum_{0 \le u < t} k_{d,e,u} m_{d,t-1-u} \right). \tag{3.5}$$

A node T calculates the scalar value $m_{e,t}$ for sending onto each outgoing channel e at time t from the accumulative information it

has received from all the incoming channels up to the end of the time slot $t-1$. The accumulative information includes the sequence $m_{d,0}, m_{d,1}, \ldots, m_{d,t-1}$ for each incoming channel d. The calculation is by the convolution (3.5) which can be implemented by circuitry in the causal manner, because the local encoding kernels in Definition 3.5 as well as the components of the global encoding kernels belong to $F\langle z \rangle$.

Example 3.7. Consider the convolutional network code in Example 3.6. When the source pipelines the message

$$x(z) = \left[\textstyle\sum_{t\geq 0} a_t z^t \ \sum_{t\geq 0} b_t z^t \right],$$

the five channels in the network carry the following power series, respectively:

$$\begin{aligned}
x(z)\cdot f_{SX}(z) &= \sum_{t\geq 0} a_t z^{t+1} \\
x(z)\cdot f_{SY}(z) &= \sum_{t\geq 0} b_t z^{t+1} \\
x(z)\cdot f_{XY}(z) &= \left(\sum_{t\geq 0} a_t z^{t+2} + \sum_{t\geq 0} b_t z^{t+4} \right) \Big/ (1-z^3) \\
&= \left(\sum_{t\geq 0} a_t z^{t+2} + \sum_{t\geq 0} b_t z^{t+4} \right) \sum_{t\geq 0} z^{3t} \\
&= a_0 z^2 + a_1 z^3 + (a_2+b_0) z^4 + (a_0+a_3+b_1) z^5 \\
&\qquad + (a_1+a_4+b_2) z^6 + (a_2+a_5+b_0+b_3) z^7 + \cdots \\
x(z)\cdot f_{YW}(z) &= \left(\sum_{t\geq 0} a_t z^{t+3} + \sum_{t\geq 0} b_t z^{t+2} \right) \Big/ (1-z^3) \\
x(z)\cdot f_{WX}(z) &= \left(\sum_{t\geq 0} a_t z^{t+4} + \sum_{t\geq 0} b_t z^{t+3} \right) \Big/ (1-z^3).
\end{aligned}$$

At each time slot $t \geq 0$, the source generates a message $x_t = [a_t, b_t]$. Thus, the channel SX carries the scalar 0 at time 0 and the scalar a_{t-1} at time t > 0. Similarly, the channel SY carries the scalar 0 at time 0

and the scalar b_{t-1} at time $t > 0$. For every channel e, write

$$\left(\sum_{t\geq 0} x_t z^t\right) \cdot f_e(z) = \sum_{t\geq 0} m_{e,t} z^t$$

as before. The actual encoding process at the node X is as follows. At the end of the time slot $t-1$, the node X has received the sequence $m_{d,0}, m_{d,1}, \ldots, m_{d,t-1}$ for $d = SX$ and WX. Accordingly, the channel XY at time $t > 0$ transmits the scalar value

$$\begin{aligned} m_{XY,t} &= \sum_{0\leq u<t} k_{SX,XY,u} m_{SX,t-1-u} + \sum_{0\leq u<t} k_{WX,XY,u} m_{WX,t-1-u} \\ &= m_{SX,t-1} + m_{WX,t-1}, \end{aligned}$$

with the convention that $m_{e,t} = 0$ for all channels e and $t < 0$. Similarly,

$$m_{YW,t} = m_{SY,t-1} + m_{XY,t-1}$$

and

$$m_{WX,t} = m_{YW,t-1}$$

for $t \geq 0$. (Note that $m_{XY,0} = m_{YW,0} = m_{WX,0} = 0$.) The values $m_{XY,t}$, $m_{YW,t}$, and $m_{WX,t}$ for $t = 0, 1, 2, 3, \ldots$ can be calculated recursively by these formulas, and they are shown in the trellis network in Figure 3.3 for small values of t. For instance, the channel XY carries the scalar values

$$\begin{aligned} &m_{XY,0} = 0,\ m_{XY,1} = 0,\ m_{XY,2} = a_0,\ m_{XY,3} = a_1, \\ &m_{XY,4} = a_2 + b_0,\ m_{XY,5} = a_0 + a_3 + b_1, \ldots \end{aligned}$$

in the initial time slots. The z-transform of this sequence is

$$x(z) \cdot f_{XY}(z) = \left(\sum_{t\geq 0} a_t z^{t+2} + \sum_{t\geq 0} b_t z^{t+4}\right) \Big/ (1 - z^3)$$

as calculated in the above. The encoding formulas in this example are especially simple, and the convolution in (3.5) is rendered trivial. Because all the local encoding kernels are scalars, the encoder at

a node does not require the memory of any previously received information other than the scalar value that has just arrived from each incoming channel. However, the scalar local encoding kernels do not offer similar advantage to the decoding process at the receiving nodes. This will be further discussed in the next example.

Example 3.8. Figure 3.5 presents another 2-dimensional convolutional network code on the same cyclic network. The salient characteristic of this convolutional network code is that every component of the global encoding kernel for every channel is simply a power of z. This simplicity renders decoding at every receiving nodes almost effortless. On the other hand, the encoders at the nodes in this case are only slightly more complicated than those in the preceding example. Thus, in terms of the total complexity of encoding and decoding, the present convolutional network code is more desirable.

Again, let the source generate a message $x_t = [a_t, b_t]$ at each time slot $t \geq 0$. Thus, the channel SX carries the scalar 0 at time 0 and the scalar a_{t-1} at time $t > 0$. Similarly, the channel SY carries the scalar 0

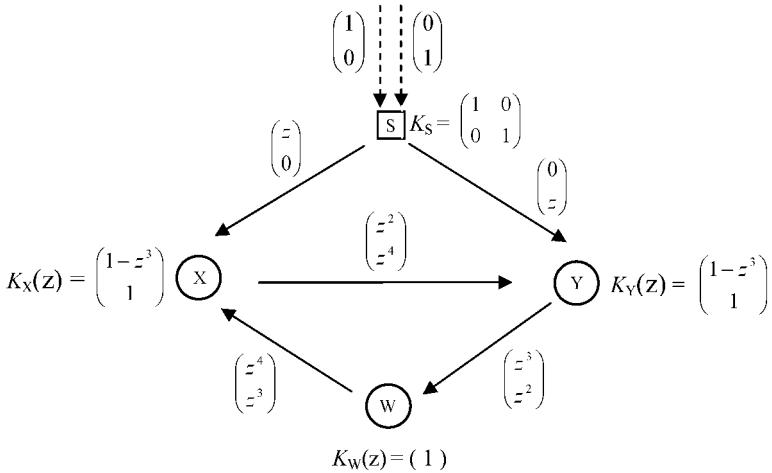

Fig. 3.5 When every component of the global encoding kernel for every channel is simply a power of z, the decoding of the convolutional network code at every receiving node is almost effortless.

at time 0 and the scalar b_{t-1} at time $t > 0$. For every channel e, write

$$\left(\sum_{t\geq 0} x_t z^t\right) \cdot f_e(z) = \sum_{t\geq 0} m_{e,t} z^t$$

as before. At the end of the time slot $t-1$, the node T has received the sequence $m_{d,0}, m_{d,1}, \ldots, m_{d,t-1}$ for $d = SX$ and WX. Accordingly, the channel XY at time t > 0 transmits the value

$$m_{XY,t} = \sum_{0\leq u<t} k_{SX,XY,u} m_{SX,t-1-u} + \sum_{0\leq u<t} k_{WX,XY,u} m_{WX,t-1-u}.$$

In this case, $k_{SX,XY,0} = k_{WX,XY,0} = 1$, $k_{WX,XY,u} = 0$ for all $u > 0$, $k_{SX,XY,3} = -1$, and $k_{SX,XY,u} = 0$ for all $u \neq 0$ or 3. Thus,

$$m_{XY,t} = m_{SX,t-1} - m_{SX,t-4} + m_{WX,t-1},$$

with the convention that $m_{e,t} = 0$ for all channels e and $t < 0$. Similarly,

$$m_{YW,t} = m_{SY,t-1} - m_{SY,t-4} + m_{XY,t-1}$$

and

$$m_{WX,t} = m_{YW,t-1}$$

for $t > 0$. The values $m_{XY,t}$, $m_{YW,t}$, and $m_{WX,t}$ for $t = 0, 1, 2, 3, \ldots$ can be calculated by these formulas, and they are shown in the trellis network in Figure 3.6 for small values of t.

Take the channel XY as an example. The encoder for this channel is to implement the arithmetic of

$$\begin{aligned} m_{XY,t} &= m_{SX,t-1} - m_{SX,t-4} + m_{WX,t-1} \\ &= a_{t-2} - a_{t-5} + (a_{t-5} + b_{t-4}) \\ &= a_{t-2} + b_{t-4}, \end{aligned}$$

which incorporates both the local encoding kernels $k_{SX,XY}(z)$ and $k_{WX,XY}(z)$. This only requires the simple circuitry in Figure 3.7, where an element labeled "z" is for a unit-time delay.

A convolutional network code over a unit-delay network can be viewed as a linear time-invariant (LTI) system defined by the local

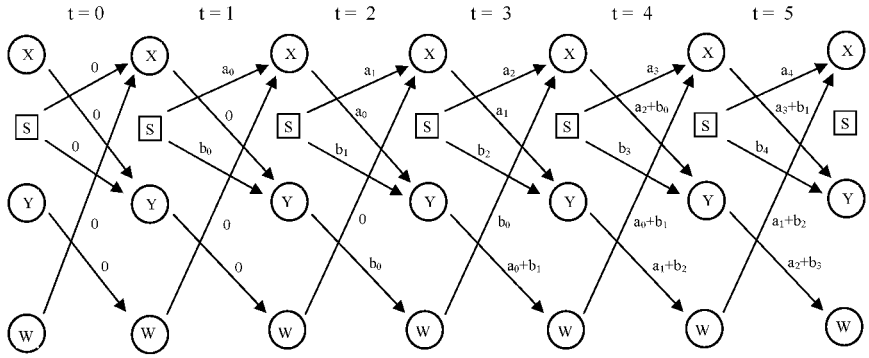

Fig. 3.6 Message transmission via a linear network code on a cyclic network means the pipelining of sequential symbols through every channel. The transmission media in the time-space domain is an indefinitely long "trellis network," where every channel carried a scalar value at each time slot.

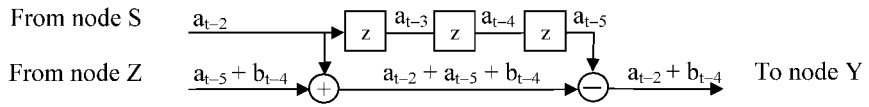

Fig. 3.7 Circuitry for the encoding at the node X for the convolutional network code in Figure 3.5, where an element labeled "z" is for a unit-time delay.

encoding kernels, which therefore uniquely determine the global encoding kernels. More explicitly, given $k_{d,e}(z) \in F\langle z \rangle$ for all adjacent pairs (d, e), there exists a unique solution to (3.1) and (3.2) for $f_e(z)$ for all channels e. The following theorem further gives a simple close-form formula for $f_e(z)$ and shows that the entries in $f_e(z)$ indeed belong to $F\langle z \rangle$, i.e., $f_e(z)$ is a rational power series, a requirement by Definition 3.5 for an F-valued convolutional network code.

Theorem 3.9. Let F be a finite field and ω a positive integer. Let $k_{d,e}(z) \in F\langle z \rangle$ be given for every adjacent pair (d, e) on a unit-delay network. Then there exists a unique ω-dimensional F-valued convolutional network code with $k_{d,e}(z)$ as the local encoding kernel for every (d, e).

Proof. Let N be the number of channels in the network, not counting the imaginary channels in $\mathrm{In}(S)$. Given an ω-dimensional vector $g_e(z)$

for every channel e, we shall adopt the notation $[g_e(z)]$ for the $\omega \times N$ matrix that puts the vectors $g_e(z)$ in juxtaposition. Let $H_S(z)$ denote the particular $\omega \times N$ matrix $[g_e(z)]$ such that, when $e \in \mathrm{Out}(S)$, $g_e(z)$ is composed of the given $k_{d,e}(z)$ for all the imaginary channels d and otherwise $g_e(z)$ is the zero vector. In other words, $H_S(z)$ is formed by appending $N - |\mathrm{Out}(S)|$ columns of zeroes to the local encoding kernel $K_S(z)$ at the node S, which is an $\omega \times |\mathrm{Out}(S)|$ matrix.

Let $[k_{d,e}(z)]$ denote the $N \times N$ matrix in which both the rows and columns are indexed by the channels and the (d,e)-th entry is equal to the given $k_{d,e}(z)$ if (d,e) is an adjacent pair, and is equal to zero otherwise. In order to have an ω-dimensional F-valued convolutional network code with $k_{d,e}(z)$ as the local encoding kernels, the concomitant global encoding kernels $f_e(z)$ must meet the requirements (3.1) and (3.2), which can be translated into the matrix equation

$$[f_e(z)] = z[f_e(z)] \cdot [k_{d,e}(z)] + zH_S(z),$$

or equivalently,

$$[f_e(z)] \cdot (I_N - z[k_{d,e}(z)]) = zH_S(z), \tag{3.6}$$

where is I_N the $N \times N$ identity matrix. Clearly, $\det(I_N - z[k_{d,e}(z)])$ is of the form $1 + zq(z)$, where $q(z) \in F\langle z \rangle$. Hence, $\det(I_N - z[k_{d,e}(z)])$ is invertible inside $F\langle z \rangle$. The unique solution of (3.6) for $[f_e(z)]$ is given by

$$[f_e(z)] = z \det(I_N - z[k_{d,e}(z)])^{-1} H_S(z) \cdot A(z), \tag{3.7}$$

where $A(z)$ denotes the adjoint matrix of $I_N - z[k_{d,e}(z)]$. Thus $[f_e(z)]$ is a matrix over $F\langle z \rangle$. With the two matrices $[k_{d,e}(z)]$ and $H_S(z)$ representing the given local encoding kernels and the matrix $[f_e(z)]$ representing the global encoding kernels, (3.7) is a close-form expression of the latter in terms of the former. □

In retrospect, Definition 3.5 may be regarded as the "global description" of a convolutional network over a unit-delay network, while Theorem 3.9 allows a "local description" by specifying only the local encoding kernels.

3.3 Decoding of convolutional network code

In this section, we define a *convolutional multicast*, the counterpart of a *linear multicast* defined in Section 2, for a unit-delay cyclic network. The existence of a convolutional multicast is also established.

Definition 3.10. Let $f_e(\mathrm{z})$ be the global encoding kernel for each channel e in an ω-dimensional F-valued convolutional network code over a unit-delay network. At every node T, let $[f_e(z)]_{e\in In(T)}$ denote the $\omega \times |\mathrm{In}(T)|$ matrix that puts vectors $f_e(z)$, $e \in \mathrm{In}(T)$, in juxtaposition. Then the convolutional network code qualifies as an ω-dimensional *convolutional multicast* if

(3.8) For every non-source node T with $\mathrm{maxflow}(T) \geq \omega$, there exists an $|\mathrm{In}(T)| \times \omega$ matrix $D_T(z)$ over $F\langle z\rangle$ and a positive integer τ such that $[f_e(z)]_{e\in In(T)} \cdot D_T(z) = z^\tau I_\omega$, where τ depends on the node T and I_ω is the $\omega \times \omega$ identity matrix.

The matrix $D_T(z)$ are called the *decoding kernel* and the *decoding delay* at the node T, respectively.

Let the source node S generate the message pipeline $x(z) = \sum_{t\geq 0} x_t z^t$, where x_t is an ω-dimensional row vector in F^ω, so that $x(z)$ is an ω-dimensional row vector over $F[[z]]$. Through the convolutional network code, a channel e carries the power series $x(z) \cdot f_e(z)$. The power series $x(z) \cdot f_e(z)$ received by a node T from all the incoming channels e form the $|\mathrm{In}(T)|$-dimensional row vector $x(z)\cdot[f_e(z)]_{e\in In(T)}$ over $F[[z]]$. When the convolutional network code is a convolutional multicast, the node T then uses the decoding kernel $D_T(\mathrm{z})$ to calculate

$$\big(x(z)\cdot[f_e(z)]_{e\in In(T)}\big) \cdot D_T(\mathrm{z}) = x(z) \cdot \big([f_e(z)]_{e\in In(T)} \cdot D_T(z)\big) = z^\tau x(z).$$

The ω-dimensional row vector $z^\tau x(z)$ of power series represents the message pipeline generated by S after a delay of τ unit times. Note that $\tau > 0$ because the message pipeline $x(z)$ is delayed by one unit time at the source node S.

The above discussion is illustrated by the two examples below, where we again let the source node S generate the message pipeline

$$x(z) = \left[\sum_{t\geq 0} a_t z^t \ \sum_{t\geq 0} b_t z^t \right].$$

Example 3.11. Consider the node X in the network in Figure 3.4. We have

$$[f_e(z)]_{e\in \mathrm{In}(X)} = \begin{bmatrix} z & z^4/(1-z^3) \\ 0 & z^3/(1-z^3) \end{bmatrix}.$$

Let

$$D_X(z) = \begin{bmatrix} z^2 & -z^3 \\ 0 & 1-z^3 \end{bmatrix}.$$

Then

$$[f_e(z)]_{e\in In(X)} \cdot D_T(z) = z^3 I_2,$$

where I_2 denotes the 2×2 identity matrix. From the channels SX and WX, the node X receives the row vector

$$x(z)\cdot[f_e(z)]_{e\in In(X)} = \left[\sum_{t\geq 0} a_t z^{t+1} \ \sum_{t\geq 0} \frac{a_t z^{t+4} + b_t z^{t+3}}{1-z^3} \right]$$

and decodes the message pipeline as

$$z^3 x(z) = \left[\sum_{t\geq 0} a_t z^{t+1} \ \sum_{t\geq 0} \frac{a_t z^{t+4} + b_t z^{t+3}}{1-z^3} \right] \cdot \begin{bmatrix} z^2 & -z^3 \\ 0 & 1-z^3 \end{bmatrix}.$$

Decoding at the node Y is similar. Thus, the 2-dimensional convolutional network code in this case is a convolutional multicast.

Example 3.12. The 2-dimensional convolutional network code in Figure 3.5 is also a convolutional multicast. Take the decoding at the node X as an example. We have

$$[f_e(z)]_{e\in In(X)} = \begin{bmatrix} z & z^4 \\ 0 & z^3 \end{bmatrix}.$$

Let

$$D_X(z) = \begin{bmatrix} z^2 & -z^3 \\ 0 & 1 \end{bmatrix}.$$

Then

$$[f_e(z)]_{e\in In(X)} \cdot D_X(z) = z^3 I_2.$$

From the channels SX and WX, the node X receives the row vector $x(z)\cdot[f_e(z)]_{e\in \mathrm{In}(X)}$ and decodes the message pipeline as

$$\begin{aligned} z^3 x(z) &= x(z)\cdot[f_e(z)]_{e\in In(X)}\cdot \begin{bmatrix} z^2 & -z^3 \\ 0 & 1 \end{bmatrix} \\ &= \left[\sum_{t\geq 0} \mathrm{a}_t \mathrm{z}^{t+1} \sum_{t\geq 0} (\mathrm{a}_t \mathrm{z}^{t+4} + \mathrm{b}_t \mathrm{z}^{t+3})\right] \cdot \begin{bmatrix} z^2 & -z^3 \\ 0 & 1 \end{bmatrix}. \end{aligned}$$

Having formulated a convolutional multicast, the natural concern is its existence. Toward proving the existence of a convolutional multicast, we first observe that Lemma 2.17 can be strengthened as follows with essentially no change in the proof.

Lemma 3.13. Let $g(y_1, y_2, \ldots, y_m)$ be a nonzero polynomial with coefficients in a field G. For any subset E of G, if $|E|$ is greater than the degree of g in every y_j, then there exist $a_1, a_2, \ldots, a_m \in E$ such that $g(a_1, a_2, \ldots, a_m) \neq 0$. The values $a_1, a_2, \ldots, a_m$ can be found by exhaustive search in E provided that E is finite. If E is infinite, simply replace E by a sufficiently large finite subset of E.

Theorem 3.14. Given a unit-delay network, a finite field F, and a positive integer ω, there exists an ω-dimensional F-valued convolutional multicast. Furthermore, if E is a sufficiently large subset of $F\langle z\rangle$, then the local encoding kernels of the convolutional multicast can be chosen to take values from E.

Proof. From Theorem 3.9, a set of arbitrarily given local encoding kernels uniquely determines a convolutional network code on a unit-delay network. Following the proof of that theorem, the global encoding kernels $f_e(z)$ concomitant to the given local encoding kernels

$k_{d,e}(z) \in F\langle z\rangle$ are calculate by (3.7). We shall show that the global encoding kernels $f_e(z)$ meet the requirement (3.8) for a convolutional multicast when $k_{d,e}(z)$ are appropriately chosen.

