# On the Relation between Linear Dispersion and Generic Network Code

**KWOK Pui Wing**

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Degree of
Master of Philosophy
in
Information Engineering

©The Chinese University of Hong Kong
August 2006

The Chinese University of Hong Kong holds the copyright of this thesis. Any person(s) intending to use a part or whole of the materials in the thesis in a proposed publication must seek copyright release from the Dean of the Graduate School.


11 OCT 2007
UNIVERSITY
LIBRARY SYSTEM
THE CHINESE UNIVERSITY OF HONG KONG

Abstract of thesis entitled:

On the Relation between Linear Dispersion and Generic Network Code

Submitted by KWOK Pui-Wing

for the degree of Master of Philosophy

at The Chinese University of Hong Kong in June 2006

Network coding is one of the most popular research topics in recent years. It refutes the folklore that the capacity of a network can be achieved by routing alone. Rather, coding may need to be employed at the intermediate nodes. It have been proved that linear network codes are sufficient for achieving the network capacity when there is only one source.

Different classes of linear network codes have been discussed in the literature, namely multicast, broadcast, dispersion and generic network code, where each of these classes is strictly stronger than the preceding one. In this thesis, we establish a nontrivial relation between linear dispersion and generic linear network code.

# 摘 要

網絡編碼是當今最熱門的研究項目之一。在網絡編碼出現之前，人們普遍認爲單單使用路由便能達到網絡的最大容量，但是網絡編碼反駁這個想法，網絡中的節點可能需要進行編碼以令網絡的容量得以充分利用。已有證明指出，當網絡中只有一個源節點的時候，線性編碼已經足以令網絡達到最大容量。

在已有的文獻中，已經有人提出了不同類別的線性網絡碼，即多重播送 (multicast)，廣播送 (broadcast)，分散播送 (dispersion) 及通用網絡碼 (generic network code)。他們還證明了這四類網絡碼分別都比其前者強。在這篇論文中，我們將會建立線性分散播送與通用網絡碼之間的非平凡關係。

# Acknowledgement

I would like to express my sincere gratitude towards my supervisor, Professor Raymond W. Yeung for his continuous guidance in this work. This work would not have been completed without his useful inputs and comments. He has consistently helped me in reviewing my work and consolidating my knowledge on network coding.

I would also like to thank the fellow graduate students in my lab. Their company added much joy and pleasure to my postgraduate life. Last, but not least, I have to thank my parents and my families for taking good care of me all these years. And I have to thank my girlfriend, Michelle, for her consistent endurance, care and love.

# Contents

**Abstract** i

**Abstract (Chinese Version)** ii

**Acknowledgement** iii

**1 Introduction** 1

**2 Linear Network Coding** 7
- 2.1 Single Source Network Coding . . . . . 8
- 2.2 Descriptions of Linear Network Codes . . . . . 9
- 2.3 Desirable Properties of Linear Network Codes . . 12
- 2.4 Linear Network Codes Constructions . . . . . 14

**3 Node-based Characterization** 16
- 3.1 Channel-based characterization . . . . . 16
- 3.2 A Necessary Condition for the Existence of Linear Network Codes . . . . . 17
- 3.3 Insufficiency of the condition . . . . . 22

**4 Relation between Linear Network Codes** 25
- 4.1 Relation between Multicast and Broadcast . . . . 26
  - 4.1.1 Auxiliary Graph . . . . . 26
- 4.2 Relation between Broadcast and Dispersion . . . 29
  - 4.2.1 Expanded Graph . . . . . 29

4.3 Relation between Dispersion and Generic Network Code . . . . . 31
4.3.1 Edge Disjoint Path . . . . . 31
4.3.2 Path Rearrangement . . . . . 34
4.3.3 Extended Graph . . . . . 50

**5 Upper Bound on the Size of the Base Field** **57**
5.1 Base Field Size Requirement . . . . . 58
5.1.1 Linear Multicast . . . . . 58
5.1.2 Linear Broadcast . . . . . 58
5.1.3 Linear Dispersion . . . . . 59
5.1.4 Generic Network Code . . . . . 60
5.2 Upper Bounds Comparison for Generic Network Code . . . . . 61

**6 Future Work** **62**

**Bibliography** **66**

# List of Figures


1.1 Multicasting over an acyclic network . . . . . 3

2.1 A message $x = [b_1\ b_2]$ of two bits $b_1$ and $b_2$ is multicast over the network . . . . . 11

3.1 There are $p$ nodes in $h_1(t)$. . . . . 21
3.2 Insufficiency of node-based characterization. . . . 23

4.1 An example of auxiliary graph. The dotted nodes and edges are added to form the auxiliary graph. 27
4.2 An example of expanded graph. The dotted nodes and edges are added. . . . . 30
4.3 There are two choices of choosing the edge-disjoint paths. . . . . 33
4.4 In the figures, the thick edges form two $A$-paths and the dotted edges form two $B$-paths. The edge $e$ cannot exist if $c_i$ and $c_{i'}$ are critical. . . . . 37
4.5 An example of extended graph. The nodes $u_1, u_2, u_3$ and $u_4$ are inserted and the edges are segmented into two. . . . . 51
4.6 We are given a dispersion on $G'$ . . . . . 52

6.1 Nodes $x$, $y$ and $w$ have maximum flow being one. 63
6.2 Re-design of network code to let node $w$ receive $b_1$. 64

# List of Tables

5.1 The upper bound of linear network codes. . . . . 60

# Chapter 1

# Introduction


**Summary**

This chapter aims to give a brief overview on the theory of network coding. Network coding suggests that when information is multicast in a point-to-point network, maximum possible throughput can be achieved if coding is allowed at the intermediate nodes within the network.