Restate (3.7) as

$$\det(I_n - z[k_{d,e}(z)])[f_e(z)] = zH_S(z)\cdot A(z). \tag{3.9}$$

We now treat the local encoding kernels $k_{d,e}(z)$ as $\sum_T |In(T)| \cdot |Out(T)|$ indeterminates. Thus all the entries in the $\omega \times N$ matrix $zH_S(z)\cdot A(z)$, as well as $\det(I_N - z[k_{d,e}(z)])$, are polynomials in these indeterminates over the integral domain $F\langle z\rangle$. Denote by $(F\langle z\rangle)[*]$ the polynomial ring in these indeterminates over $F\langle z\rangle$.

Let T be a non-source node with maxflow$(T) \geq \omega$. Then there exist ω disjoint paths starting at the ω imaginary channels and ending at ω distinct channels in In(T), respectively. Let $L_T(z)$ be the $\omega \times \omega$ matrix that puts the global encoding kernels of these ω channels in juxtaposition. Thus $L_T(Z)$ is an $\omega \times \omega$ matrix over $(F\langle z\rangle)[*]$. Claim that:

$$\det(L_T(z)) \neq 0 \in (F\langle z\rangle)[*]. \tag{3.10}$$

Toward proving this claim, it suffices to show that $\det(L_T(z)) \neq 0 \in F\langle z\rangle$ when evaluated at some particular values of the indeterminates $k_{d,e}(z)$. Arguing similarly as in the alternative proof of Corollary 2.24, we set the indeterminates $k_{d,e}(z)$ to 1 for all adjacent pairs (d, e) along any one of the ω disjoint paths and to 0 otherwise. Then the matrix $L_T(z)$ becomes diagonal with all the diagonal entries being powers of z. Hence $\det(L_T(z))$ also becomes a power of z. This proves the claim.

The statement (3.10) applies to every non-source node T with maxflow$(T) \geq \omega$. Thus

(3.11) $\prod_{T:\text{maxflow}(T)\geq\omega} \det(L_T(z)) \neq 0$ in $(F\langle z\rangle)[*]$.

Apply Lemma 3.13 to $G = F(z)$, where $F(z)$ is the conventional notation for the field of rational functions over F. We can choose a value $a_{d,e}(z) \in E \subset F\langle z\rangle \subset F(z)$ for each of the indeterminates $k_{d,e}(z)$ so that

(3.12) $\prod_{T:\text{maxflow}(T)\geq\omega} \det(L_T(z)) \neq 0$ in $(F\langle z\rangle)[*]$ when evaluated at $k_{d,e}(z) = a_{d,e}(z)$ for all (d, e).

As the integral domain $F\langle z\rangle$ is infinite, this statement applies in particular to the case where $E = F\langle z\rangle$.

From now on, the local encoding kernel $k_{d,e}(z)$ will be fixed at the appropriately chosen value $a_{d,e}(z)$ for all (d, e). Denote by $J_T(z)$ the adjoint matrix of $L_T(z)$. Without loss of generality, we shall assume that $L_T(z)$ consists of the first ω columns of $[f_e(z)]_{e\in In(T)}$. From (3.12), $L_T(z)$ is a nonsingular matrix over $F\langle z\rangle$. Therefore, we can write

$$\det(L_T(z)) = z^t(1 + zq(z))/p(z),$$

where τ is some positive integer, and $p(z)$ and $q(z)$ are polynomials over F. Take the $\omega \times \omega$ matrix $[p(z)/(1 + zq(z))]J_T(z)$ and append to it $|\mathrm{In}(T)| - \omega$ rows of zeroes to form an $|\mathrm{In}(T)| \times \omega$ matrix $D_T(z)$. Then,

$$\begin{aligned}[f_e(z)]_{e\in In(T)}\cdot D_T(z) &= [p(z)/(1 + zq(z))]L_T(z)\cdot J_T(z)\\ &= [p(z)/(1 + zq(z))]\det(L_T(z))I_\omega\\ &= z^\tau I_\omega,\end{aligned}$$

where I_ω denotes the $\omega \times \omega$ identity matrix. Thus the matrix $D_T(z)$ meets the requirement (3.8) for a convolutional multicast. □

When F is a sufficiently large finite field, this theorem can be applied with $E = F$ so that the local encoding kernels of the convolutional multicast can be chosen to be scalars. This special case is the convolutional counterpart to Corollary 2.24 on the existence of a linear multicast over an acyclic network. In this case, the local encoding kernels can be found by exhaustive search over F. This result was first established in [184].

More generally, by virtue of Lemma 3.13, the same exhaustive search applies to any large enough subset E of $F\langle z\rangle$. For example, F can be $GF(2)$ and E can be the set of all binary polynomials up to a sufficiently large degree. More explicit and efficient construction of a convolutional multicast over the integral domain of binary rational power series have been reported in [171][174][172].

4

Network Coding and Algebraic Coding

Algebraic coding theory deals with the design of error-correcting/ erasure channel codes using algebraic tools for reliable transmission of information across noisy channels. As we shall see in this section, there is much relation between network coding theory and algebraic coding theory, and in fact, algebraic coding can be viewed as an instance of network coding. For comprehensive treatments of algebraic coding theory, we refer the reader to [161][190][162][205].

4.1 The combination network

Consider a classical (n, k) linear block code with *generator matrix* G, where G is a $k \times n$ matrix over some base field F. As discussed in the remark following Definition 2.5, the global encoding kernels are analogous to the columns of the generator matrix of a classical linear block code. It is therefore natural to formulate an (n, k) linear block code as a linear network code on the network in Figure 4.1. In this network, a channel connects the source node S to each of the n non-source node. Throughout this section, we shall assume that there are k imaginary channels at the the source node, i.e., the dimension of the

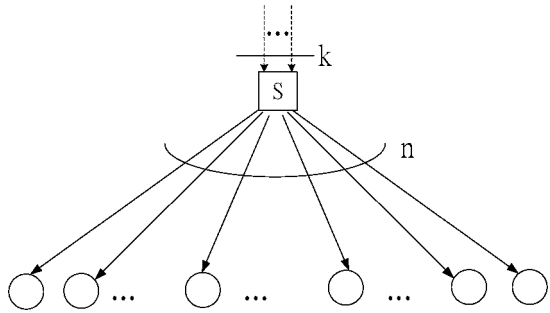

Fig. 4.1 A network representation of a classical linear block code.

network code is k. The linear network code is specified by taking the global encoding kernels of the n edges in $\mathrm{Out}(S)$ to be the columns of G, or equivalently, by taking K_S, the local encoding kernel of the source node S, to be G. Traditionally, the columns of the generator matrix G are indexed in "time." In the network coding formulation, however, they are indexed in "space." It is readily seen that the symbols received by the non-source nodes in Figure 4.1 constitute the codeword of the classical linear block code.

The above formulation is nothing but just another way to describe a classical linear block code. In order to gain further insight into the relation between network coding and algebraic coding, we consider the network in Figure 4.2, which is an extension of the network in Figure 4.1. In this network, the top two layers are exactly as the network in Figure 4.1. The bottom layer consists of $\binom{n}{r}$ nodes, each connecting to a distinct subset of r nodes on the middle layer. We call this network an $\binom{n}{r}$ *combination network*, or simply an $\binom{n}{r}$ network, where $1 \leq r \leq n$.

4.2 The Singleton bound and MDS codes

Consider a classical (n,k) linear block code with *minimum distance* d and regard it as a linear network code on the $\binom{n}{n-d+1}$ network. In this network, the assignment of global encoding kernels for the channels between the first layer and the second layer is the same as in Figure 4.1. For each node on middle layer, since there is only one input channel,

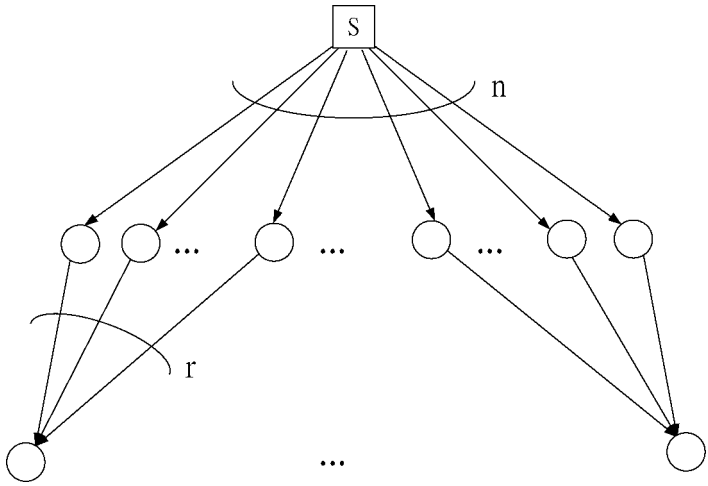

Fig. 4.2 An $\binom{n}{r}$ combination network.

we assume without loss of generality that the global encoding kernel of all the output channels are the same as that of the input channel.

Since the (n, k) code has minimum distance d, by accessing a subset of $n - d + 1$ of the nodes on the middle layer (corresponding to $d - 1$ erasures), each node T on the bottom layer can decode the message x generated at the source node uniquely, where x consists of k symbols from F. Then by the Max-flow Min-cut theorem,

$$\text{maxflow}(T) \geq k. \tag{4.1}$$

Since

$$\text{maxflow}(T) = n - d + 1,$$

it follows that

$$k \leq n - d + 1,$$

or

$$d \leq n - k + 1, \tag{4.2}$$

which is precisely the Singleton bound [202] for classical linear block code. Thus the Singleton bound is a special case of the Max-flow

Min-cut theorem. Moreover, by (4.1), the non-source nodes in the network with maximum flow at least equal to k are simply all the nodes on the bottom layer, and each of them can decode the message x. Hence, we conclude that an (n, k) classical linear block code with minimum distance d is a k-dimensional linear multicast on the $\binom{n}{n-d+1}$ network.

More generally, an (n, k) classical linear block code with minimum distance d is a k-dimensional linear multicast on the $\binom{n}{r}$ network for all $r \geq n - d + 1$. The proof is straightforward (we already have shown it for $r = n - d + 1$). On the other hand, it is readily seen that a k-dimensional linear multicast on the $\binom{n}{r}$ network, where $r \geq k$, is an (n, k) classical linear block code with minimum distance d such that

$$d \geq n - r + 1.$$

A classical linear block code achieving tightness in the Singleton bound is called a *maximum distance separation* (MDS) code [202]. From the foregoing, the Singleton bound is a special case of the Max-flow Min-cut theorem. Since a linear multicast, broadcast, or dispersion achieves tightness in the Max-flow Min-cut theorem to different extents, they can all be regarded as network generalizations of an MDS code. The existence of MDS codes corresponds, in the more general paradigm of network coding, to the existence of linear multicasts, linear broadcasts, linear dispersions, and generic linear network codes, which have been discussed in great detail in Section 2.

4.3 Network erasure/error correction and error detection

Consider the network in Figure 4.3, which is the setup of a classical point-to-point communication system. A message of k symbols is generated at the node S and is to be transmitted to the node T via n channels, where $n \geq k$. For a linear network code on this network to be qualified as a static linear multicast, if no more than $(n - k)$ channels are removed (so that $\text{maxflow}(T) \geq k$), the message x can be decoded at the node T. Equivalently, a static linear multicast on this network can be described as a classical (n, k) linear block code that can correct $(n - k)$ erasures. Therefore, a static linear multicast can be viewed as a network generalization of a classical erasure-correcting code.

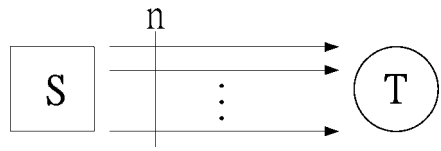

Fig. 4.3 A classical point-to-point communication system.

It is evident that a linear multicast on the network in Figure 4.2 is a static linear multicast on the network in Figure 4.3, and vice versa. An (n,k) MDS code, whose minimum distance is $(n-k+1)$, can correct up to $(n-k)$ erasures. So it is readily seen that an (n,k) MDS code is a static linear multicast on the network in Figure 4.3. Thus a static linear multicast can also be viewed as a network generalization of an MDS code.

A static linear multicast, broadcast, or dispersion is a network code designed for erasure correction in a point-to-point network. In the same spirit, a network code can also be designed for error detection or error correction. For the former, the use of random error detection codes for robust network communications has been investigated in [180]. For the latter, network generalizations of the Hamming bound, the Singleton bound, and the Gilbert-Varshamov bound for classical error-correcting codes have been obtained in [165][210][164]. Some basic properties and the constructions of network error-correcting codes have been studied in [213].

4.4 Further remarks

A primary example of an MDS code is the Reed-Solomon code [198]. The construction of a Reed-Solomon code is based on the Vandermonde matrix, which has the form

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_k \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_k^2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{k-1} & \alpha_2^{k-1} & \cdots & \alpha_k^{k-1} \end{bmatrix},$$

where $k \geq 1$ and α_i, $1 \leq i \leq k$ are distinct elements in some field F (in our context F is taken to be a finite field). The essential properties of the Vandermonde matrix in the context of algebraic coding are that i) each column has exactly the same form and is parametrized by one field element; ii) its determinant is always nonzero. By appending columns of the same form parametrized by distinct field elements to a Vandermonde matrix, the generator matrix of a Reed-Solomon code is obtained.

The constructions of linear multicast, linear broadcast, linear dispersion, and generic linear network code may be regarded as extensions of the kind of matrix construction rendered by the Vandermonde matrix. However, although the constructions of these network codes are explicit as discussed in Section 2, they are not in closed-form as the Vandermonde matrix.

Fountain codes [163][193], a class of randomly generated rateless erasure codes, are finding applications in robust network communications. They guarantee near-optimal bandwidth consumption as well as very efficient decoding with high probability. The random linear network codes discussed in [192][176][191] may be regarded as a kind of generalization of fountain codes, except that very efficient decoding algorithms do not exist for such codes. The main distinction between these codes and fountain codes is that a fountain code may encode only at the source node, while a network code may encode at every node in the network[1].

[1] In the setting of a fountain code, the communication network between the source node and a receiving node is basically modeled as a classical point-to-point communication system as in Figure 4.3.

Part II

MULTIPLE SOURCES

5

Superposition Coding and Max-Flow Bound

In Part I of this tutorial, we have discussed the single-source network coding problem in an algebraic setting. Each communication channel in the network is assumed to have unit capacity. The maximum rate at which information can be multicast has a simple characterization in terms of the maximum flows in the network. In Part II, we consider the more general multi-source network coding problem in which more than one *mutually independent* information sources are generated at possibly different nodes, where each information source is transmitted to a certain set of nodes in the network. We continue to assume that the communication channels in the network are free of error.

The *achievable information rate region* for a multi-source network coding problem, which will be formally defined in Section 6, refers to the set of all possible rate tuples at which multiple information sources can be multicast simultaneously on a network. In a single-source network coding problem, a primary goal is to characterize the maximum rate at which information can be multicast from the source node to all the sink nodes. In a multi-source network coding problem, we are interested in characterizing the achievable information rate region.

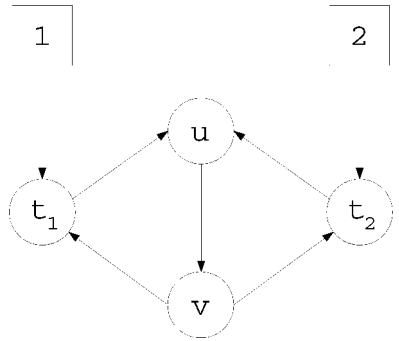

Fig. 5.1 A network for which superposition coding is suboptimal.

Multi-source network coding turns out *not* to be a simple extension of single-source network coding. In the rest of this section, we discuss two characteristics of multi-source networking coding which differentiate it from single-source network coding. In all the examples, the unit of information is the bit.

In Part I, nodes are labelled by capital letters. In Part II, since captical letters are reserved for random variables, nodes will instead be labelled by small letters.

5.1 Superposition coding

Let us first revisit the network in Figure 1.2(b) of Part I which is reproduced here as Figure 5.1 in a slightly different manner. Here, we assume that each channel has unit capacity. For $i = 1, 2$, the source node i generates a bit b_i which is sent to the node t_i. We have shown in Example 1.3 of Part I that in order for the nodes t_1 and t_2 to exchange the two bits b_1 and b_2, network coding must be performed at the node u. This example in fact has a very intriguing implication. Imagine that on the Internet a message in English and a message in Chinese are generated at two different locations. These two messages are to be transmitted from one point to another point within the network, and we can assume that there is no correlation between the two messages. Then this example shows that we may have to perform joint coding of the two messages in the network in order to achieve bandwidth optimality!

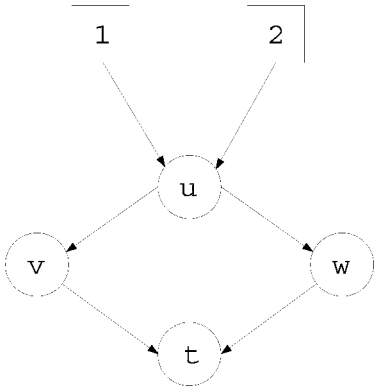

Fig. 5.2 A network for which superposition coding is optimal.

We refer to the method of coding individual information sources separately as *superposition coding*. The above example simply shows that superposition coding can be suboptimal.

We now give an example for which superposition coding does achieve optimality. Consider the network in Figure 5.2. To simply the discussion, we set the capacities of the channels $1u$ and $2u$ to infinity so that the information generated at both source nodes are directly available to the node u. For all the other channels, we set the capacity to 1. We want to multicast the information generated at the source node 1 to the nodes v, w and t, and to transmit the information generated at the source node 2 to the node t.

Let X_1 and X_2 be independent random variables representing the information generated respectively at the source nodes 1 and 2 for one unit time. The rate of the information generated at the source node s is given by $\omega_s = H(X_s)$ for $s = 1, 2$. Let U_{ij} be the random variable sent on the channel ij, where $H(U_{ij}) \leq 1$ due to the bit rate constraint for the channel. Then for any coding scheme achieving the prescribed communication goals, we have

$$\begin{aligned} 2\omega_1 + \omega_2 &= 2H(X_1) + H(X_2) \\ &= 2H(X_1) + H(X_2|X_1) \\ &\overset{a)}{\leq} 2H(X_1) + H(U_{vt}, U_{wt}|X_1) \end{aligned}$$

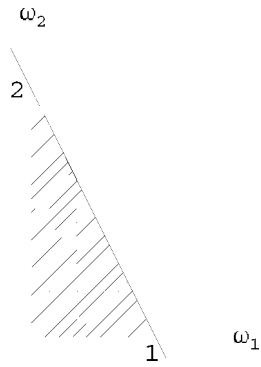

Fig. 5.3 The information rate region for the network in Figure 5.2.

$$\begin{aligned}
&\overset{b)}{\leq} 2H(X_1) + H(U_{uv}, U_{uw}|X_1) \\
&\leq 2H(X_1) + H(U_{uv}|X_1) + H(U_{uw}|X_1) \\
&= H(U_{uv}, X_1) + H(U_{uw}, X_1) \\
&\overset{c)}{=} H(U_{uv}) + H(U_{uw}) \\
&\leq 2,
\end{aligned}$$

where a) follows because X_2 is a function of U_{vt} and U_{wt}, b) follows because U_{vt} is a function of U_{uv} and U_{wt} is a function of U_{uw}, and c) follows because X_1 is a function of U_{uv} and a function of U_{uw}.

This region is illustrated in Figure 5.3. To see that the whole region is achievable by superposition coding, let $r_{ij}^{(s)}$ be the bit rate on the channel ij for transmitting the information generated at the source node s. Due to the bit rate constraint for each channel ij, the following must be satisfied:

$$r_{ij}^{(1)} + r_{ij}^{(2)} \leq 1.$$

Then the rate pair $(\omega_1, \omega_2) = (1, 0)$ is achieved by taking

$$r_{uv}^{(1)} = r_{uw}^{(1)} = r_{vt}^{(1)} = 1$$

and

$$r_{wt}^{(1)} = r_{uv}^{(2)} = r_{uw}^{(2)} = r_{vt}^{(2)} = r_{wt}^{(2)} = 0,$$

while the rate pair $(0,2)$ is achieved by taking

$$r_{uv}^{(1)} = r_{uw}^{(1)} = r_{vt}^{(1)} = r_{wt}^{(1)} = 0$$

and

$$r_{uv}^{(2)} = r_{uw}^{(2)} = r_{vt}^{(2)} = r_{wt}^{(2)} = 1.$$

Then the whole information rate region depicted in Figure 5.3 is seen to be achievable via a time-sharing argument.

From the above two examples, we see that superposition coding is sometimes but not always optimal. Optimality of superposition coding for certain classes of multilevel diversity coding problems (special cases of multi-source network coding) has been reported in [207], [200], [212]. For a class of multilevel diversity coding problems (special cases of multi-source network coding) studied in [177], superposition coding is optimal for 86 out of 100 configurations. In any case, superposition coding always induces an inner bound on the information rate region.

5.2 The max-flow bound

In this section, we revisit the two examples in the last section from a different angle. First, for the network in Figure 5.1, we already have seen that superposition coding is suboptimal. Now consideration of the max-flows from t_1 to t_2 and from t_2 to t_1 gives

$$\omega_1, \omega_2 \leq 1.$$

This outer bound on the information rate region, referred to as the *max-flow bound*, is depicted in Figure 5.4. Here the rate pair $(1,1)$ is achieved by using network coding at the node u as we have discussed, which implies the achievability of the whole region. Therefore, the max-flow bound is tight.

We now consider the network in Figure 5.2. Consideration of the max-flow at either node v or w gives

$$\omega_1 \leq 1, \tag{5.1}$$

while consideration of the max-flow at node t gives

$$\omega_1 + \omega_2 \leq 2. \tag{5.2}$$

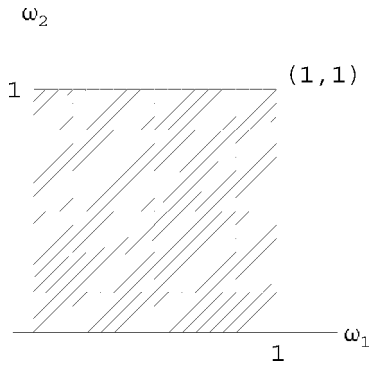

Fig. 5.4 The max-flow bound for the network in Figure 5.1.

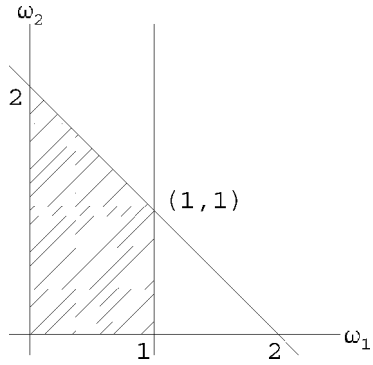

Fig. 5.5 The max-flow bound for the network in Figure 5.2.

Figure 5.5 is an illustration of the region of all (ω_1,ω_2) satisfying these bounds, which constitute the max-flow bound. Comparing with the achievable information rate region shown in Figure 5.3, we see that the max-flow bound is not tight. From these two examples, we see that like superposition coding, the max-flow bound is sometimes but not always tight. Nevertheless, it always gives an outer bound on the information rate region. It has been shown in [170][194] that the max-flow bound is tight for networks with two sink nodes.

6

Network Codes for Acyclic Networks

6.1 Achievable information rate region

In Part I, the capacity of direct transmission from a node to its neighbor is determined by the multiplicity of the channels between them. This is to facilitate the discussion of linear codes. In this section, codes not necessarily linear are considered and we assume that the capacity of a channel can take any positive real number. We, however, continue to allow multiple channels between a pair of nodes to facilitate subsequent comparison with linear codes.

Convention. The following convention applies to every acyclic communication network in this section.

- The set of all nodes and the set of all channels are denoted by V and E, respectively.
- The nodes are ordered in a way such that if there exists a channel from a node i to a node j, then the node i precedes the node j. This is possible by the acyclicity of the network.
- The capacity of a channel e is denoted by R_e.

- An independent information source X_s is generated at a source node s.
- A source node has no input channels.
- The set of all the source nodes in the network is denoted by S, which is a subset of V.
- The set of all sink nodes is denoted by T, where a sink node receives at least one information source[1]. The set of information sources received by a sink node i is denoted by $\beta(i)$.

In the above setup, the decoding requirements are described by the functions $\beta(i), i \in T$. Equivalently, we may think of each information source X_s being multicast to the set of nodes

$$\{i \in T : s \in \beta(i)\}.$$

We now consider a block code with length n. The information source X_s is a random variable which takes values in the set

$$\mathcal{X}_s = \{1, 2, \cdots, \lceil 2^{n\tau_s} \rceil\}$$

according to the uniform distribution. The rate of the information source X_s is τ_s. According to our assumption, the random variables $X_s, s \in S$ are mutually independent.

Definition 6.1. An

$$(n, (\eta_e : e \in E), (\tau_s : s \in S))$$

code on a given communication network is defined by

1) for all source node $s \in S$ and all channel $e \in \mathrm{Out}(s)$, a local encoding mapping

$$\tilde{k}_e : \mathcal{X}_s \to \{1, \cdots, \eta_e\}; \tag{6.1}$$

2) for all node $i \in V \backslash S$ and all channel $e \in \mathrm{Out}(i)$, a local encoding mapping

$$\tilde{k}_e : \prod_{d \in \mathrm{In}(i)} \{1, \cdots, \eta_d\} \to \{1, \cdots, \eta_e\}; \tag{6.2}$$

[1] Since a source node has no input channels, it cannot be a sink node.

3) for all sink node $i \in T$, a decoding mapping

$$g_i : \prod_{d \in \mathrm{In}(i)} \{1 \cdots, \eta_d\} \to \prod_{s \in \beta(i)} \mathcal{X}_s.$$

In a coding session, if a node i precedes a node j, then the encoding mappings $\tilde{k}_e, e \in \mathrm{Out}(i)$ are applied before the encoding mappings $\tilde{k}_e, e \in \mathrm{Out}(j)$. If $e, e' \in \mathrm{Out}(i)$, then $\tilde{k}_e$ and $\tilde{k}_{e'}$ can be applied in any order. Since a node i precedes a node j if there exists a channel from the node i to the node j, a node does not encode until all the necessary information is received on the input channels.

Introduce the notation $X_{S'}$ for $(X_s : s \in S')$, where $S' \subset S$. For all $i \in T$, define

$$\Delta_i = \Pr\left\{\hat{g}_i(X_S) \neq X_{\beta(i)}\right\},$$

where $\hat{g}_i(X_S)$ denotes the value of g_i as a function of X_S. Δ_i is the probability that the set of information sources $X_{\beta(i)}$ is decoded incorrectly at the node i.

In the subsequent discussion, all the logarithms are in the base 2.

Definition 6.2. An information rate tuple

$$\boldsymbol{\omega} = (\omega_s : s \in S),$$

where $\boldsymbol{\omega} \geq 0$ (componentwise), is asymptotically achievable if for any $\epsilon > 0$, there exists for sufficiently large n an

$$(n, (\eta_e : e \in E), (\tau_s : s \in S))$$

code such that

$$n^{-1} \log \eta_e \leq R_e + \epsilon$$

for all $e \in E$, where $n^{-1} \log \eta_e$ is the average bit rate of the code on the channel e,

$$\tau_s \geq \omega_s - \epsilon$$

for all $s \in S$, and

$$\Delta_i \leq \epsilon$$

for all $i \in T$. For brevity, an asymptotically achievable information rate tuple will be referred to as an achievable information rate tuple.

Definition 6.3. The achievable information rate region, denoted by $\mathcal{R}$, is the set of all achievable information rate tuples $\boldsymbol{\omega}$.

Remark 6.4. It follows from the definition of the achievability of an information rate tuple that if $\boldsymbol{\omega}$ is achievable, then $\boldsymbol{\omega}'$ is achievable for all $0 \leq \boldsymbol{\omega}' \leq \boldsymbol{\omega}$. Also, for any sequence of achievable rate tuples $\boldsymbol{\omega}^{(k)}$, $k \geq 1$, it can be proved that

$$\boldsymbol{\omega} = \lim_{k \to \infty} \boldsymbol{\omega}^{(k)},$$

if exists, is also achievable, i.e., $\mathcal{R}$ is closed. It can then be shown by invoking a time-sharing argument that $\mathcal{R}$ is closed and convex.

In this chapter, we discuss characterizations of the information rate region of a general multi-source network coding problem. Unlike single-source network coding which already has explicit algebraic code constructions, the current understanding of multi-source network coding is quite far from being complete. Specifically, only inner and outer bounds on the achievable information rate region $\mathcal{R}$ are known for *acyclic* networks, and only existence proof of codes by random coding technique is available. The tools we shall use are mainly probabilistic instead of algebraic.

We note that the definition of a network code in this section does not reduce directly to the definitions of a network code in Part I when there is only one information source. It is because in Part I, a network code is defined in a way such that various notions specific to linear codes for a single information source (namely linear broadcast, linear dispersion, and generic network code) can be incorporated. Essentially, the definition of a network code here is the local description of a network code for multicast.

6.2 Inner bound $\mathcal{R}_{in}$

In this section, we discuss an inner bound on the achievable information rate region $\mathcal{R}$ for acyclic networks. We start with some standard definitions and properties of *strong typicality*, a fundamental tool in information theory. For proofs and further details, We refer the reader to [160], [166], [209]. Here, we adopt the convention in [209].

6.2.1 Typical sequences

Consider an information source $\{X_k, k \geq 1\}$ where X_k are i.i.d. with distribution $p(x)$. We use X to denote the generic random variable, $\mathcal{S}_X$ to denote the support of X, and $H(X)$ to denote the common entropy for all X_k, where $H(X) < \infty$. Let $\mathbf{X} = (X_1, X_2, \cdots, X_n)$.

Definition 6.5. The strongly typical set $T^n_{[X]\delta}$ with respect to $p(x)$ is the set of sequences $\mathbf{x} = (x_1, x_2, \cdots, x_n) \in \mathcal{X}^n$ such that $N(x;\mathbf{x}) = 0$ for $x \notin \mathcal{S}_X$, and

$$\sum_x \left| \frac{1}{n} N(x;\mathbf{x}) - p(x) \right| \leq \delta, \tag{6.3}$$

where $N(x;\mathbf{x})$ is the number of occurrences of x in the sequence $\mathbf{x}$, and δ is an arbitrarily small positive real number. The sequences in $T^n_{[X]\delta}$ are called strongly δ-typical sequences.

Theorem 6.6. (Strong asymptotic equipartition property) In the following, η is a small positive quantity such that $\eta \to 0$ as $\delta \to 0$.

1) If $\mathbf{x} \in T^n_{[X]\delta}$, then

$$2^{-n(H(X)+\eta)} \leq p(\mathbf{x}) \leq 2^{-n(H(X)-\eta)}. \tag{6.4}$$

2) For n sufficiently large,

$$\Pr\{\mathbf{X} \in T^n_{[X]\delta}\} > 1 - \delta.$$

3) For n sufficiently large,

$$(1-\delta)2^{n(H(X)-\eta)} \le |T^n_{[X]\delta}| \le 2^{n(H(X)+\eta)}. \tag{6.5}$$

Next, we discuss strong joint typicality with respect to a bivariate distribution. Generalization to a multivariate distribution is straightforward.

Consider a bivariate information source $\{(X_k,Y_k), k \ge 1\}$ where (X_k,Y_k) are i.i.d. with distribution $p(x,y)$. We use (X,Y) to denote the pair of generic random variables, and assume that $H(X,Y) < \infty$.

Definition 6.7. The strongly jointly typical set $T^n_{[XY]\delta}$ with respect to $p(x,y)$ is the set of $(\mathbf{x},\mathbf{y}) \in \mathcal{X}^n \times \mathcal{Y}^n$ such that $N(x,y;\mathbf{x},\mathbf{y}) = 0$ for $(x,y) \notin \mathcal{S}_{XY}$, and

$$\sum_x \sum_y \left| \frac{1}{n} N(x,y;\mathbf{x},\mathbf{y}) - p(x,y) \right| \le \delta, \tag{6.6}$$

where $N(x,y;\mathbf{x},\mathbf{y})$ is the number of occurrences of (x,y) in the pair of sequences $(\mathbf{x},\mathbf{y})$, and δ is an arbitrarily small positive real number. A pair of sequences $(\mathbf{x},\mathbf{y})$ is called strongly jointly δ-typical if it is in $T^n_{[XY]\delta}$.

Strong typicality satisfies the following *consistency* and *preservation* properties.

Theorem 6.8. (Consistency) If $(\mathbf{x},\mathbf{y}) \in T^n_{[XY]\delta}$, then $\mathbf{x} \in T^n_{[X]\delta}$ and $\mathbf{y} \in T^n_{[Y]\delta}$.

Theorem 6.9. (Preservation) Let $Y = f(X)$. If

$$\mathbf{x} = (x_1, x_2, \cdots, x_n) \in T^n_{[X]\delta},$$

then

$$f(\mathbf{x}) = (y_1, y_2, \cdots, y_n) \in T^n_{[Y]\delta}, \tag{6.7}$$

where $y_i = f(x_i)$ for $1 \le i \le n$. ([209], Lemma 15.10.)

For a bivariate i.i.d. source $\{(X_k, Y_k)\}$, we have the *strong joint asymptotic equipartition property* (strong JAEP), which can readily be obtained by applying the strong AEP to the source $\{(X_k, Y_k)\}$.

Theorem 6.10. (Strong JAEP) Let

$$(\mathbf{X}, \mathbf{Y}) = ((X_1, Y_1), (X_2, Y_2), \cdots, (X_n, Y_n)),$$

where (X_i, Y_i) are i.i.d. with generic pair of random variables (X, Y). In the following, λ is a small positive quantity such that $\lambda \to 0$ as $\delta \to 0$.

1) If $(\mathbf{x}, \mathbf{y}) \in T^n_{[XY]\delta}$, then

$$2^{-n(H(X,Y)+\lambda)} \le p(\mathbf{x}, \mathbf{y}) \le 2^{-n(H(X,Y)-\lambda)}.$$

2) For n sufficiently large,

$$\Pr\{(\mathbf{X}, \mathbf{Y}) \in T^n_{[XY]\delta}\} > 1 - \delta.$$

3) For n sufficiently large,

$$(1 - \delta)2^{n(H(X,Y)-\lambda)} \le |T^n_{[XY]\delta}| \le 2^{n(H(X,Y)+\lambda)}.$$

6.2.2 First example

Consider a point-to-point communication system, the simplest possible example of a communication network:

$$V = \{1, a\},\ E = \{1a\},\ S = \{1\},\ T = \{a\},\ \beta(a) = \{1\}.$$

This network is illustrated in Figure 6.1, and we call this network G_1. By the source coding theorem [201], the information rate ω_1 is achievable if and only if $\omega_1 \le R_{1a}$. The following theorem can be regarded as an alternative form of the direct part of the source coding theorem.

Fig. 6.1 The network G_1 for the first example.

Theorem 6.11. For the network G_1, an information rate ω_1 is achievable if there exists auxiliary random variables Y_1 and U_{1a} such that

$$H(Y_1) > \omega_1 \tag{6.8}$$
$$H(U_{1a}|Y_1) = 0 \tag{6.9}$$
$$H(U_{1a}) < R_{1a} \tag{6.10}$$
$$H(Y_1|U_{1a}) = 0. \tag{6.11}$$

We first note that (6.9) and (6.11) together imply that the random variables Y_1 and U_{1a} determines each other, so we write

$$U_{1a} = u_{1a}(Y_1)$$

and

$$Y_1 = y_1(U_{1a}),$$

which imply

$$Y_1 = y_1(u_{1a}(Y_1)). \tag{6.12}$$

Moreover,

$$H(Y_1) = H(U_{1a}).$$

Then for any ω_1 satisfying (6.8) to (6.11) for some auxiliary random variables Y_1 and U_{1a}, we have

$$R_{1a} > H(U_{1a}) = H(Y_1) > \omega_1,$$

which is essentially the direct part of the source coding theorem except that the inequality is strict here. By invoking the remark following Definition 6.3, we see that the rate

$$R_{1a} = \omega_1$$

is indeed achievable.

We should think of Y_1 and U_{1a} as random variables representing the information source X_1 and the codeword sent on the channel $1a$, respectively. Accordingly, we have (6.8) as the entropy constraint on Y_1, and (6.10) corresponds to the capacity constraint for the channel $1a$.

Proof of Theorem 6.11. Let δ to be a small positive real number to be specified later. For given random variables Y_1 and U_{1a} satisfying (6.8) to (6.11), we construct a random code by the following procedure:

1. Generate $2^{n\omega_1}$ sequences of length n independently according to $p^n(y_1)$.
2. If the message is i, map it to the ith sequence generated in Step 1. Denote this sequence by $\mathbf{y}_1$.
3. If $\mathbf{y}_1 \in T^n_{[Y_1]\delta}$, obtain the sequence

$$\mathbf{u}_{1a} = u_{1a}(\mathbf{y}_1)$$

(recall the notation $f(\mathbf{x})$ in Theorem 6.9). By Theorem 6.9, $\mathbf{u}_{1a} \in T^n_{[U_{1a}]\delta}$. Otherwise, let $\mathbf{u}_{1a}$ be a constant sequence in $T^n_{[U_{1a}]\delta}$.
4. Output the index of $\mathbf{u}_{1a}$ in $T^n_{[U_{1a}]\delta}$ as the codeword and send on the channel $1a$.
5. At the node b, upon receiving the index of $\mathbf{u}_{1a} \in T^n_{[U_{1a}]\delta}$, recover $\mathbf{u}_{1a}$ and obtain

$$\tilde{\mathbf{y}}_1 = y_1(\mathbf{u}_{1a}).$$

If $\tilde{\mathbf{y}}_1 = \mathbf{y}_1$ and $\mathbf{y}_1$ is unique among all the sequences generated in Step 1 of the random coding procedure, then the message i can be decoded correctly.

A decoding error is said to occur if the message i is decoded incorrectly. Note that the total number of codewords is upper bounded by

$$|T^n_{[U_{1a}]\delta}| < 2^{n(H(U_{1a})+\eta)}$$

(cf. (6.5)), so that the rate of the code is at most

$$H(U_{1a}) + \eta < R_{1a} + \eta.$$

We now analyze the probability of decoding error of this random code. Consider

$$\begin{aligned}
&\Pr\{\text{decoding error}\} \\
&= \Pr\{\text{decoding error}|\mathbf{y}_1 \notin T^n_{[Y_1]\delta}\}\Pr\{\mathbf{y}_1 \notin T^n_{[Y_1]\delta}\} \\
&\quad + \Pr\{\text{decoding error}|\mathbf{y}_1 \in T^n_{[Y_1]\delta}\}\Pr\{\mathbf{y}_1 \in T^n_{[Y_1]\delta}\} \\
&\le 1 \cdot \Pr\{\mathbf{y}_1 \notin T^n_{[Y_1]\delta}\} + \Pr\{\text{decoding error}|\mathbf{y}_1 \in T^n_{[Y_1]\delta}\} \cdot 1 \\
&= \Pr\{\mathbf{y}_1 \notin T^n_{[Y_1]\delta}\} + \Pr\{\text{decoding error}|\mathbf{y}_1 \in T^n_{[Y_1]\delta}\}.
\end{aligned}$$

By the strong AEP,

$$\Pr\{\mathbf{y}_1 \notin T^n_{[Y_1]\delta}\} \to 0$$

as $n \to \infty$. So it remains to show that

$$\Pr\{\text{decoding error}|\mathbf{y}_1 \in T^n_{[Y_1]\delta}\} \to 0$$

as $n \to \infty$ with an appropriate choice of δ. Toward this end, we observe that if $\mathbf{y}_1 \in T^n_{[Y_1]\delta}$, then

$$\mathbf{u}_{1a} = u_{1a}(\mathbf{y}_1)$$

(instead of being a constant sequence in $T^n_{[U_{1a}]\delta}$), so that

$$\tilde{\mathbf{y}}_1 = y_1(\mathbf{u}_{1a}) = y_1(u_{1a}(\mathbf{y}_1)).$$

Then from (6.12), we see that

$$\tilde{\mathbf{y}}_1 = \mathbf{y}_1.$$

In other words, if $\mathbf{y}_1 \in T^n_{[Y_1]\delta}$, a decoding error occurs if and only if the sequence $\mathbf{y}_1$ is drawn more than once in Step 1. Thus,

$$\begin{aligned}
&\Pr\{\text{decoding error}|\mathbf{y}_1 \in T^n_{[Y_1]\delta}\} \\
&= \Pr\{\mathbf{y}_1 \text{ drawn more than once}|\mathbf{y}_1 \in T^n_{[Y_1]\delta}\} \\
&= \Pr\left\{\cup_{j\neq i}\{\text{obtain } \mathbf{y}_1 \text{ in the } j\text{th drawing}|\mathbf{y}_1 \in T^n_{[Y_1]\delta}\}\right\}
\end{aligned}$$

$$
\begin{aligned}
&\le \sum_{j\neq i} \Pr\{\text{obtain } \mathbf{y}_1 \text{ in the } j\text{th drawing}|\mathbf{y}_1 \in T^n_{[Y_1]\delta}\} \\
&< 2^{n\omega_1} \cdot \Pr\{\text{obtain } \mathbf{y}_1 \text{ in any drawing}|\mathbf{y}_1 \in T^n_{[Y_1]\delta}\} \\
&< 2^{n\omega_1} \cdot 2^{-n(H(U_{1a})-\eta)} \\
&= 2^{-n(H(U_{1a})-\omega_1-\eta)} \\
&= 2^{-n(H(Y_1)-\omega_1-\eta)},
\end{aligned}
$$

where we have invoked the strong AEP in the last inequality. Since $H(Y_1) > \omega_1$ and $\eta \to 0$ as $\delta \to 0$, by taking δ to be sufficiently small, we have $H(Y_1) - \omega_1 - \eta > 0$, and hence

$$\Pr\{\text{decoding error}|\mathbf{y}_1 \in T^n_{[Y_1]\delta}\} \to 0$$

as $n \to \infty$.

It appears that Theorem 6.11 only complicates the direct part of the source coding theorem, but as we shall see, it actually prepares us to obtain a characterization of the achievable information rate region for more general networks.

6.2.3 Second example

In the next section, we shall state without proof an inner bound on the achievable information rate region $\mathcal{R}$ for a general acyclic network. We already have proved a special case of this inner bound in Theorem 6.11 for a point-to-point communication system. In this section, we prove this inner bound for another network considerably more complicated than the one in the last section. Although this network is still far from being general, the proof of the inner bound for this network contains all the essential ingredients. Besides, the ideas are more transparent without the overwhelming notation in the general proof.

The second network we consider here is the network in Figure 6.2 with the following specification:

$$
\begin{aligned}
&V = \{1,2,a,b,c,d\},\ E = \{1a,2b,ab,ac,bc,bd,cd\} \\
&S = \{1,2\},\ T = \{c,d\},\ \beta(c) = \{1\},\ \beta(d) = \{1,2\}.
\end{aligned}
$$

Call this network G_2.

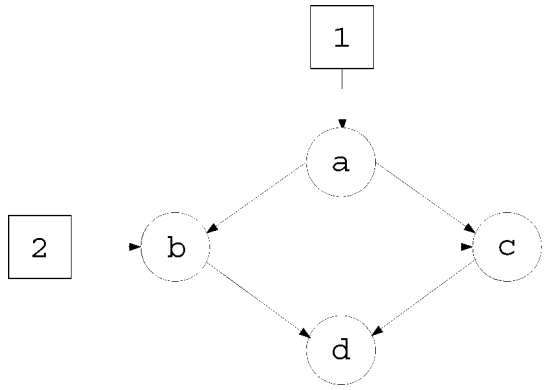

Fig. 6.2 The network G_2 for the second example.

For the network G_2, we first make the observation that the source nodes 1 and 2 each has only one output channel. By the source coding theorem, if either $R_{1a} < \omega_1$ or $R_{2b} < \omega_2$, the sink node d cannot possibly receive both X_1 and X_2. Therefore, in order to make the problem meaningful, we make the assumptions that $R_{1a} \geq \omega_1$ and $R_{2b} \geq \omega_2$, so that we can regard X_1 and X_2 as being directly available to the nodes a and b, respectively.

Theorem 6.12. For the network G_2, an information rate pair (ω_1, ω_2) is achievable if there exist auxiliary random variables $Y_s, s \in S$ and $U_e, e \in E$ such that

$$H(Y_1, Y_2) = H(Y_1) + H(Y_2) \tag{6.13}$$

$$H(Y_s) > \omega_s, \quad s \in S \tag{6.14}$$

$$H(U_{ab}, U_{ac}|Y_1) = 0 \tag{6.15}$$

$$H(U_{bc}, U_{bd}|Y_2, U_{ab}) = 0 \tag{6.16}$$

$$H(U_{cd}|U_{ac}, U_{bc}) = 0 \tag{6.17}$$

$$H(U_e) < R_e, \quad e \in E \tag{6.18}$$

$$H(Y_1|U_{ac}, U_{bc}) = 0 \tag{6.19}$$

$$H(Y_1, Y_2|U_{bd}, U_{cd}) = 0. \tag{6.20}$$

The interpretations of (6.13) to (6.20) are as follows. Similar to our discussion on the network in the last section, Y_s and U_e are random variables representing the information source X_s and the codeword sent on the channel e, respectively. The equality in (6.13) says that the information sources 1 and 2 are independent. The inequality (6.14) is the entropy constraint on the auxiliary random variable Y_s. The equality (6.15) says that the codewords sent on the channels ab and ac depend only on the information source X_1. The equality (6.16) says that the codewords sent on the channels bc and bd depend only on the information source X_2 and the codeword sent on the channel ab. The equality (6.17) says that the codeword sent on the channel cd depends only on the codeword sent on the channels ac and bc. The inequality (6.18) is the capacity constraint for the channel e. The equality (6.19) says that the information source 1 can be recovered (at the sink node c) from the codewords sent on the channels ac and bc, and finally the equality (6.20) says that both the information sources X_1 and X_2 can be recovered (at the sink node d) from the codewords sent on the channels bd and cd.

From (6.15), we see that U_{ab} and U_{ac} are both functions of Y_1. Thus we write

$$U_{ab} = u_{ab}(Y_1) \tag{6.21}$$

and

$$U_{ac} = u_{ac}(Y_1). \tag{6.22}$$

In the same way, from (6.16), (6.17), (6.19), and (6.20), we write

$$U_{bc} = u_{bc}(Y_2, U_{ab}) \tag{6.23}$$

$$U_{bd} = u_{bd}(Y_2, U_{ab}) \tag{6.24}$$

$$U_{cd} = u_{cd}(U_{ac}, U_{bc}) \tag{6.25}$$

$$Y_1 = y_1^{(c)}(U_{ac}, U_{bc}) \tag{6.26}$$

$$Y_1 = y_1^{(d)}(U_{bd}, U_{cd}) \tag{6.27}$$

$$Y_2 = y_2^{(d)}(U_{bd}, U_{cd}). \tag{6.28}$$

In (6.26) to (6.28), the superscript denotes the sink node with which the function is associated.

Proof of Theorem 6.12. Let δ to be a small positive real number to be specified later. For given random variables $Y_s, s \in S$ and $U_e, e \in E$ satisfying (6.13) to (6.20), we construct a random code by the following procedure:

1. For the information source j $(= 1, 2)$,
   a) Generate $2^{n\omega_j}$ sequences of length n independently according to $p^n(y_j)$.
   b) If the message is i_j, map it to the i_j-th sequence generated in Step 1a). Call this sequence $\mathbf{y}_j$.
2. If $\mathbf{y}_1 \in T^n_{[Y_1]}$, obtain the sequences
$$\mathbf{u}_{ab} = u_{ab}(\mathbf{y}_1) \in T^n_{[U_{ab}]\delta}$$
and
$$\mathbf{u}_{ac} = u_{ac}(\mathbf{y}_1) \in T^n_{[U_{ac}]\delta}$$
(cf. (6.21) for the definition of $u_{ab}(\cdot)$, etc, and Theorem 6.9 for the notation $f(\mathbf{x})$). Here, $u_{ab}(\mathbf{y}_1) \in T^n_{[U_{ab}]\delta}$ and $u_{ac}(\mathbf{y}_1) \in T^n_{[U_{ac}]\delta}$ as follow from Theorem 6.8. Otherwise, let $\mathbf{u}_{ab}$ and $\mathbf{u}_{ac}$ be constant sequences in $T^n_{[U_{ab}]\delta}$ and $T^n_{[U_{ac}]\delta}$, respectively.
3. Output the indices of $\mathbf{u}_{ab}$ in $T^n_{[U_{ab}]\delta}$ and $\mathbf{u}_{ac}$ in $T^n_{[U_{ac}]\delta}$ as codewords and send on the channels ab and ac, respectively.
4. If $(\mathbf{y}_2, \mathbf{u}_{ab}) \in T^n_{[Y_2U_{ab}]\delta}$, obtain the sequences
$$\mathbf{u}_{bc} = u_{bc}(\mathbf{y}_2, \mathbf{u}_{ab}) \in T^n_{[U_{bc}]}$$
and
$$\mathbf{u}_{bd} = u_{bd}(\mathbf{y}_2, \mathbf{u}_{ab}) \in T^n_{[U_{bd}]}.$$
Otherwise, let $\mathbf{u}_{bc}$ and $\mathbf{u}_{bd}$ be constant sequences in $T^n_{[U_{bc}]\delta}$ and $T^n_{[U_{bd}]\delta}$, respectively.
5. Output the indices of $\mathbf{u}_{bc}$ in $T^n_{[U_{bc}]\delta}$ and $\mathbf{u}_{bd}$ in $T^n_{[U_{bd}]\delta}$ as codewords and send on the channels bc and bd, respectively.
6. If $(\mathbf{u}_{ac}, \mathbf{u}_{bc}) \in T^n_{[U_{ab}U_{bc}]\delta}$, obtain the sequence
$$\mathbf{u}_{cd} = u_{cd}(\mathbf{u}_{ab}, \mathbf{u}_{bc}) \in T^n_{[U_{cd}]}.$$
Otherwise, let $\mathbf{u}_{cd}$ be a constant sequence in $T^n_{[U_{cd}]\delta}$.

7. Output the index of $\mathbf{u}_{cd}$ in $T^n_{[U_{cd}]\delta}$ as the codeword and send on the channel cd.
8. At the node c, upon receiving the indices of $\mathbf{u}_{ac} \in T^n_{[U_{ac}]\delta}$ and $\mathbf{u}_{bc} \in T^n_{[U_{bc}]\delta}$, $\mathbf{u}_{ac}$ and $\mathbf{u}_{bc}$ can be recovered. Then obtain

$$\tilde{\mathbf{y}}_1^{(c)} = y_1^{(c)}(\mathbf{u}_{ac}, \mathbf{u}_{bc}). \tag{6.29}$$

   If $\tilde{\mathbf{y}}_1^{(c)} = \mathbf{y}_1$ and $\mathbf{y}_1$ is unique among all the sequences generated in Step 1a) for $j = 1$, then the message i_1 can be decoded correctly.
9. At the node d, upon receiving the indices of $\mathbf{u}_{bd} \in T^n_{[U_{bd}]\delta}$ and $\mathbf{u}_{cd} \in T^n_{[U_{cd}]\delta}$, $\mathbf{u}_{bd}$ and $\mathbf{u}_{cd}$ can be recovered. For $j = 1, 2$, obtain

$$\tilde{\mathbf{y}}_j^{(d)} = y_j^{(d)}(\mathbf{u}_{bd}, \mathbf{u}_{cd}).$$

   If $\tilde{\mathbf{y}}_j^{(d)} = \mathbf{y}_j$ and $\mathbf{y}_j$ is unique among all the sequences generated in Step 1a), then the message i_j can be decoded correctly.

If either i_1 is decoded incorrectly at the node c or (i_1, i_2) is decoded incorrectly at the node d, we say that a decoding error occurs. Note that for each channel $e \in E$, the total number of codewords is upper bounded by

$$|T^n_{[U_e]\delta}| < 2^{nH(U_e)+\eta}$$

(cf. (6.5)), so that the rate on the channel e is at most

$$H(U_e) + \eta < R_e + \eta.$$

We now analyze the probability of decoding error of this random code. Analogous to the proof of Theorem 6.11 in the last section, we have

$$\begin{aligned}&\Pr\{\text{decoding error}\}\\ &\le \Pr\{(\mathbf{y}_1, \mathbf{y}_2) \notin T^n_{[Y_1Y_2]\delta}\} + \Pr\{\text{decoding error}|(\mathbf{y}_1, \mathbf{y}_2) \in T^n_{[Y_1]\delta}\}.\end{aligned}$$

Since the pair of sequence $(\mathbf{y}_1, \mathbf{y}_2)$ is generated according to

$$p^n(y_1)p^n(y_2) = p^n(y_1, y_2),$$

by the strong JAEP,

$$\Pr\{(\mathbf{y}_1, \mathbf{y}_2) \notin T^n_{[Y_1Y_2]\delta}\} \to 0$$

as $n \to \infty$, so it suffices to show that

$$\Pr\{\text{decoding error}|(\mathbf{y}_1, \mathbf{y}_2) \in T^n_{[Y_1Y_2]\delta}\} \to 0$$

as $n \to \infty$ with an appropriate choice of δ. Toward this end, we analyze the random coding procedure when $(\mathbf{y}_1, \mathbf{y}_2) \in T^n_{[Y_1Y_2]\delta}$:

- By Theorem 6.8, we have $\mathbf{y}_j \in T^n_{[Y_j]\delta}$, $j = 1, 2$.
- In Step 2, since $\mathbf{y}_1 \in T^n_{[Y_1]\delta}$, we have

  $$\mathbf{u}_{ab} = u_{ab}(\mathbf{y}_1) \tag{6.30}$$

  (instead of a constant sequence in $T^n_{[U_{ab}]\delta}$) and

  $$\mathbf{u}_{ac} = u_{ac}(\mathbf{y}_1). \tag{6.31}$$

- In Step 4, by (6.30), we have

  $$(\mathbf{y}_2, \mathbf{u}_{ab}) = (\mathbf{y}_2, u_{ab}(\mathbf{y}_1)).$$

  Since $(\mathbf{y}_1, \mathbf{y}_2) \in T^n_{[Y_1Y_2]\delta}$,

  $$(\mathbf{y}_2, u_{ab}(\mathbf{y}_1)) \in T^n_{[Y_2U_{ab}]\delta}$$

  by Theorem 6.9. Therefore,

  $$\mathbf{u}_{bc} = u_{bc}(\mathbf{y}_2, \mathbf{u}_{ab}) \tag{6.32}$$

  and

  $$\mathbf{u}_{bd} = u_{bd}(\mathbf{y}_2, \mathbf{u}_{ab}). \tag{6.33}$$

- In Step 6, by applying (6.31), (6.32) and (6.30), we have

$$\begin{aligned}(\mathbf{u}_{ac}, \mathbf{u}_{bc}) &= (u_{ac}(\mathbf{y}_1), u_{bc}(\mathbf{y}_2, \mathbf{u}_{ab})) \\ &= (u_{ac}(\mathbf{y}_1), u_{bc}(\mathbf{y}_2, u_{ab}(\mathbf{y}_1))). \end{aligned} \tag{6.34}$$

  Again, since $(\mathbf{y}_1, \mathbf{y}_2) \in T^n_{[Y_1Y_2]\delta}$,

$$(\mathbf{u}_{ac}, \mathbf{u}_{bc}) \in T^n_{[U_{ac}U_{bc}]\delta}$$

  by Theorem 6.9. Therefore,

$$\mathbf{u}_{cd} = u_{cd}(\mathbf{u}_{ac}, \mathbf{u}_{bc}).$$

- By (6.26), (6.22), (6.23), and (6.21), we can write

$$\begin{aligned} Y_1 &= y_1^{(c)}(U_{ac}, U_{bc}) \\ &= y_1^{(c)}(u_{ac}(Y_1), u_{bc}(Y_2, U_{ab})) \\ &= y_1^{(c)}(u_{ac}(Y_1), u_{bc}(Y_2, u_{ab}(Y_1))). \end{aligned} \tag{6.35}$$

  On the other hand, from (6.29) and (6.34), we have

$$\begin{aligned} \tilde{\mathbf{y}}_1^{(c)} &= y_1^{(c)}(\mathbf{u}_{ac}, \mathbf{u}_{bc}) \\ &= y_1^{(c)}(u_{ac}(\mathbf{y}_1), u_{bc}(\mathbf{y}_2, u_{ab}(\mathbf{y}_1))). \end{aligned} \tag{6.36}$$

  A comparison of (6.35) and (6.36) reveals that

$$\tilde{\mathbf{y}}_1^{(c)} = \mathbf{y}_1. \tag{6.37}$$

  Similarly, it can be shown that

$$\tilde{\mathbf{y}}_1^{(d)} = \mathbf{y}_1. \tag{6.38}$$

  and

$$\tilde{\mathbf{y}}_2^{(d)} = \mathbf{y}_2. \tag{6.39}$$

In conclusion, whenever $(\mathbf{y}_1, \mathbf{y}_2) \in T^n_{[Y_1Y_2]\delta}$, (6.37) to (6.39) hold. By the strong AEP,

$$\Pr\{(\mathbf{y}_1, \mathbf{y}_2) \in T^n_{[Y_1Y_2]\delta}\} \to 1$$

as $n \to \infty$. Therefore, if $(\mathbf{y}_1, \mathbf{y}_2) \in T^n_{[Y_1Y_2]\delta}$, a decoding error occurs if and only if either $\mathbf{y}_1$ or $\mathbf{y}_2$ is drawn more than once in Step 1a).

By means of an argument similar to the one in the proof of Theorem 6.11, it can be shown that

$$\Pr\{\text{decoding error}|(\mathbf{y}_1,\mathbf{y}_2) \in T^n_{[Y_1Y_2]\delta}\} \to 0$$

as $n \to \infty$ with an appropriate choice of δ. The details are omitted here.

6.2.4 General acyclic networks

In this section, we present an inner bound $\mathcal{R}_{in}$ on the information rate region for a general acyclic network. The reader should have no problem understanding the meaning of $\mathcal{R}_{in}$ after studying the special cases in the previous two sections. In the sequel, we will use the abbreviations Y_S, $U_{\mathrm{In}(i)}$ respectively for $\{Y_s : s \in S\}$, $\{U_e : e \in \mathrm{In}(i)\}$, etc.

Definition 6.13. Let $\mathcal{R}'$ be the set of all information rate tuples $\boldsymbol{\omega}$ such that there exist auxiliary random variables $Y_s, s \in S$ and $U_e, e \in E$ which satisfy the following conditions:

$$H(Y_S) = \sum_{s\in S} H(Y_s) \tag{6.40}$$

$$H(Y_s) > \omega_s, \quad s \in S \tag{6.41}$$

$$H(U_{\mathrm{Out}(s)}|Y_s) = 0, \quad s \in S \tag{6.42}$$

$$H(U_{\mathrm{Out}(i)}|U_{\mathrm{In}(i)}) = 0, \quad i \in V\backslash S \tag{6.43}$$

$$H(U_e) < R_e, \quad e \in E \tag{6.44}$$

$$H(Y_{\beta(i)}|U_{\mathrm{In}(i)}) = 0, \quad i \in T. \tag{6.45}$$

Theorem 6.14. $\mathcal{R}' \subset \mathcal{R}$.

The proof of Theorem 6.14 involves a set of techniques originally developed in [211] and [203]. The proof of Theorem 6.12 in the last section, though a special case of Theorem 6.16 here, contains all the essential ingredients necessary for proving Theorem 6.14.

Definition 6.15. Let $\mathcal{R}_{in} = \overline{\mathrm{con}}(\mathcal{R}')$, the convex closure of $\mathcal{R}'$.

Theorem 6.16. $\mathcal{R}_{in} \subset \mathcal{R}$.

Theorem 6.16 can readily be obtained from Theorem 6.14 as a corollary by invoking the remark following Definition 6.3. Specifically, by taking the convex closure on both sides in

$$\mathcal{R}' \subset \mathcal{R},$$

we have

$$\overline{\text{con}}(\mathcal{R}') \subset \overline{\text{con}}(\mathcal{R}) = \mathcal{R}.$$

For a complete proof of Theorem 6.16, we refer the reader to [203] and [209], Ch. 15[2]. The inner bound proved in [203] is for zero-error variable-length network codes.

6.2.5 $\mathcal{R}_{in}$ recasted

In this section, $\mathcal{R}_{in}$ will be recasted in the framework of information inequalities developed in [208]. As we shall see, this alternative characterization of $\mathcal{R}_{in}$, developed in [211] and [203], enables the region to be described on the same footing for different multi-source network coding problems.

Let $\mathcal{N}$ be a collection of discrete random variables whose joint distribution is unspecified, and let

$$\mathcal{Q}_{\mathcal{N}} = 2^{\mathcal{N}} \backslash \{\emptyset\},$$

the set of all nonempty subsets of random variables in $\mathcal{N}$. Then

$$|\mathcal{Q}_{\mathcal{N}}| = 2^{|\mathcal{N}|} - 1.$$

Let $\mathcal{H}_{\mathcal{N}}$ be the $|\mathcal{Q}_{\mathcal{N}}|$-dimensional Euclidean space with the coordinates labeled by $h_A, A \in \mathcal{Q}_{\mathcal{N}}$. We will refer to $\mathcal{H}_{\mathcal{N}}$ as the entropy space for the set of random variables $\mathcal{N}$. A vector

$$\mathbf{h} = (h_A : A \in \mathcal{Q}_{\mathcal{N}}) \in \mathcal{H}_{\mathcal{N}} \tag{6.46}$$

[2] The proof given in Section 6.2.3 is a simplified version of the proofs in [209] and [203].

is said to be an *entropy function* if there exists a joint distribution for $(Z : Z \in \mathcal{N})$ such that

$$h_A = H(Z : Z \in A)$$

for all $A \in \mathcal{Q}_{\mathcal{N}}$. We then define the region

$$\Gamma^*_{\mathcal{N}} = \{\mathbf{h} \in \mathcal{H}_{\mathcal{N}} : \mathbf{h} \text{ is an entropy function}\}.$$

To simplify notation in the sequel, for any nonempty $A, A' \in \mathcal{Q}_{\mathcal{N}}$, we define

$$h_{A|A'} = h_{AA'} - h_{A'}, \tag{6.47}$$

where we use juxtaposition to denote the union of two sets. In using the above notation, we do not distinguish elements and singletons of $\mathcal{N}$, i.e., for a random variable $Z \in \mathcal{N}$, h_Z is the same as $h_{\{Z\}}$. Note that (6.47) corresponds to the information-theoretic identity

$$H(A|A') = H(AA') - H(A').$$

To describe $\mathcal{R}_{in}$ in terms of the above framework, we let

$$\mathcal{N} = \{Y_s : s \in S; U_e : e \in E\}.$$

Observe that the constraints (6.40) to (6.45) in the definition of $\mathcal{R}'$ correspond to the following constraints in $\mathcal{H}_{\mathcal{N}}$, respectively:

$$h_{Y_S} = \sum_{s \in S} h_{Y_s} \tag{6.48}$$

$$h_{Y_s} > \omega_s, \quad s \in S \tag{6.49}$$

$$h_{U_{\mathrm{Out}(s)}|Y_s} = 0, \quad s \in S \tag{6.50}$$

$$h_{U_{\mathrm{Out}(i)}|U_{\mathrm{In}(i)}} = 0, \quad i \in V \backslash S \tag{6.51}$$

$$h_{U_e} < R_e, \quad e \in E \tag{6.52}$$

$$h_{Y_{\beta(i)}|U_{\mathrm{In}(i)}} = 0, \quad i \in T. \tag{6.53}$$

Then we have the following alternative definition of $\mathcal{R}'$.

Definition 6.17. Let $\mathcal{R}'$ be the set of all information rate tuples $\boldsymbol{\omega}$ such that there exists $\mathbf{h} \in \Gamma^*_{\mathcal{N}}$ which satisfies (6.48) to (6.53).

Although the original definition of $\mathcal{R}'$ as given in Definition 6.13 is more intuitive, the region so defined appears to be totally different from one problem to another problem. On the other hand, the alternative definition of $\mathcal{R}'$ above enables the region to be described on the same footing for all cases. Moreover, if $\tilde{\Gamma}_{\mathcal{N}}$ is an explicit inner bound on $\Gamma^*_{\mathcal{N}}$, upon replacing $\Gamma^*_{\mathcal{N}}$ by $\tilde{\Gamma}_{\mathcal{N}}$ in the above definition of $\mathcal{R}'$, we immediately obtain an explicit inner bound on $\mathcal{R}_{in}$ for all cases. We shall see further advantage of this alternative definition when we discuss an explicit outer bound on $\mathcal{R}$ in the next section.

6.3 Outer bound $\mathcal{R}_{out}$

In this section, we prove an outer bound $\mathcal{R}_{out}$ on $\mathcal{R}$. This outer bound is in terms of $\overline{\Gamma}^*_{\mathcal{N}}$, the closure of $\Gamma^*_{\mathcal{N}}$.

Definition 6.18. Let $\mathcal{R}_{out}$ be the set of all information rate tuples $\boldsymbol{\omega}$ such that there exists $\mathbf{h} \in \overline{\Gamma}^*_{\mathcal{N}}$ which satisfies the following constraints:

$$h_{Y_S} = \sum_{s \in S} h_{Y_s} \tag{6.54}$$

$$h_{Y_s} \geq \omega_s, \quad s \in S \tag{6.55}$$

$$h_{U_{\mathrm{Out}(s)}|Y_s} = 0, \quad s \in S \tag{6.56}$$

$$h_{U_{\mathrm{Out}(i)}|U_{\mathrm{In}(i)}} = 0, \quad i \in V \backslash S \tag{6.57}$$

$$h_{U_e} \leq R_e, \quad e \in E \tag{6.58}$$

$$h_{Y_{\beta(i)}|U_{\mathrm{In}(i)}} = 0, \quad i \in T. \tag{6.59}$$

The definition of $\mathcal{R}_{out}$ is the same as the alternative definition of $\mathcal{R}'$ (Definition 6.17) except that

1. $\Gamma^*_{\mathcal{N}}$ is replaced by $\overline{\Gamma}^*_{\mathcal{N}}$.
2. The inequalities in (6.49) and (6.52) are strict, while the inequalities in (6.55) and (6.58) are nonstrict.

From the definitions of $\mathcal{R}'$ and $\mathcal{R}_{out}$, it is clear that

$$\mathcal{R}' \subset \mathcal{R}_{out}. \tag{6.60}$$

It is also easy to verify that the convexity of $\overline{\Gamma}^*_{\mathcal{N}}$ ([209], Theorem 14.5) implies the convexity of $\mathcal{R}_{out}$. Then upon taking convex closure in (6.60), we see that

$$\mathcal{R}_{in} = \overline{\mathrm{con}}(\mathcal{R}') \subset \overline{\mathrm{con}}(\mathcal{R}_{out}) = \mathcal{R}_{out},$$

where the last equality follows because $\mathcal{R}_{out}$ is close and convex. However, it is not apparent that the two regions $\mathcal{R}_{in}$ and $\mathcal{R}_{out}$ coincide in general. This will be further discussed in the next section. We first prove that $\mathcal{R}_{out}$ is indeed an outer bound on $\mathcal{R}$.

Theorem 6.19. $\mathcal{R} \subset \mathcal{R}_{out}$.

Proof. Let $\boldsymbol{\omega}$ be an achievable information rate tuple and n be a sufficiently large integer. Then for any $\epsilon > 0$, there exists an

$$(n, (\eta_e : e \in E), (\tau_s : s \in S))$$

code on the network such that

$$n^{-1} \log \eta_e \le R_e + \epsilon \tag{6.61}$$

for all $e \in E$,

$$\tau_s \ge \omega_s - \epsilon \tag{6.62}$$

for all $s \in S$, and

$$\Delta_i \le \epsilon \tag{6.63}$$

for all $i \in T$.

We consider such a code for a fixed ϵ and a sufficiently large n. Since the information sources $X_s, s \in S$ are mutually independent, we have

$$H(X_S) = \sum_{s \in S} H(X_s). \tag{6.64}$$

For all $s \in S$, from (6.62),

$$H(X_s) = \log |\mathcal{X}_s| = \log \lceil 2^{n\tau_s} \rceil \ge n\tau_s \ge n(\omega_s - \epsilon). \tag{6.65}$$

For $e \in E$, let U_e be the codeword sent on the channel e. For all $s \in S$ and $e \in \mathrm{Out}(s)$, since U_e is a function of the information source X_s,

$$H(U_{\mathrm{Out}(s)}|X_s) = 0. \tag{6.66}$$

Similarly, for all $i \in V \backslash S$,

$$H(U_{\mathrm{Out}(i)}|U_{\mathrm{In}(i)}) = 0. \tag{6.67}$$

From (6.1), (6.2), and (6.61), for all $e \in E$,

$$H(U_e) \leq \log|U_e| = \log(\eta_e + 1) \leq n(R_e + 2\epsilon). \tag{6.68}$$

For $i \in T$, by Fano's inequality (cf. [209], Corollary 2.48), we have

$$\begin{aligned} H(X_{\beta(i)}|U_{\mathrm{In}(i)}) &\leq 1 + \Delta_i \log\left(\prod_{s\in\beta(i)} |\mathcal{X}_s|\right) \\ &= 1 + \Delta_i H(X_{\beta(i)}) \qquad (6.69) \\ &\leq 1 + \epsilon H(X_{\beta(i)}), \qquad (6.70) \end{aligned}$$

where (6.69) follows because X_s distributes uniformly on $\mathcal{X}_s$ and X_s, $s \in S$ are mutually independent, and (6.70) follows from (6.63). Then

$$\begin{aligned} H(X_{\beta(i)}) &= I(X_{\beta(i)}; U_{\mathrm{In}(i)}) + H(X_{\beta(i)}|U_{\mathrm{In}(i)}) \\ &\overset{a)}{\leq} I(X_{\beta(i)}; U_{\mathrm{In}(i)}) + 1 + \epsilon H(X_{\beta(i)}) \\ &\leq H(U_{\mathrm{In}(i)}) + 1 + \epsilon H(X_{\beta(i)}) \\ &\overset{b)}{\leq} \left(\sum_{e\in\mathrm{In}(i)} \log\eta_e\right) + 1 + \epsilon H(X_{\beta(i)}) \\ &\overset{c)}{\leq} \left(\sum_{e\in\mathrm{In}(i)} n(R_e + \epsilon)\right) + 1 + \epsilon H(X_{\beta(i)}), \qquad (6.71) \end{aligned}$$

where

a) follows from (6.70);
b) follows from $H(Z) \leq \log|\mathcal{Z}|$, cf. [209], Theorem 2.43;
c) follows from (6.61).

Rearranging the terms in (6.71), we obtain

$$\begin{aligned} H(X_{\beta(i)}) &\le \frac{n}{1-\epsilon}\left(\sum_{e\in \mathrm{In}(i)}(R_e+\epsilon)+\frac{1}{n}\right) \\ &< 2n\sum_{e\in \mathrm{In}(i)}(R_e+\epsilon) \end{aligned} \tag{6.72}$$

for sufficiently small ϵ and sufficiently large n. Substituting (6.72) into (6.70), we have

$$\begin{aligned} H(X_{\beta(i)}|U_{\mathrm{In}(i)}) &< n\left(\frac{1}{n}+2\epsilon\sum_{e\in \mathrm{In}(i)}(R_e+\epsilon)\right) \\ &= n\phi_i(n,\epsilon), \end{aligned} \tag{6.73}$$

where

$$\phi_i(n,\epsilon)=\left(\frac{1}{n}+2\epsilon\sum_{e\in \mathrm{In}(i)}(R_e+\epsilon)\right)\to 0$$

as $n\to\infty$ and then $\epsilon\to 0$. Thus for this code, from (6.64), (6.65), (6.67), (6.68), and (6.73), we have

$$H(X_S)=\sum_{s\in S}H(X_s) \tag{6.74}$$

$$H(X_s)\ge n(\omega_s-\epsilon),\quad s\in S \tag{6.75}$$

$$H(U_{\mathrm{Out}(s)}|X_s)=0,\quad s\in S \tag{6.76}$$

$$H(U_{\mathrm{Out}(i)}|U_{\mathrm{In}(i)})=0,\quad i\in V\backslash S \tag{6.77}$$

$$H(U_e)\le n(R_e+2\epsilon),\quad e\in E \tag{6.78}$$

$$H(X_{\beta(i)}|U_{\mathrm{In}(i)})\le n\phi_i(n,\epsilon),\quad i\in T. \tag{6.79}$$

We note the one-to-one correspondence between (6.74) to (6.79) and (6.54) to (6.59). By letting $Y_s=X_s$ for all $s\in S$, we see that there exists $\mathbf{h}\in\Gamma^*_{\mathcal{N}}$ such that

$$h_{Y_S}=\sum_{s\in S}h_{Y_s} \tag{6.80}$$

$$h_{Y_s}\ge n(\omega_s-\epsilon),\quad s\in S \tag{6.81}$$

$$h_{U_{\mathrm{Out}(s)}|Y_s} = 0, \quad s \in S \tag{6.82}$$

$$h_{U_{\mathrm{Out}(i)}|U_{\mathrm{In}(i)}} = 0, \quad i \in V \backslash S \tag{6.83}$$

$$h_{U_e} \leq n(R_e + 2\epsilon), \quad e \in E \tag{6.84}$$

$$h_{Y_{\beta(i)}|U_{\mathrm{In}(i)}} \leq n\phi_i(n,\epsilon), \quad i \in T. \tag{6.85}$$

By Theorem 14.5 in [209], $\overline{\Gamma}^*_{\mathcal{N}}$ is a convex cone. Therefore, if $\mathbf{h} \in \Gamma^*_{\mathcal{N}}$, then $n^{-1}\mathbf{h} \in \overline{\Gamma}^*_{\mathcal{N}}$. Dividing (6.80) through (6.85) by n and replacing $n^{-1}\mathbf{h}$ by $\mathbf{h}$, we see that there exists $\mathbf{h} \in \overline{\Gamma}^*_{\mathcal{N}}$ such that

$$\begin{aligned}
h_{Y_S} &= \sum_{s \in S} h_{Y_s} \\
h_{Y_s} &\geq \omega_s - \epsilon, \quad s \in S \\
h_{U_{\mathrm{Out}(s)}|Y_s} &= 0, \quad s \in S \\
h_{U_{\mathrm{Out}(i)}|U_{\mathrm{In}(i)}} &= 0, \quad i \in V \backslash S \\
h_{U_e} &\leq R_e + 2\epsilon, \quad e \in E \\
h_{Y_{\beta(i)}|U_{\mathrm{In}(i)}} &\leq \phi_i(n,\epsilon), \quad i \in T.
\end{aligned}$$

We then let $n \to \infty$ and then $\epsilon \to 0$ to conclude that there exists $\mathbf{h} \in \overline{\Gamma}^*_{\mathcal{N}}$ which satisfies (6.54) to (6.59). Hence, $\mathcal{R} \subset \mathcal{R}_{out}$, and the theorem is proved. □

6.4 $\mathcal{R}_{LP}$ – An explicit outer bound

In Section 6.2.5, we stated the inner bound $\mathcal{R}_{in}$ on $\mathcal{R}$ in terms of $\Gamma^*_{\mathcal{N}}$, and in Section 6.3, we proved the outer bound $\mathcal{R}_{out}$ on $\mathcal{R}$ in terms of $\overline{\Gamma}^*_{\mathcal{N}}$. So far, there exists no full characterization of either $\Gamma^*_{\mathcal{N}}$ or $\overline{\Gamma}^*_{\mathcal{N}}$. Therefore, these bounds cannot be evaluated explicitly. In this section, we give a geometrical interpretation of these bounds which leads to an explicit outer bound on $\mathcal{R}$ called the LP bound (LP for *linear programming*).

Let A be a subset of $\mathcal{Q}_{\mathcal{N}}$. For a vector $\mathbf{h} \in \mathcal{H}_{\mathcal{N}}$, let

$$\mathbf{h}_A = (h_Z : Z \in A).$$

For a subset $\mathcal{B}$ of $\mathcal{H}_{\mathcal{N}}$, let

$$\mathrm{proj}_A(\mathcal{B}) = \{\mathbf{h}_A : \mathbf{h} \in \mathcal{B}\}$$

be the projection of the set $\mathcal{B}$ on the coordinates $h_Z, Z \in A$. For a subset $\mathcal{B}$ of $\mathcal{H}_{\mathcal{N}}$, define

$$\Lambda(\mathcal{B}) = \{\mathbf{h} \in \mathcal{H}_{\mathcal{N}} : 0 \leq \mathbf{h} < \mathbf{h}' \text{ for some } \mathbf{h}' \in \mathcal{B}\}$$

and

$$\bar{\Lambda}(\mathcal{B}) = \{\mathbf{h} \in \mathcal{H}_{\mathcal{N}} : 0 \leq \mathbf{h} \leq \mathbf{h}' \text{ for some } \mathbf{h}' \in \mathcal{B}\}.$$

A vector $\mathbf{h} \geq 0$ is in $\Lambda(\mathcal{B})$ if and only if it is *strictly* inferior to some vector $\mathbf{h}'$ in $\mathcal{B}$, and is in $\bar{\Lambda}(\mathcal{B})$ if and only if it is inferior to some vector $\mathbf{h}'$ in $\mathcal{B}$.

Define the following subsets of $\mathcal{H}_{\mathcal{N}}$:

$$\begin{aligned}
\mathcal{C}_1 &= \left\{\mathbf{h} \in \mathcal{H}_{\mathcal{N}} : h_{Y_S} = \sum_{s \in S} h_{Y_s}\right\} \\
\mathcal{C}_2 &= \left\{\mathbf{h} \in \mathcal{H}_{\mathcal{N}} : h_{U_{\mathrm{Out}(s)}|Y_s} = 0 \text{ for all } s \in S\right\} \\
\mathcal{C}_3 &= \left\{\mathbf{h} \in \mathcal{H}_{\mathcal{N}} : h_{U_{\mathrm{Out}(i)}|U_{\mathrm{In}(i)}} = 0 \text{ for all } i \in V \backslash S\right\} \\
\mathcal{C}_4 &= \{\mathbf{h} \in \mathcal{H}_{\mathcal{N}} : h_{U_e} < R_e \text{ for all } e \in E\} \\
\mathcal{C}_5 &= \left\{\mathbf{h} \in \mathcal{H}_{\mathcal{N}} : h_{Y_{\beta(i)}|U_{\mathrm{In}(i)}} = 0 \text{ for all } i \in T\right\}.
\end{aligned}$$

These sets contain points in $\mathcal{H}_{\mathcal{N}}$ that satisfy the constraints in (6.48) and (6.50) to (6.53), respectively. The set $\mathcal{C}_1$ is a hyperplane in $\mathcal{H}_{\mathcal{N}}$. Each of the sets $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_5$ is the intersection of a collection of hyperplanes in $\mathcal{H}_{\mathcal{N}}$. The set $\mathcal{C}_4$ is the intersection of a collection of open half-spaces in $\mathcal{H}_{\mathcal{N}}$. Then from the alternative definition of $\mathcal{R}'$ (Definition 6.17), we see that

$$\mathcal{R}' = \Lambda(\mathrm{proj}_{Y_S}(\Gamma^*_{\mathcal{N}} \cap \mathcal{C}_1 \cap \mathcal{C}_2 \cap \mathcal{C}_3 \cap \mathcal{C}_4 \cap \mathcal{C}_5)).$$

and

$$\mathcal{R}_{in} = \overline{\mathrm{con}}(\Lambda(\mathrm{proj}_{Y_S}(\Gamma^*_{\mathcal{N}} \cap \mathcal{C}_1 \cap \mathcal{C}_2 \cap \mathcal{C}_3 \cap \mathcal{C}_4 \cap \mathcal{C}_5))).$$

Similarly, we see that

$$\mathcal{R}_{out} = \bar{\Lambda}(\mathrm{proj}_{Y_S}(\overline{\Gamma}^*_{\mathcal{N}} \cap \mathcal{C}_1 \cap \mathcal{C}_2 \cap \mathcal{C}_3 \cap \mathcal{C}_4 \cap \overline{\mathcal{C}_5})). \tag{6.86}$$

It can be shown that if $\Gamma^*_{\mathcal{N}} \cap (\mathcal{C}_1 \cap \mathcal{C}_2 \cap \mathcal{C}_3 \cap \mathcal{C}_5)$ is dense in $\overline{\Gamma}^*_{\mathcal{N}} \cap (\mathcal{C}_1 \cap \mathcal{C}_2 \cap \mathcal{C}_3 \cap \mathcal{C}_5)$, i.e.,

$$\overline{\Gamma^*_{\mathcal{N}} \cap (\mathcal{C}_1 \cap \mathcal{C}_2 \cap \mathcal{C}_3 \cap \mathcal{C}_5)} = \overline{\Gamma}^*_{\mathcal{N}} \cap (\mathcal{C}_1 \cap \mathcal{C}_2 \cap \mathcal{C}_3 \cap \mathcal{C}_5),$$

then

$$\mathcal{R}_{out} = \overline{\mathcal{R}'} \subset \overline{\text{con}}(\mathcal{R}') = \mathcal{R}_{in},$$

which implies

$$\mathcal{R}_{in} = \mathcal{R}_{out}.$$

Note that $(\mathcal{C}_1 \cap \mathcal{C}_2 \cap \mathcal{C}_3 \cap \mathcal{C}_5)$ is a closed subset of $\mathcal{H}_{\mathcal{N}}$. However, while

$$\overline{\Gamma^*_{\mathcal{N}} \cap \mathcal{C}} \subset \overline{\Gamma}^*_{\mathcal{N}} \cap \mathcal{C}$$

for any closed subset $\mathcal{C}$ of $\mathcal{H}_{\mathcal{N}}$, it is not in general true that

$$\overline{\Gamma^*_{\mathcal{N}} \cap \mathcal{C}} = \overline{\Gamma}^*_{\mathcal{N}} \cap \mathcal{C}.$$

As a counterexample, it has been shown in [214] (also see [209], Theorem 14.2) that $\overline{\Gamma^*_3 \cap \tilde{\mathcal{C}}}$ is a proper subset of $\overline{\Gamma}^*_3 \cap \tilde{\mathcal{C}}$, where Γ^*_n denotes $\Gamma^*_{\mathcal{N}}$ for

$$\mathcal{N} = \{X_1, X_2, \cdots, X_n\}$$

and

$$\tilde{\mathcal{C}} = \left\{\mathbf{h} \in \Gamma^*_3 : h_{X_j} + h_{X_k} = h_{\{X_j, X_k\}}, 1 \le j < k \le 3\right\}.$$

To facilitate our discussion, we further define

$$i_{A;A'} = h_A - h_{A|A'} \tag{6.87}$$

and

$$i_{A;A'|A''} = h_{A|A''} - h_{A|A'A''} \tag{6.88}$$

for $A, A', A'' \in \mathcal{Q}_{\mathcal{N}}$. Note that (6.87) and (6.88) correspond to the information-theoretic identities

$$I(A; A') = H(A) - H(A|A')$$

and

$$I(A; A'|A'') = H(A|A'') - H(A|A'A''),$$

respectively. Let $\Gamma_{\mathcal{N}}$ be the set of $\mathbf{h} \in \mathcal{H}_{\mathcal{N}}$ such that $\mathbf{h}$ satisfies all the *basic inequalities* involving some or all of the random variables in $\mathcal{N}$, i.e., for all $A, A', A'' \in \mathcal{Q}_{\mathcal{N}}$,

$$\begin{aligned} h_A &\geq 0 \\ h_{A|A'} &\geq 0 \\ i_{A;A'} &\geq 0 \\ i_{A;A'|A''} &\geq 0. \end{aligned}$$

These inequalities are equivalent to the nonnegativity of all Shannon's information measures (entropy, conditional entropy, mutual information, and conditional mutual information). The significance of the region $\Gamma_{\mathcal{N}}$ is that it fully characterizes all the *Shannon-type information inequalities* involving the random variables in $\mathcal{N}$, namely those inequalities implied by the above set of basic inequalities. Since the basic inequalities are satisfied by all joint distributions (i.e., $\mathbf{h} \in \Gamma^*_{\mathcal{N}}$ implies $\mathbf{h} \in \Gamma_{\mathcal{N}}$) and that $\Gamma_{\mathcal{N}}$ is closed, we have $\overline{\Gamma}^*_{\mathcal{N}} \subset \Gamma_{\mathcal{N}}$. Then upon replacing $\overline{\Gamma}^*_{\mathcal{N}}$ by $\Gamma_{\mathcal{N}}$ in the definition of $\mathcal{R}_{out}$, we immediately obtain an outer bound on $\mathcal{R}_{out}$. This is called the *LP bound*, denoted by $\mathcal{R}_{LP}$. In other words, $\mathcal{R}_{LP}$ is obtained by replacing $\overline{\Gamma}^*_{\mathcal{N}}$ by $\Gamma_{\mathcal{N}}$ on the right hand side of (6.86), i.e.,

$$\mathcal{R}_{LP} = \bar{\Lambda}(\mathrm{proj}_{Y_S}(\Gamma_{\mathcal{N}} \cap \mathcal{C}_1 \cap \mathcal{C}_2 \cap \mathcal{C}_3 \cap \mathcal{C}_4 \cap \overline{\mathcal{C}_5})).$$

Since all the constraints defining $\mathcal{R}_{LP}$ are linear, $\mathcal{R}_{LP}$ can in principle be evaluated explicitly, although the computation involved can be nontrivial.

However, it has been shown in [215] by means of the discovery of what is known as a *non-Shannon-type information inequality* that $\overline{\Gamma}^*_n \neq \Gamma_n$ for $n \geq 4$, so there is a potential gap between $\mathcal{R}_{out}$ and $\mathcal{R}_{LP}$. In short, a non-Shannon-type information inequality is an outer bound on $\Gamma^*_{\mathcal{N}}$ which is not implied by the basic inequalities. Specifically, it is

proved in [215] that for any 4 random variables X_1, X_2, X_3, and X_4,

$$\begin{aligned} 2I(X_3;X_4) \le\ & I(X_1;X_2) + I(X_1;X_3,X_4) \\ & + 3I(X_3;X_4|X_1) + I(X_3;X_4|X_2). \end{aligned} \tag{6.89}$$

We refer the reader to [209], Ch. 14, for a detailed discussion.

Now return to the question of whether there is indeed a gap between $\mathcal{R}_{out}$ and $\mathcal{R}_{LP}$. This important question has recently been answered in [167], where it is shown by means of the non-Shannon-type inequality (6.89) that $\mathcal{R}_{LP}$ is not tight for a particular multi-source network coding problem constructed from matroid theory. This result implies that $\mathcal{R}_{out}$ is generally tighter than $\mathcal{R}_{LP}$.

Nonetheless, it has been proved in [209], Ch. 15, and [211] that $\mathcal{R}_{LP}$ is tight for all special cases of multi-source network coding for which the achievable information rate region is known. These include single-source network coding discussed in Part I as well as the models described in [207][177][200][212][211]. Since $\mathcal{R}_{LP}$ encompasses all Shannon-type information inequalities and the converse proofs of the achievable information rate region for all these special cases do not involve non-Shannon-type inequalities, the tightness of $\mathcal{R}_{LP}$ for all these cases is not surprising.

7

Fundamental Limits of Linear Codes

In Part I, we have shown that for single-source network coding, linear codes are sufficient for achieving asymptotic optimality. It is not clear whether this continues to hold for multi-source network coding. In this section, we present a framework for discussion and explore a potential gap between the asymptotic performance of linear codes and nonlinear codes.

7.1 Linear network codes for multiple sources

We first generalize the global description of a linear network code in Definition 2.5 of Part I for multiple sources. As in Part I, to facilitate our discussion of linear codes, we assume that each channel has unit capacity. Let F be a finite field,

$$\boldsymbol{\omega} = (\omega_s : s \in S)$$

be a tuple of positive integers, and

$$\Omega = \sum_{s \in S} \omega_s.$$

Consider the space F^{Ω}. The information source generated at a source node s is regarded as an ω_s-dimensional subspace of F^{Ω}, denoted by W_s, and it is assumed that the subspaces for different information sources are linearly independent, i.e.,

$$W_s \cap W_{s'} = 0 \quad \text{for } s \neq s', \tag{7.1}$$

where 0 denotes the zero vector.

As in Part I, the information source generated at a source node s is modelled by ω_s imaginary channels terminating at the node s. We adopt the convention that these channels are labeled by $s(1), s(2), \cdots, s(\omega_s)$.

Definition 7.1. (Global Description of a Linear Network Code) Let F be a finite field, and $\boldsymbol{\omega} = (\omega_s : s \in S)$ be a tuple of positive integers. For $s \in S$, let W_s be an ω_s-dimensional subspace of F^{Ω} such that $W_s \cap W_{s'} = 0$ for $s \neq s'$. An $\boldsymbol{\omega}$-dimensional F-valued linear network code on an acyclic network with respect to $\{W_s\}$ consists of a scalar $k_{d,e}$ for every adjacent pair (d, e) in the network as well as an Ω-dimensional column vector f_e for every channel e such that:

(7.2) $f_e = \sum_{d \in \mathrm{In}(i)} k_{d,e} f_d$, where $e \in \mathrm{Out}(i)$.

(7.3) For $s \in S$, the vectors $f_{s(1)}, f_{s(2)}, \cdots, f_{s(\omega_s)}$ for the ω_s imaginary channels terminating at the node source node s constitute a basis for the subspace W_s.

The scalar $k_{d,e}$ is called the *local encoding kernel* for the adjacent pair (d, e), while the vector f_e is called the *global encoding kernel* for the channel e.

We note that in the above definition, for given $\omega_s, s \in S$, the specific choice of the set of subspaces $\{W_s\}$ is not important. While it is convenient to choose W_s for $s \in S$ and f_e for all imaginary channels e such that the latter form the natural basis for F^{Ω}, in order to keep the definition general and to facilitate subsequent discussion, we do not impose this requirement. In fact, a linear network code as defined in Definition 7.1 that does not satisfy this requirement can readily be converted into one by means of a linear transformation.

Introduce the notations

$$f_s = \left[f_{s(1)} \; f_{s(2)} \cdots f_{s(\omega_s)} \right] \tag{7.4}$$

for $s \in S$ and

$$f_{E'} = [f_e]_{e \in E'} \tag{7.5}$$

for $E' \subset E$. In (7.5), the matrix elements f_e are put in juxtaposition. This convention will be adopted throughout this section.

Definition 7.2. An information rate tuple

$$\boldsymbol{\omega} = (\omega_s : s \in S)$$

is *linearly achievable* if for some base field F, there exists an $\boldsymbol{\omega}'$-dimensional linear code on the network, where $\boldsymbol{\omega}' \geq \boldsymbol{\omega}$ (componentwise), satisfying: For all $i \in T$, for all $s \in \beta(i)$, there exists an $|\mathrm{In}(i)| \times \omega'_s$ matrix $G_i(s)$ such that

$$f_s = f_{\mathrm{In}(i)} \cdot G_i(s). \tag{7.6}$$

The matrix $G_i(s)$ is called the *decoding kernel* at the node i for the information source generated at the source node s.

7.2 Entropy and the rank function

In this section, we establish a fundamental relation (Theorem 7.4) between entropy and the *rank function* of matrices. This relation is instrumental for the discussion in the next section, where we explore the asymptotic limitation of linear network codes for multiple sources.

Theorem 7.3. Let F be a finite field, Y be an Ω-dimensional random row vector that distributes uniformly on F^{Ω}, and A be an F-valued $\Omega \times l$ matrix. Let $Z = g(Y)$, where $g(Y) = Y \cdot A$. Then $H(Z) = \mathrm{rank}(A) \log |F|$.

Proof. Let $\mathbf{y} \in F^{\Omega}$ and $\mathbf{z} \in F^{l}$ be row vectors. Consider the system of simultaneous equations

$$\mathbf{y} \cdot A = \mathbf{z}$$

with $\mathbf{y}$ being unknown and $\mathbf{z}$ fixed, and let $S_{\mathbf{z}}$ denote the solution set for a particular $\mathbf{z}$. It is readily seen that S_0, where 0 denotes the zero vector, is a linear subspace of F^{Ω}.

For a particular $\mathbf{z}$, $S_{\mathbf{z}}$ may or may not be empty. For distinct $\mathbf{z}_1, \mathbf{z}_2 \in \text{range}(g)$, i.e., both $S_{\mathbf{z}_1}$ and $S_{\mathbf{z}_2}$ are nonempty, it is readily seen that

$$S_{\mathbf{z}_1} \cap S_{\mathbf{z}_2} = \emptyset. \tag{7.7}$$

Now regard the vectors in F^{Ω} together with vector addition as a group, and hence S_0 is a subgroup of F^{Ω}. For a fixed $\mathbf{z}$ such that $S_{\mathbf{z}}$ is nonempty, consider any $\tilde{\mathbf{y}} \in S_{\mathbf{z}}$. Then it is easy to verify that

$$S_{\mathbf{z}} = \{\tilde{\mathbf{y}} + \mathbf{y} : \mathbf{y} \in S_0\}.$$

Thus $S_{\mathbf{z}}$ is a coset of S_0 with respect to $\tilde{\mathbf{y}}$, and by the Lagrange theorem (see for example [175]), $|S_{\mathbf{z}}| = |S_0|$. It follows that $|S_{\mathbf{z}}|$ is equal to a constant for all $\mathbf{z} \in \text{range}(g)$.

Finally, for all $\mathbf{z} \in \text{range}(g)$,

$$\begin{aligned} \Pr\{Z = z\} &= \Pr\{Y \in S_z\} \\ &= \frac{|S_z|}{|F|^{\Omega}} \\ &= \frac{|S_0|}{|F|^{\Omega}}, \end{aligned}$$

which does not depend on z. Thus Z has a uniform distribution on range(g). Since range(g) is a subspace of F^l with dimension rank(A), it follows that

$$H(Z) = \log |F|^{\text{rank}(A)} = \text{rank}(A) \log |F|.$$

The theorem is proved. □

Before we proceed further, we first define a region in the entropy space $\mathcal{H}_{\mathcal{N}}$ which is closely related to the region $\Gamma^*_{\mathcal{N}}$, where we recall from Section 6.2.5 that

$$\mathcal{N} = \{Y_s : s \in S; U_e : e \in E\}.$$

Let Ω be any integer such that $\Omega \geq 1$. For each $e \in E$, associate with the random variable U_e an unspecified Ω-dimensional column vector

denoted by v_{U_e}, and for each $s \in S$, associate with the random variable Y_s an unspecified $\Omega \times \omega_s$ matrix denoted by v_{Y_s} (here v_{Y_s} is regarded as a collection of ω_s Ω-dimensional column vectors). The use of these unspecified vectors/matrices will become clear shortly. For $A \in \mathcal{Q}_{\mathcal{N}}$, let

$$v_A = [v_Z]_{Z \in A}.$$

A vector

$$\mathbf{h} = (h_A : A \in \mathcal{Q}_{\mathcal{N}})$$

as defined in (6.46) is a *rank function* for a finite base field F if there exists a collection of column vectors $\{v_Z : Z \in \mathcal{N}\}$ in F such that

$$h_A = \text{rank}(v_A) \tag{7.8}$$

for all $A \in \mathcal{Q}_{\mathcal{N}}$. We then define the region

$$\Psi^*_{\mathcal{N}} = \{\mathbf{h} \in \mathcal{H}_{\mathcal{N}} : \mathbf{h} \text{ is a rank function for some base field } F \text{ and some } \Omega \geq 1\}.$$

The possible gap between the asymptotic performance between linear and nonlinear codes, as we shall see, hinges on a gap between the region $\Psi^*_{\mathcal{N}}$ and $\Gamma^*_{\mathcal{N}}$ characterized by an inequality on the rank function known as the Ingleton inequality [181]. We first establish the following fundamental theorem.

Theorem 7.4. $\text{con}(\Psi^*_{\mathcal{N}}) \subset \overline{\Gamma}^*_{\mathcal{N}}$, where $\text{con}(\Psi^*_{\mathcal{N}})$ denotes the convex hull of $\Psi^*_{\mathcal{N}}$.

Proof. Consider $\mathbf{h} \in \Psi^*_{\mathcal{N}}$. Then for some finite base field F and some $\Omega \geq 1$, there exists a collection of vectors $\{v_Z : Z \in \mathcal{N}\}$ such that (7.8) is satisfied. Let

$$Y = \begin{bmatrix} Y_1 & Y_2 & \cdots & Y_\Omega \end{bmatrix}$$

be an Ω-dimensional row vector, where Y_i, $1 \leq i \leq \Omega$ are i.i.d. random variables each distributing uniformly on F, so that Y distributes uniformly on F^Ω. Define the random variable

$$Z = Y \cdot v_Z$$

for every $Z \in \mathcal{N}$, so that for every $A \in \mathcal{Q}_{\mathcal{N}}$,

$$[Z]_{Z \in A} = Y \cdot v_A.$$

Then by Theorem 7.3,

$$H(Z : Z \in A) = \operatorname{rank}(v_A) \log |F|. \tag{7.9}$$

From (7.8) and (7.9), we have

$$h_A = \operatorname{rank}(v_A) = (\log |F|)^{-1} H(Z : Z \in A),$$

or

$$(\log |F|) h_A = H(Z : Z \in A).$$

This implies that $(\log |F|)\mathbf{h}$ is an entropy function, or

$$(\log |F|)\mathbf{h} \in \Gamma^*_{\mathcal{N}}.$$

Since $\overline{\Gamma}^*_{\mathcal{N}}$ is a convex cone,

$$\mathbf{h} \in \overline{\Gamma}^*_{\mathcal{N}}.$$

Therefore, we conclude that

$$\Psi^*_{\mathcal{N}} \subset \overline{\Gamma}^*_{\mathcal{N}}.$$

The proof is then completed by taking the convex hull in the above. □

7.3 Can nonlinear codes be better asymptotically?

Recall the notation

$$f_{E'} = [f_e]_{e \in E'}$$

for $E' \subset E$ and introduce a similar notation

$$f_{S'} = [f_s]_{s \in S'}$$

for $S' \subset S$. For a linear code as defined in Definition 7.1, we observe that the assumption (7.1) is equivalent to

$$\operatorname{rank}(f_S) = \sum_{s \in S} \operatorname{rank}(f_s),$$

while the requirement (7.2) is equivalent to

$$\operatorname{rank}(f_{\mathrm{In}(i)\cup\mathrm{Out}(i)}) = \operatorname{rank}(f_{\mathrm{In}(i)}).$$

Furthermore, in Definition 7.2, the decoding requirement prescribed in (7.6) is equivalent to

$$\operatorname{rank}(f_{\beta(i)\cup\mathrm{In}(i)}) = \operatorname{rank}(f_{\mathrm{In}(i)}).$$

Letting

$$v_{Y_s} = f_s$$

for $s \in S$ and

$$v_{U_e} = f_e$$

for $e \in E$, and following Definitions 7.1 and 7.2 and the foregoing, we see that an information rate tuple $\boldsymbol{\omega}$ is linearly achievable if and only if for some finite base field F, there exists a collection of Ω-dimensional column vectors $\{v_Z : Z \in \mathcal{N}\}$, where $\Omega = \sum_{s\in S}\omega_s$, which satisfies the following conditions:

$$\operatorname{rank}(v_{Y_S}) = \sum_{s\in S}\operatorname{rank}(v_{Y_s}) \tag{7.10}$$

$$\operatorname{rank}(v_{Y_s}) \geq \omega_s, \quad s \in S \tag{7.11}$$

$$\operatorname{rank}(v_{U_{\mathrm{Out}(s)}\cup Y_s}) = \operatorname{rank}(v_{Y_s}), \quad s \in S \tag{7.12}$$

$$\operatorname{rank}(v_{U_{\mathrm{In}(i)\cup\mathrm{Out}(i)}}) = \operatorname{rank}(v_{U_{\mathrm{In}(i)}}), \quad i \in V\backslash S \tag{7.13}$$

$$\operatorname{rank}(v_{U_e}) \leq 1, \quad e \in E \tag{7.14}$$

$$\operatorname{rank}(v_{Y_{\beta(i)}\cup U_{\mathrm{In}(i)}}) = \operatorname{rank}(v_{U_{\mathrm{In}(i)}}), \quad i \in T. \tag{7.15}$$

In other words, there exists $\mathbf{h} \in \Psi^*_{\mathcal{N}}$ which satisfy the following conditions:

$$h_{Y_S} = \sum_{s\in S} h_{Y_s} \tag{7.16}$$

$$h_{Y_s} \geq \omega_s, \quad s \in S \tag{7.17}$$

$$h_{U_{\mathrm{Out}(s)}|Y_s} = 0, \quad s \in S \tag{7.18}$$

$$h_{U_{\mathrm{Out}(i)}|U_{\mathrm{In}(i)}} = 0, \quad i \in V\backslash S \tag{7.19}$$

$$h_{U_e} \leq 1, \quad e \in E \tag{7.20}$$

$$h_{Y_{\beta(i)}|U_{\mathrm{In}(i)}} = 0, \quad i \in T, \tag{7.21}$$

where (7.18), (7.19), and (7.21) follow because these equalities are equivalent to

$$h_{U_{\mathrm{Out}(s)} \cup Y_s} = h_{Y_s}$$
$$h_{U_{\mathrm{Out}(i)} \cup \mathrm{In}(i)} = h_{U_{\mathrm{In}(i)}}$$

and

$$h_{Y_{\beta(i)} \cup U_{\mathrm{In}(i)}} = h_{U_{\mathrm{In}(i)}},$$

which correspond to (7.12), (7.13), and (7.15), respectively. If we allow time-sharing of linear codes, then we simply replace the region $\Psi^*_{\mathcal{N}}$ by the region $\mathrm{con}(\Psi^*_{\mathcal{N}})$. The discussion above is summarized by the following definition and theorem.

Definition 7.5. Let $\mathcal{R}_{linear}$ be the set of all information rate tuple $\boldsymbol{\omega}$ such that there exists $\mathbf{h} \in \mathrm{con}(\Psi^*_{\mathcal{N}})$ satisfying (7.16) to (7.21).

Theorem 7.6. An information rate tuple is achievable by time-sharing of linear codes, possibly defined on base fields with different characteristics, if and only if $\boldsymbol{\omega} \in \mathcal{R}_{linear}$.

By setting $R_e = 1$ in (6.58), (7.16) to (7.21) become exactly the same as (6.54) to (6.59). Invoking Theorem 7.4, we see that

$$\mathcal{R}_{linear} \subset \mathcal{R}_{out},$$

which is expected.

The regions $\mathcal{R}_{in}$ and $\mathcal{R}_{out}$ are in terms of $\Gamma^*_{\mathcal{N}}$ and $\overline{\Gamma}^*_{\mathcal{N}}$, respectively, while the region $\mathcal{R}_{linear}$ is in terms of $\mathrm{con}(\Psi^*_{\mathcal{N}})$. Let A and B be any collections of vectors. It is well known that the rank function satisfies the following properties:

P1. $0 \leq \mathrm{rank}(A) \leq |A|$.
P2. $\mathrm{rank}(A) \leq \mathrm{rank}(B)$ if $A \subset B$.
P3. $\mathrm{rank}(A) + \mathrm{rank}(B) \geq \mathrm{rank}(A \cup B) + \mathrm{rank}(A \cap B)$.

In addition, a rank function also satisfies the *Ingleton inequality* [181]: For any collections of vectors $A_i, i = 1, 2, 3, 4$,

$$\begin{aligned}&\text{rank}(A_{13}) + \text{rank}(A_{14}) + \text{rank}(A_{23}) + \text{rank}(A_{24}) + \text{rank}(A_{34})\\ &\quad \geq \text{rank}(A_3) + \text{rank}(A_4) + \text{rank}(A_{12}) + \text{rank}(A_{134}) + \text{rank}(A_{234}),\end{aligned}$$

where A_{13} denotes $A_1 \cup A_3$, etc.

It has been shown in [215] that there exists entropy functions involving 4 random variables which do not satisfy the corresponding Ingleton inequality for entropy functions. The gap between $\text{con}(\Psi^*_{\mathcal{N}})$ and $\Gamma^*_{\mathcal{N}}$ so implied indicates that for certain multi-source network coding problems, $\mathcal{R}_{Out}$ may be strictly larger than $\mathcal{R}_{Linear}$, opening up the possibility that nonlinear codes can outperform linear codes asymptotically.

In fact, examples have been reported by various authors that nonlinear codes can outperform linear codes [197][199][168][196][169]. In particular, it is shown in [169] that there exist multi-source network coding problems for which nonlinear codes can outperform very general forms of linear codes, including mixtures of linear codes discussed here. This shows that there is indeed a gap between $\mathcal{R}_{Linear}$ and $\mathcal{R}_{Out}$.

Appendix A

Global Linearity versus Nodal Linearity

In this appendix, we define *global linearity* and *local linearity* of a network code based on the first principle. We shall show that global linearity implies local linearity. This justifies the generality of the local and global descriptions of a linear network code on an acyclic network in Definitions 2.4 and 2.5 of Part I.

Definition A.1. (Global Linearity) A network code on an acyclic network is globally linear if the global encoding mappings $\tilde{f}_e, e \in E$ are all linear, i.e.,

$$\tilde{f}_e(a_1 x_1 + a_2 x_2) = a_1 \tilde{f}_e(x_1) + a_2 \tilde{f}_e(x_2), \tag{A.1}$$

where x_1 and x_2 are row vectors in F^ω and $a_1, a_2 \in F$.

Definition A.2. (Local Linearity) A network code on an acyclic network is locally linear if the local encoding mappings $\tilde{k}_e, e \in E$ are all linear.

It can easily be seen by induction that local linearity implies global linearity, but the converse is not immediate. We shall prove that this is indeed the case.

We shall need a few preliminary results. We begin with the following lemma whose proof is elementary, but we nevertheless include it so that the reader can compare it with the proof of the next lemma.

Lemma A.3. Let $g : F^m \to F$, where F^m denotes the linear space of F-valued m-dimensional row vectors. Then g is linear if and only if there exists an F-valued m-dimensional column vector a such that

$$g(y) = y \cdot a$$

for all $y \in F^m$.

Proof. It is clear that if $g(y) = y \cdot a$ for all $y \in F^m$, then g is linear. We only need to prove the converse. Let u_k denote the row vector in F^m such that the kth component is equal to 1 while all other components are equal to 0. Write

$$y = \sum_k y_k u_k,$$

where y_k is the kth component of y. Then

$$\begin{aligned} g(y) &= g\left(\sum_k y_k u_k\right) \\ &= \sum_k y_k g(u_k). \end{aligned}$$

Upon letting a be the column vector $[g(u_k)]$, we have

$$g(y) = y \cdot a,$$

proving the lemma. □

This lemma has the following less trivial generalization.

Lemma A.4. Let $g : S \to F$, where S denotes a subspace of row vectors in F^m. Then g is linear if and only if there exists an F-valued

m-dimensional column vector k such that

$$g(y) = y \cdot k$$

for all $y \in S$.

Proof. Again, it is clear that if $g(y) = y \cdot k$ for all $y \in S$, then g is linear. So we only prove the converse.

Denote the dimension of S by κ. Let $\{u_1, \cdots, u_\kappa\}$ be a basis for S and let U be the $\kappa \times m$ matrix with the rows being $u_1, \cdots, u_\kappa$ in this order. Then $y \in S$ if and only if

$$y = w \cdot U$$

for some row vector $w \in F^\kappa$. Since U is full rank by construction, it's right inverse, denoted by U_r^{-1} $(m \times \kappa)$, exists, and we can write

$$w = y \cdot U_r^{-1}.$$

Define a function $\tilde{g} : F^\kappa \to F$ such that

$$\tilde{g}(w) = g(w \cdot U).$$

Since g is linear, it can readily be verified that so is $\tilde{g}$. Then by Lemma A.3,

$$\tilde{g}(w) = w \cdot a$$

for some column vector $a \in F^\kappa$. Hence,

$$\begin{aligned} g(y) &= g(w \cdot U) \\ &= \tilde{g}(w) \\ &= w \cdot a \\ &= (y \cdot U_r^{-1}) \cdot a \\ &= y \cdot (U_r^{-1} \cdot a). \end{aligned}$$

Upon letting $k = U_r^{-1} \cdot a$, we have

$$g(y) = y \cdot k,$$

proving the lemma. □

This lemma has the following immediate matrix generalization.

Corollary A.5. Let $g : S \to F^l$, where S denotes a subspace of row vectors in F^m. Then g is a linear transformation if and only if there exists an F-valued matrix K with dimension $m \times l$ such that

$$g(y) = y \cdot K$$

for all $y \in S$.

Now consider a globally linear network code and any non-source node i. Let $\tilde{K}_i$ be the local encoding mapping at i, i.e.,

$$(\tilde{f}_d(x), d \in \mathrm{In}(i)) \mapsto (\tilde{f}_e(x), e \in \mathrm{Out}(i)).$$

Introduce the notations

$$\tilde{f}_{\mathrm{In}(i)}(x) = [\tilde{f}_d(x)]_{d \in \mathrm{In}(i)}$$

and

$$f_{\mathrm{In}(i)} = [f_d]_{d \in \mathrm{In}(i)},$$

where $\tilde{f}_{\mathrm{In}(i)}(x)$ and $f_{\mathrm{In}(i)}$ are row vectors, and recall that f_d denotes the global encoding kernel of the channel d. In a similar fashion, $\tilde{f}_{\mathrm{Out}(i)}(x)$ and $f_{\mathrm{Out}(i)}$ are defined. It is easy to see that $\{\tilde{f}_{\mathrm{In}(i)}(x) : x \in F^\omega\}$ forms a subspace (of row vectors) in $F^{|\mathrm{In}(i)|}$. In other words, $\tilde{K}_i$ is a mapping from a subspace of $F^{|\mathrm{In}(i)|}$ to $F^{|\mathrm{Out}(i)|}$.

We now show that encoding mapping $\tilde{K}_i$ is linear. Let

$$y_j = \tilde{f}_{\mathrm{In}(i)}(x_j)$$

for $j = 1, 2$. Then for any $c_1, c_2 \in F$,

$$\begin{aligned}
\tilde{K}_i(c_1 y_1 + c_2 y_2) &= \tilde{K}_i(c_1 \tilde{f}_{\mathrm{In}(T)}(x_1) + c_2 \tilde{f}_{\mathrm{In}(T)}(x_2)) \\
&= \tilde{K}_i(\tilde{f}_{\mathrm{In}(T)}(c_1 x_1 + c_2 x_2)) \\
&= \tilde{f}_{\mathrm{Out}(T)}(c_1 x_1 + c_2 x_2) \\
&= c_1 \tilde{f}_{\mathrm{Out}(T)}(x_1) + c_2 \tilde{f}_{\mathrm{Out}(T)}(x_2) \\
&= c_1 \tilde{K}_i(\tilde{f}_{\mathrm{In}(T)}(x_1)) + c_2 \tilde{K}_i(\tilde{f}_{\mathrm{In}(T)}(x_2)) \\
&= c_1 \tilde{K}_i(y_1) + c_2 \tilde{K}_i(y_2).
\end{aligned}$$

Thus $\tilde{K}_i$ is linear. Hence, global linearity implies local linearity.

Now since $\tilde{K}_i$ is linear, by Corollary A.5, there exists an $|\mathrm{In}(i)| \times |\mathrm{Out}(i)|$ matrix K_i (encoding kernel for the node i) such that

$$g_i(y) = y \cdot K_i$$

for all $\{\tilde{f}_{\mathrm{In}(i)}(x) : x \in F^\omega\}$. Then for any row vector $x \in F^\omega$, we have

$$\begin{aligned} x \cdot f_{\mathrm{Out}(i)} &= \tilde{f}_{\mathrm{Out}(i)}(x) \\ &= \tilde{K}_i(\tilde{f}_{\mathrm{In}(i)}(x)) \\ &= \tilde{f}_{\mathrm{In}(i)}(x) \cdot K_i \\ &= (x \cdot f_{\mathrm{In}(i)}) \cdot K_i \\ &= x \cdot (f_{\mathrm{In}(i)} \cdot K_i). \end{aligned}$$

Since the above holds for every $x \in F^\omega$, it implies that

$$f_{\mathrm{Out}(i)} = f_{\mathrm{In}(i)} \cdot K_i,$$

or for every $e \in \mathrm{Out}(T)$,

$$f_e = \sum_{d \in \mathrm{In}(T)} k_{d,e} f_e.$$

This justifies Definition 2.5, and we have shown that this definition as well as Definition 2.4 define the most general linear network code on an acyclic network.

Acknowledgements

The authors would like to thank Chung Ping Kwong and David Tse for the useful discussions, and Siu-Wai Ho for converting part of the manuscript from Word to LaTeX. They also would like to thank Ken Zeger for clarifying their results in [169]. The work of Raymond Yeung and Bob Li were partially supported by grants from the Research Grant Council of the Hong Kong Special Administrative Region, China (RGC Ref. No. CUHK4214/03E and 414005).

References

Literature Survey

[1] R. W. Yeung, "Multilevel diversity coding with distortion," *IEEE Trans. Inform. Theory*, IT-41: 412-422, 1995.

[2] K. P. Hau, "Multilevel diversity coding with independent data streams," M.Phil. thesis, The Chinese University of Hong Kong, Jun. 1995.

[3] J. R. Roche, R. W. Yeung, and K. P. Hau, "Symmetrical multilevel diversity coding," *IEEE Trans. Inform. Theory*, IT-43: 1059-1064, 1997.

[4] R. Ahlswede, N. Cai, and R. W. Yeung, "Network information flow theory," 1998 IEEE International Symposium on Information Theory, MIT, Aug 16-21, 1998.

[5] S.-Y. R. Li and R. W. Yeung, "Network multicast flow via linear coding," International Symposium on Operations Research and its Applications (ISORA 98), Kunming, China, pp. 197-211, Aug 1998.

[6] R. W. Yeung and Z. Zhang, "On symmetrical multilevel diversity coding," *IEEE Trans. Inform. Theory*, IT-45: 609-621, 1999.

[7] R. W. Yeung and Z. Zhang, "Distributed source coding for satellite communications," *IEEE Trans. Inform. Theory*, IT-45: 1111-1120, 1999.

[8] S.-Y. R. Li and R. W. Yeung, "Single-source network information flow," 1999 IEEE Information Theory Workshop, Metsovo, Greece, Jun 27-Jul 1, 1999.

[9] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inform. Theory*, IT-46: 1204-1216, 2000.

[10] R. Koetter and M. Medard, "An algebraic approach to network coding and robust networks," 2001 IEEE International Symposium on Information Theory, Washington, DC, Jun 24-29, 2001.

[11] T. Ho, M. Medard and R. Koetter, "A coding view of network recovery and managment for single receiver communication," 2002 Conference on Information Science and Systems, Princeton University, Mar 20-22, 2002.

[12] R. Koetter and M. Medard, "Beyond Routing: An algebraic approach to network coding," INFOCOM 2002, New York, NY, USA, Jun 23-27, 2002.

[13] S. Borade, "Network information flow: Limits and achievability," 2002 IEEE International Symposium on Information Theory, Lausanne, Switzerland, Jun 30-Jul 5, 2002.

[14] N. Cai and R. W. Yeung, "Secure network coding," 2002 IEEE International Symposium on Information Theory, Lausanne, Switzerland, Jun 30-Jul 5, 2002.

[15] N. Cai and R. W. Yeung, "Network coding and error correction," 2002 IEEE Information Theory Workshop, Bangalore, India, Oct 20-25, 2002.

[16] S.-Y. R. Li, R. W. Yeung and N. Cai, "Linear network coding," *IEEE Trans. Inform. Theory*, IT-49: 371-381, 2003.

[17] M. Effros, M. Medard, T. Ho, S. Ray, D. Karger, R. Koetter, "Linear network codes: A unified framework for source, channel, and network coding," DIMACS workshop on Network Information Theory, Mar 2003.

[18] T. Ho, M. Medard, and R. Koetter, "An information theoretic view of network management," INFOCOM 2003, San Francisco, CA, USA, Mar 30 - Apr 3, 2003.

[19] E. Erez and M. Feder, "Capacity region and network codes for two receivers multicast with private and common data," Workshop on Coding, Cryptography and Combinatorics, 2003.

[20] T. Noguchi, T. Matsuda, M. Yamamoto, "Performance evaluation of new multicast architecture with network coding," *IEICE Trans. Comm.*, vol. E86-B, 1788-1795, 2003.

[21] P. Sanders, S. Egner, and L. Tolhuizen, "Polynomial time algorithms for network information flow," 15th ACM Symposium on Parallelism in Algorithms and Architectures, San Diego, CA, Jun 7-9, 2003.

[22] T. Ho, D. Karger, M. Medard, and R. Koetter, "Network coding from a network flow perspective," 2003 IEEE International Symposium on Information Theory, Yokohama, Japan, Jun 29-Jul 4, 2003.

[23] T. Ho, R. Koetter, M. Medard, D. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," 2003 IEEE International Symposium on Information Theory, Yokohama, Japan, Jun 29-Jul 4, 2003.

[24] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," 41st Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL, Oct 2003.

[25] E.Erez and M. Feder, "On codes for network multicast," 41st Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL, Oct 2003.

[26] M. Feder, D. Ron, A. Tavory, "Bounds on linear codes for network multicast," Electronic Colloquium on Computational Complexity (ECCC) 10(033): (2003).

[27] T. Ho, M. Medard, J. Shi, M. Effros, and D. R. Karger, "On randomized network coding," 41st Annual Allerton Conference on Communication Control and Computing, Monticello, IL, Oct 2003.

[28] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Networking*, vol. 11, 782-795, 2003.

[29] A. Rasala-Lehman and E. Lehman, "Complexity classification of network information flow problems," 41st Annual Allerton Conference on Communication Control and Computing, Monticello, IL, Oct 2003.

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

[31] A. Ramamoorthy, J. Shi, and R. Wesel, "On the capacity of network coding for wireless networks," 41st Annual Allerton Conference on Communication Control and Computing, Monticello, IL, Oct 2003.

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

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

[34] L. Song, R. W. Yeung and N. Cai, "Zero-error network coding for acyclic networks," *IEEE Trans. Inform. Theory*, IT-49: 3129-3139, 2003.

[35] A. Lehman and E. Lehman "Complexity classification of network information flow problems," ACM-SIAM Symposium on Discrete Algorithms, New Orleans, LA, Jan 11-13, 2004.

[36] Y. Zhu, B. Li, J. Guo, "Multicast with network coding in application-layer overlay networks," *IEEE J. Selected Areas Comm.* (special issue on Service Overlay Networks), vol. 22, 107-120, 2004.

[37] K. Jain, "Security based on network topology against the wiretapping attack," *IEEE Wireless Comm.*, 68-71, Feb 2004.

[38] S. Deb, C. Choute, M. Medard, and R. Koetter, "Data harvesting: A random coding approach to rapid dissemination and efficient storage of data," IEEE INFOCOM 2005, Miami, FL, USA, Mar 13-17, 2005.

[39] R. Dougherty, C. Freiling, and K. Zeger, "Linearity and solvability in multicast networks," 38th Annual Conference on Information Sciences and Systems, Princeton, NJ, Mar 17-19, 2004.

[40] C. Fragouli, E. Soljanin, A. Shokrollahi, "Network coding as a coloring problem," 38th Annual Conference on Information Sciences and Systems, Princeton, NJ, Mar 17-19, 2004.

[41] T. Ho, M. Medard, M. Effros, R. Koetter, "Network coding for correlated sources," 38th Annual Conference on Information Sciences and Systems, Princeton, NJ, Mar 17-19, 2004.

[42] Z. Li, B. Li, "Network coding in undirected networks," 38th Annual Conference on Information Sciences and Systems, Princeton, NJ, Mar 17-19, 2004.

[43] D. S. Lun, N. Ratnakar, R. Koetter, M. Medard, E. Ahmed, and H. Lee, "Achieving minimum-cost Multicast: A decentralized approach based on network coding," IEEE INFOCOM 2005, Miami, FL, USA, Mar 13-17, 2005.

[44] Y. Wu, P. A. Chou, Q. Zhang, K. Jain, W. Zhu, and S.-Y. Kung, "Achievable throughput for multiple multicast sessions in wireless ad hoc networks," submitted to IEEE Globecom 2004.

[45] S. Deb and M. Medard, "Algebraic Gossip: A network coding approach to optimal multiple rumor mongering," preprint.

[46] D. Lun, M. Medard, T. Ho, and R. Koetter, "Network coding with a cost criterion," MIT LIDS TECHNICAL REPORT P-2584, Apr 2004.

[47] Z. Li, B. Li, D. Jiang, and L. C. Lau, "On achieving optimal end-to-end throughput in data networks: Theoretical and empirical studies," Technical Report, University of Toronto, May 2004.

[48] S. Che and X. Wang, "Network coding in wireless network," 16th International Conference on Computer Communication, China, 2004.

[49] E. Erez and M. Feder, "Convolutional network codes," 2004 IEEE International Symposium on Information Theory, Chicago, IL, Jun 27-Jul 2, 2004.

[50] C. Fragouli and E. Soljanin, "Required alphabet size for linear network coding," 2004 IEEE International Symposium on Information Theory, Chicago, IL, USA, Jun 27 -Jul 2.

[51] C. Fragouli and E. Soljanin, "A connection between network coding and convolutional codes," IEEE International Conference on Communications, Paris, France, Jun 20-24, 2004.

[52] T. Ho, B. Leong, M. Medard, R.Koetter, Y. Chang, and M. Effros, "On the utility of network coding in dynamic environments," International Workshop on Wireless Ad-hoc Networks (IWWAN), University of Oulu, Finland, May 31-Jun 3, 2004.

[53] T. Ho, B. Leong, R. Koetter, M. Medard, M. Effros, and D. R. Karger, "Byzantine modification detection in multicast networks using randomized network coding," 2004 IEEE International Symposium on Information Theory, Chicago, IL, Jun 27-Jul 2, 2004.

[54] G. Kramer and S. A. Savari, "Cut sets and information flow in networks of two-way channels," 2004 IEEE International Symposium on Information Theory, Chicago, IL, Jun 27-Jul 2, 2004.

[55] C. K. Ngai and R.W. Yeung, "Multisource network coding with two sinks," International Conference on Communications, Circuits and Systems (ICCCAS), Chengdu, China, Jun 27-29, 2004.

[56] Y. Wu, P. A. Chou, K. Jain, "A comparison of network coding and tree packing," 2004 IEEE International Symposium on Information Theory, Chicago, IL, Jun 27-Jul 2, 2004.

[57] Y. Cui, Y. Xue, and K. Nahrstedt, "Optimal distributed multicast routing using network coding: Theory and applications," preprint UIUCDCS-R-2004-2473, University of Illinois, Urbana-Champaign, Aug 2004.

[58] Y. Wu, P. A. Chou, and S.-Y. Kung, "Information exchange in wireless networks with network coding and physical-layer broadcast," Microsoft Technical Report, MSR-TR-2004-78, Aug 2004.

[59] J. Feldman, T. Malkin, C. Stein, and R. A. Servedio, "On the capacity of secure network coding," 42nd Annual Allerton Conference on Communication, Control, and Computing, Sept 29-Oct 1, 2004.

[60] C. Fragouli and E. Soljanin, "On average throughput benefit for network coding," 42nd Annual Allerton Conference on Communication, Control, and Computing, Sept 29-Oct 1, 2004.

[61] N. Harvey, R. Kleinberg, and A. Lehman, "Comparing network coding with multicommodity flow for the k-pairs communication problem," MIT LCS Technical Report 964, Sept 28, 2004.

[62] S. Jaggi, M. Effros T. C. Ho, and M. Medard, "On linear network coding," 42nd Annual Allerton Conference on Communication, Control, and Computing, Sept 29-Oct 1, 2004.

[63] D. S. Lun, M. Medard, and M. Effros, "On coding for reliable communication over packet networks," 42nd Annual Allerton Conference on Communication, Control, and Computing, Sept 29-Oct 1, 2004.

[64] A. Ramamoorthy, K. Jain, P. A. Chou, and M. Effros, "Separating distributed source coding from network coding," 42nd Annual Allerton Conference on Communication, Control, and Computing, Sept 29-Oct 1, 2004.

[65] Y. Wu , K. Jain, and S.-Y. Kung, "A unification of Edmonds' graph theorem and Ahlswede et al's network coding theorem," 42nd Annual Allerton Conference on Communication, Control, and Computing, Sept 29-Oct 1, 2004.

[66] A. Argawal and M. Charikar, "On the advantage of network coding for improving network throughput," 2004 IEEE Information Theory Workshop, San Antonio, Oct 25-29, 2004.

[67] R. Dougherty, C. Freiling, and K. Zeger, "Linearity and solvability in multicast networks," *IEEE Trans. Inform. Theory*, IT-50: 2243-2256, 2004.

[68] C. Fragouli and E. Soljanin, "Decentralized network coding," 2004 IEEE Information Theory Workshop, San Antonio, Oct 25-29, 2004.

[69] J. Han and P. H. Siegel, "Reducing acyclic network coding problems to single-transmitter-single-demand form," 42nd Allerton Conference on Communication, Control, and Computing, Monticello, IL, Spet 29-Oct 1, 2004.

[70] D. S. Lun, M. Medard, T. Ho , and R. Koetter, "Network coding with a cost criterion," International Symposium on Information Theory and its Applications, Parma, Italy, Oct 10-13, 2004.

[71] C. K. Ngai and R. W. Yeung, "Network coding gain of combination networks," 2004 IEEE Information Theory Workshop, San Antonio, Oct 25-29, 2004.

[72] D. Tuninetti and C. Fragouli, "Processing along the way: Forwarding vs. Coding," International Symposium on Information Theory and its Applications, Parma, Italy, Oct 10-13, 2004.

[73] Y. Wu, P. A. Chou, and S.-Y. Kung, "Minimum-energy multicast in mobile ad hoc networks using network coding," 2004 IEEE Information Theory Workshop, San Antonio, Oct 25-29, 2004.

[74] R. W. Yeung, "Two approaches to quantifying the bandwidth advantage of network coding," presented at 2004 IEEE Information Theory Workshop, San Antonio, Oct 25-29, 2004.

[75] S. C. Zhang, I. Koprulu, R. Koetter, and D. L. Jones, "Feasibility analysis of stochastic sensor networks," IEEE International Conference on Sensor and Ad hoc Communications and Networks, Santa Clara, CA, USA, Oct 4-7, 2004.

[76] N. Harvey, D. Karger, and K. Murota, "Deterministic network coding by matrix completion," ACM-SIAM Symposium on Discrete Algorithms (SODA), Vancouver, British Columbia, Canada, Jan 23-25, 2005.

[77] M. Langberg, A. Sprintson and J. Bruck, "The encoding complexity of network coding," ETR063, California Institute of Technology.

[78] A. R. Lehman and E. Lehman, "Network coding: Does the model need tuning?" ACM-SIAM Symposium on Discrete Algorithms (SODA), Vancouver, British Columbia, Canada, Jan 23-25, 2005.

[79] Y. Wu, P. A. Chou, Q. Zhang, K. Jain, W. Zhu, and S.-Y. Kung, "Network planning in wireless ad hoc networks: a cross-layer approach," *IEEE J. Selected Areas Comm.* (Special Issue on Wireless Ad Hoc Networks), vol. 23, 136-150, 2005.

[80] A. Rasala-Lehman, "Network coding," Ph.D. thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Feb 2005.

[81] X. B. Liang, "Matrix games in the multicast networks: Maximum information flows with network switching," revised version (original version: Mar 2005), preprint.

[82] Y. Wu, P. A. Chou, S.-Y. Kung, "Information exchange in wireless networks with network coding and physical-layer broadcast," 2005 Conference on Information Science and Systems, Johns Hopkins University, Mar 16-18, 2005.

[83] Y. Wu and S.-Y. Kung, "Reduced-complexity network coding for multicasting over ad hoc networks," IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Philadelphia, PA, USA, Mar 18-23, 2005.

[84] S. Acedański, S. Deb, M. Medard, and R. Koetter, "How good is random linear coding based distributed networked storage?" NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.

[85] K. Bhattad and K. R. Nayayanan, "Weakly secure network coding," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.

[86] T. Coleman, M. Medard, and M. Effros, "Practical universal decoding for combined routing and compression in network coding," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.

[87] A. G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Ubiquitous access to distributed data in large-scale sensor networks through decentralized erasure codes," The Fourth International Symposium on Information Processing in Sensor Networks (IPSN'05), UCLA, Los Angeles, CA, Apr 25-27, 2005.

[88] E. Erez and M. Feder, "Convolutional network codes for cyclic networks," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.

[89] T. Ho, B. Leong, R. Koetter, M. Medard, "Distributed asynchronous algorithms for multicast network coding," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.

[90] T. Ho, M. Medard, and R. Koetter, "An information theoretic view of network management," *IEEE Trans. Inform. Theory*, IT-51: 1295-1312, 2005.

[91] R. Khalili and K. Salamatian, "On the capacity of multiple input erasure relay channels," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.

[92] D. S. Lun, M. Medard, D. Karger, "On the dynamic multicast problem for coded networks," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.
[93] D. Petrović, K. Ramchandran, and J. Rabaey, "Overcoming untuned radios in wireless networks with network coding," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.
[94] N. Ratnakar and G. Kramer, "The multicast capacity of acyclic, deterministic, relay networks with no interference," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.
[95] S. Riis and R. Alswede, "Problems in network coding and error correcting codes," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.
[96] Y. Sagduyu and A. Ephremides, "Joint scheduling and wireless network coding," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.
[97] J. Widmer, C. Fragouli, and J.-Y. Le Boudec, "Energy-efficient broadcasting in wireless ad-hoc networks," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.
[98] Y. Wu, V. Stankovic, Z. Xiong, and S.-Y. Kung, "On practical design for joint distributed source and network coding," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.
[99] X. Yan, J. Yang, and Z. Zhang, "An improved outer bound for multisource multisink network coding," NetCod 2005, Riva del Garda, Italy, Apr 7, 2005.
[100] C. Gkantsidis and P. R. Rodriguez, "Network coding for large scale content distribution," IEEE INFOCOM 2005, Miami, FL, Mar 13-17, 2005.
[101] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Inform. Theory*, IT 51: 1973-1982, 2005.
[102] Y. Wu, M. Chiang, and S.-Y. Kung, "Distributed utility maximization for network coding based multicasting: a critical cut approach," submitted to IEEE INFOCOM 2006.
[103] Y. Wu and S.-Y. Kung, "Distributed utility maximization for network coding based multicasting: a shorted path approach," submitted to IEEE INFOCOM 2006.
[104] K. K. Chi and X. M. Wang, "Analysis of network error correction based on network coding," *IEE Proc. Commun.*, vol. 152, No. 4, 393-396, 2005.
[105] R. Dougherty, C. Freiling, and K. Zeger, "Insufficiency of linear coding in network information flow," *IEEE Trans. Inform. Theory*, IT-51: 2745-2759, 2005.
[106] H. Wang, P. Fan, and Z. Cao, "On the statistical properties of maximum flows based on random graphs," IEEE 2005 International Symposium on Microwave, Antenna, Propagation and EMC Technologies for Wireless Communications, Beijing, China, Aug 8-12, 2005.
[107] J. Widmer and J.-Y. Le Boudec, "Network coding for efficient communication in extreme networks," Workshop on Delay Tolerant Networking and Related Topics (WDTN-05), Philadelphia, PA, USA, Aug 22-26, 2005.
[108] X. Bao and J. (T). Li, "Matching code-on-graph with network-on-graph: Adaptive network coding for wireless relay networks," 43rd Allerton Conference on Communication, Control, and Computing, Monticello, IL, Sept 28-30, 2005.

[109] K. Bhattad, N. Ratnakar, R. Koetter, and K. R. Narayanan, "Minimal network coding for multicast," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[110] Y. Cassuto and J. Bruck, "Network coding for nonuniform demands," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[111] T. H. Chan, "On the optimality of group network codes," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[112] C. Chekuri, C. Fragouli, and E. Soljanin, "On average throughput and alphabet size in network coding," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[113] S. Deb, M. Medard, and C. Choute, "On random network coding based information dissemination," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[114] R. Dougherty, C. Freiling, and K. Zeger, "Insufficiency of linear coding in network information flow," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[115] R. Dougherty and K. Zeger, "Nonreversibility of multiple unicast networks," 43rd Allerton Conference on Communication, Control, and Computing, Monticello, IL, Sept 28-30, 2005.

[116] E. Erez and M. Feder, "Efficient network codes for cyclic networks," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[117] C. Fragouli and A. Markopoulou, "A network coding approach to network monitoring," 43rd Allerton Conference on Communication, Control, and Computing, Monticello, IL, Sept 28-30, 2005.

[118] N. Harvey and R. Kleinberg, "Tighter cut-based bounds for k-pairs communication problems," 43rd Allerton Conference on Communication, Control, and Computing, Monticello, IL, Sept 28-30, 2005.

[119] C. Hausl, F. Schreckenbach, I. Oikonomidis, and G. Bauch, "Iterative network and channel decoding on a Tanner graph," 43rd Allerton Conference on Communication, Control, and Computing, Monticello, IL, Sept 28-30, 2005.

[120] T. Ho, B. Leong, Y.-H. Chang, Y. Wen, and R. Koetter, "Network monitoring in multicast networks using network coding," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[121] T. Ho and H. Viswanathan, "Dynamic algorithms for multicast with intra-session network coding," 43rd Allerton Conference on Communication, Control, and Computing, Monticello, IL, Sept 28-30, 2005.

[122] K. Jain, "On the power (saving) of network coding," 43rd Allerton Conference on Communication, Control, and Computing, Monticello, IL, Sept 28-30, 2005.

[123] S. Katti, D. Katabi, W. Hu, and R. Hariharan, "The importance of being opportunistic: Practical network coding for wireless environments," 43rd Allerton Conference on Communication, Control, and Computing, Monticello, IL, Sept 28-30, 2005.

[124] G. Kramer and S. Savari, "Progressive d-separating edge set bounds on network coding rates," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[125] M. Langberg, A. Sprintson, and J. Bruck, "The encoding complexity of network coding," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[126] A. Lee and M. Medard, "Simplified random network codes for multicast networks," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[127] S.-Y. R. Li, N. Cai, and R. W. Yeung, "On theory of linear network coding," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[128] R. W. Yeung and S-Y. R. Li, "Polynomial time construction of generic linear network codes," 43rd Allerton Conference on Communication, Control, and Computing, Monticello, IL, Sept 28-30, 2005.

[129] D. Lun, M. Medard, R. Koetter, and M. Effros, "Further results on coding for reliable communication over packet networks," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[130] N. Ratnakar and G. Kramer, "On the separation of channel and network coding in Aref networks," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[131] Y. Sagduyu and A. Ephremides, "Crosslayer design for distributed MAC and network coding in wireless ad hoc networks," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[132] X. Wu, B. Ma, and N. Sarshar, "Rainbow network problems and multiple description coding," 2005 IEEE International Symposium on Information Theory, Adelaide, Australia, Sept 4-9, 2005.

[133] Y. Xi and E. M. Yeh, "Distributed algorithms for minimum cost multicast with network coding," 43rd Allerton Conference on Communication, Control, and Computing, Monticello, IL, Sept 28-30, 2005.

[134] K. Cai and P. Fan, "An algebraic approach to link failures based on network coding," submitted to *IEEE Trans. Inform. Theory*.

[135] N. Cai and R. W. Yeung, "The Singleton bound for network error-correcting codes," 4th International Symposium on Turbo Codes and Related Topics, Munich, Germany, Apr 3-7, 2006.

[136] Y. Ma, W. Li, P. Fan, and X. Liu, "Queuing model and delay analysis on network coding," International Symposium on Communications and Information Technologies 2005, Beijing, China, Oct 12-14, 2005.

[137] R. W. Yeung, "Avalanche: A network coding analysis," preprint.

[138] Y. Wu, P. A. Chou, and S.-Y. Kung, "Minimum-energy multicast in mobile ad hoc networks using network coding," *IEEE Trans. Comm.*, vol. 53, 1906-1918, 2005.

[139] P. Fan, "Upper bounds on the encoding complexity of network coding with acyclic underlying graphs," preprint.

[140] J. Barros and S. D. Servetto, "Network Information Flow with Correlated Sources," *IEEE Trans. Inform. Theory*, IT-52: 155-170, 2006.

[141] Y. Wu, "Network coding for multicasting," Ph.D. Dissertation, Dept. of Electrical Engineering, Princeton University, Nov 2005.
[142] J. Cannons, R. Dougherty, C. Freiling, and K. Zeger, "Network Routing Capacity," *IEEE Trans. Inform. Theory*, IT-52: 777-788, 2006.
[143] C. Fragouli and E. Soljanin, "Information flow decomposition for network coding," *IEEE Trans. Inform. Theory*, IT-52: 829-848, 2006.
[144] A. L. Toledo and X. Wang, "Efficient multipath in sensor networks using diffusion and network coding," 40th Annual Conference on Information Sciences and Systems, Princeton University, NJ, USA, Mar 22-24, 2006.
[145] R. Dougherty, C. Freiling, and K. Zeger, "Unachievability of network coding capacity," to appear in *IEEE Trans. Inform. Theory* and *IEEE/ACM Trans. Networking* (joint special issue on Networking and Information Theory).
[146] R. Dougherty, C. Freiling, and K. Zeger, "Matroids, networks, and non-Shannon information inequalities," submitted to *IEEE Trans. Inform. Theory.*
[147] N. J. A. Harvey, R. Kleinberg and A. R. Lehman, "On the capacity of information networks," to appear in *IEEE Trans. Inform. Theory* and *IEEE/ACM Trans. Networking* (joint special issue on Networking and Information Theory).
[148] T. Ho, R. Koetter, M. Medard, M. Effros, J. Shi, and D. Karger, "Toward a random operation of networks," submitted to *IEEE Trans. Inform. Theory.*
[149] S. Riis, "Reversible and irreversible information networks" submitted.
[150] L. Song, R. W. Yeung and N. Cai, "A separation theorem for single-source network coding," to appear in *IEEE Trans. Inform. Theory.*
[151] X. Yan, J. Yang, and Z. Zhang, "An outer bound for multi-source multi-sink network coding with minimum cost consideration," to appear in *IEEE Trans. Inform. Theory* and *IEEE/ACM Trans. Networking* (joint special issue on Networking and Information Theory).
[152] R. W. Yeung and N. Cai, "Network error correction, Part I, Basic concepts and upper bounds," to appear in *Communications in Information and Systems.*
[153] N. Cai and R. W. Yeung, "Network error correction, Part II: Lower bounds," to appear in *Communications in Information and Systems.*
[154] S.-Y. R. Li and R. W. Yeung, "On the theory of linear network coding," submitted to *IEEE Trans. Inform. Theory.*
[155] S.-Y. R. Li and R. W. Yeung, "On convolutional network coding," submitted to *IEEE Trans. Inform. Theory.*
[156] Z. Zhang, "Network error correction coding in packetized networks," submitted to *IEEE Trans. Inform. Theory.*

References cited in text

[157] "Network Coding Homepage," http://www.networkcoding.info.
[158] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inform. Theory*, vol. IT-46, pp. 1204–1216, 2000.
[159] A. Argawal and M. Charikar, "On the advantage of network coding for improving network throughput," in *2004 IEEE Information Theory Workshop*, (San Antonio), October 25–29, 2004.

[160] T. Berger, "Multiterminal source coding," in *The Information Theory Approach to Communications*, (G. Longo, ed.), 1978. CISM Courses and Lectures #229, Springer-Verlag, New York.
[161] E. R. Berlekamp, "Block coding for the binary symmetric channel with noiseless, delayless feedback," in *Error Correcting Codes*, (H. B. Mann, ed.), (Wiley, New York), 1968.
[162] R. E. Blahut, *Theory and practice of error control codes.* 1983.
[163] J. Byers, M. Luby, and M. Mitzenmacher, "A digital foundation approach to asynchronous reliable multicast," *IEEE J. Selected Areas Comm.*, vol. 20, pp. 1528–1540, (A preliminary versin appeared in ACM SIGCOMM '98.), 2002.
[164] N. Cai and R. W. Yeung, "Network error correction, Part II: Lower bounds," to appear in *Communications in Information and Systems.*
[165] N. Cai and R. W. Yeung, "Secure network coding," in *2002 IEEE International Symposium on Information Theory*, (Lausanne, Switzerland), June 30–July 5 2002.
[166] T. M. Cover and J. A. Thomas, *Elements of information theory.* 1991.
[167] R. Dougherty, C. Freiling, and K. Zeger, "Matroids, networks, and non-shannon information inequalities," submitted to *IEEE Trans. Inform. Theory.*
[168] R. Dougherty, C. Freiling, and K. Zeger, "Linearity and solvability in multicast networks," in *38th Annual Conference on Information Sciences and Systems*, (Princeton, NJ), March 17–19 2004.
[169] R. Dougherty, C. Freiling, and K. Zeger, "Insufficiency of linear coding in network information flow," *IEEE Trans. Inform. Theory*, vol. IT-51, pp. 2745–2759, 2005.
[170] E. Erez and M. Feder, "Capacity region and network codes for two receivers multicast with private and common data," in *Workshop on Coding, Cryptography and Combinatorics*, 2003.
[171] E. Erez and M. Feder, "Convolutional network codes," in *2004 IEEE International Symposium on Information Theory*, (Chicago, IL), June 27–July 2 2004.
[172] E. Erez and M. Feder, "Convolutional network codes for cyclic networks," in *NetCod 2005*, (Riva del Garda, Italy), April 7, 2005.
[173] C. Fragouli, J.-Y. L. Boudec, and J. Widmer, "Network Coding: An Instant Primer," http://algo.epfl.ch/christin/primer.ps.
[174] C. Fragouli and E. Soljanin, "A connection between network coding and convolutional codes," in *IEEE International Conference on Communications*, (Paris, France), pp. 20–24, June 2004.
[175] J. B. Fraleigh, *A first course in abstract algebra.* 7th ed., 2003.
[176] C. Gkantsidis and P. R. Rodriguez, "Network coding for large scale content distribution," in *IEEE INFOCOM 2005*, (Miami, FL), March 13–17, 2005.
[177] K. P. Hau, *Multilevel diversity coding with independent data streams.* June 1995. M.Phil. thesis, The Chinese University of Hong Kong.
[178] S. Haykin, "Communications Systems," Wiley, 2001.

[179] T. Ho, R. Koetter, M. Medard, D. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *2003 IEEE International Symposium on Information Theory*, (Yokohama, Japan), June 29–July 4 2003.

[180] T. Ho, B. Leong, R. Koetter, M. Medard, M. Effros, and D. R. Karger, "Byzantine modification detection in multicast networks using randomized network coding," in *2004 IEEE International Symposium on Information Theory*, (Chicago, IL), June 27–July 2 2004.

[181] A. W. Ingleton, "Representation of matroids," in *Combinatorial Mathematics and its Applications*, (D. J. A. Welsh, ed.), (London), pp. 149–167, Academic Press, 1971.

[182] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A scalable and robust communication paradigm for sensor networks," in *6th Annual International Conference on Mobile Computing and Networking (Mobicom 2000)*, (Boston, MA, USA), August 6–11 2000.

[183] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Inform. Theory*, vol. IT-51, pp. 1973–1982, 2005.

[184] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Networking*, vol. 11, pp. 782–795, 2003.

[185] G. Kramer and S. A. Savari, "Cut sets and information flow in networks of two-way channels," in *2004 IEEE International Symposium on Information Theory*, (Chicago, IL), June 27–July 2 2004.

[186] S.-Y. R. Li and R. W. Yeung, "On Convolutional Network Coding," submitted to *IEEE Trans. Inform. Theory*.

[187] S.-Y. R. Li and R. W. Yeung, "On the Theory of Linear Network Coding," submitted to *IEEE Trans. Inform. Theory*.

[188] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inform. Theory*, vol. IT-49, pp. 371–381, 2003.

[189] Z. Li and B. Li, "Network coding in undirected networks," in *38th Annual Conference on Information Sciences and Systems*, (Princeton, NJ), March 17–19 2004.

[190] S. Lin and D. J. Costello Jr., *Error control coding: Fundamentals and applications*. 1983.

[191] D. Lun, M. Medard, R. Koetter, and M. Effros, "Further results on coding for reliable communication over packet networks," in *2005 IEEE International Symposium on Information Theory*, (Adelaide, Australia), September 4–9 2005.

[192] D. S. Lun, M. Medard, and M. Effros, "On coding for reliable communication over packet networks," in *42nd Annual Allerton Conference on Communication, Control, and Computing*, September 29–October 1, 2004.

[193] M. Mitzenmacher, "Digital fountain: A survey and look forward," in *2004 IEEE Information Theory Workshop*, (San Antonio, TX), October 24–29 2004.

[194] C. K. Ngai and R. W. Yeung, "Multisource network coding with two sinks," in *International Conference on Communications, Circuits and Systems (ICCCAS)*, (Chengdu, China), June 27–29 2004.

[195] C. H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: Algorithms and complexity*. 1982.

[196] A. Rasala-Lehman, *Network coding*. Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, February 2005.

[197] A. Rasala-Lehman and E. Lehman, "Complexity classification of network information flow problems," in *41st Annual Allerton Conference on Communication Control and Computing*, (Monticello, IL), October 2003.

[198] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *SIAM Journal Appl. Math.*, vol. 8, pp. 300–304, 1960.

[199] S. Riis, "Linear versus non-linear boolean functions in network flow," preprint, November 2003.

[200] J. R. Roche, R. W. Yeung, and K. P. Hau, "Symmetrical multilevel diversity coding," *IEEE Trans. Inform. Theory*, vol. IT-43, pp. 1059–1064, 1997.

[201] C. E. Shannon, "A mathematical theory of communication," *Bell Sys. Tech. Journal*, vol. 27, pp. 379–423, 623–656, 1948.

[202] R. C. Singleton, "Maximum distance Q-nary codes," *IEEE Trans. Inform. Theory*, vol. IT-10, pp. 116–118, 1964.

[203] L. Song, R. W. Yeung, and N. Cai, "Zero-error network coding for acyclic networks," *IEEE Trans. Inform. Theory*, vol. IT-49, pp. 3129–3139, 2003.

[204] A. L. Toledo and X. Wang, "Efficient multipath in sensor networks using diffusion and network coding," in *40th Annual Conference on Information Sciences and Systems*, (Princeton University, NJ, USA), March 22–24 2006.

[205] S. B. Wicker, *Error control systems for digital communication and storage*. 1995.

[206] R. W. Yeung, "Avalanche: A network Coding Analysis," preprint.

[207] R. W. Yeung, "Multilevel diversity coding with distortion," *IEEE Trans. Inform. Theory*, vol. IT-41, pp. 412–422, 1995.

[208] R. W. Yeung, "A framework for linear information inequalities," *IEEE Trans. Inform. Theory*, vol. IT-43, pp. 1924–1934, 1997.

[209] R. W. Yeung, *A first course in information theory*. Kluwer Academic/Plenum Publishers, 2002.

[210] R. W. Yeung and N. Cai, "Network Error Correction, Part I, Basic Concepts and Upper Bounds," to appear in *Communications in Information and Systems*.

[211] R. W. Yeung and Z. Zhang, "Distributed source coding for satellite communications," *IEEE Trans. Inform. Theory*, vol. IT-45, pp. 1111–1120, 1999.

[212] R. W. Yeung and Z. Zhang, "On symmetrical multilevel diversity coding," *IEEE Trans. Inform. Theory*, vol. IT-45, pp. 609–621, 1999.

[213] Z. Zhang, "Network Error Correction Coding in Packetized Networks," submitted to *IEEE Trans. Inform. Theory*.

[214] Z. Zhang and R. W. Yeung, "A non-shannon-type conditional inequality of information quantities," *IEEE Trans. Inform. Theory*, vol. IT-43, pp. 1982–1986, 1997.

[215] Z. Zhang and R. W. Yeung, "On characterization of entropy function via information inequalities," *IEEE Trans. Inform. Theory*, vol. IT-44, pp. 1440–1452, 1998.