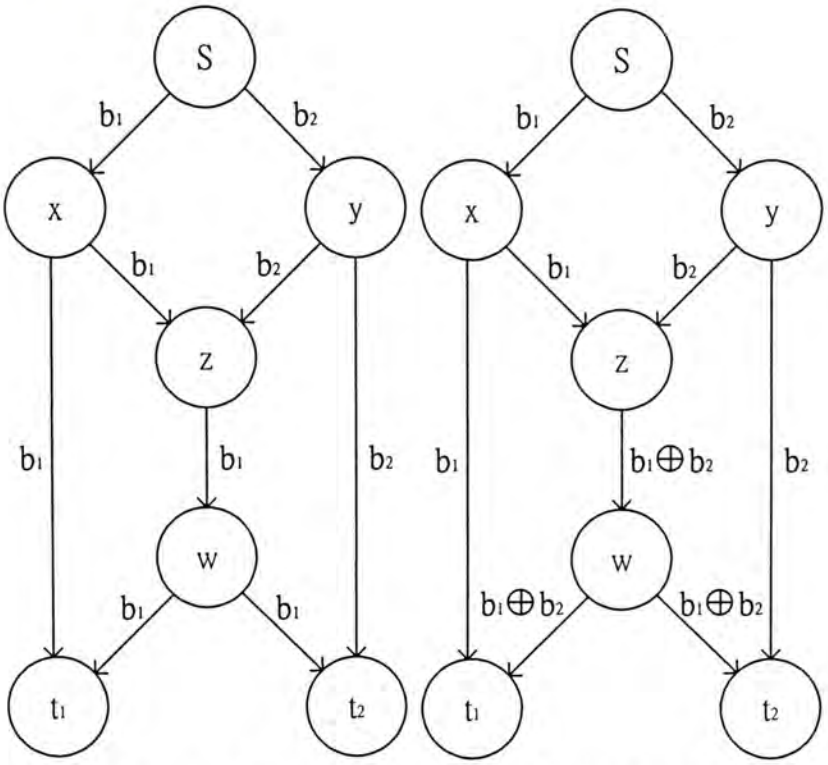

The notion of network coding was first introduced in [1]. Since its emergence, network coding has been become one of the hottest research topics [3][7][8]. We model a communication network by a finite directed graph $G = (V, E)$ where multiple edges from one node to another are allowed. Each vertex $v \in V$ represents a communication node in the network, which may be

a personal computer, a router or a switch. The edges between the vertices are the channels between the nodes through which information is sent. Some of the nodes are designated as source nodes from which the message is generated and sent out. The capacity of direct transmission from a node to its neighbour is determined by the multiplicity of channels between them. In addition, a communication network is said to be acyclic if it contains no directed cycles. The network in Figure 1.1(a) is an example of acyclic networks.

Consider a communication network as shown in Figure 1.1(a) [9]. There is one source node $s$, which generates two data bits $b_1$ and $b_2$. These two bits are to be multicast from source node $s$ to the sink nodes $t_1$ and $t_2$. All the channels in the network are of capacity 1. If only store-and-forward is allowed, i.e., every intermediate node is only allowed to send out replica of the bits it received, every channel would carry either bit $b_1$ or bit $b_2$ as shown in Figure 1.1(a). It is easy to check that a multicast rate of only 1.5 bits per unit time can be achieved.

If the intermediate nodes are allowed to re-encode the bits it received, as shown in Figure 1.1(b), a higher rate can be achieved. Instead of forwarding the bits it received, node $z$ de-

rives a new bit $b_1 \oplus b_2$, the exclusive-OR of $b_1$ and $b_2$, and sends it out. Node $w$ forwards the bit it received to nodes $t_1$ and $t_2$. Both sink nodes $t_1$ and $t_2$ can decode both $b_1$ and $b_2$ from the bits they received.




(a) No coding is allowed (b) Coding is allowed

Figure 1.1: Multicasting over an acyclic network

In a general setting, a communication network is a directed graph $G = (V, E)$, allowing multiple edges from one node to another. Every edge represents a communication channel with capacity of one data unit per unit time. Throughout this thesis, the source node will be denoted by $s$. For every node $t \in V$,

let $In(t)$ denote the set of incoming channels to $t$ and $Out(t)$ the set of outgoing channels from $t$. In particular, let $In(s)$ 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 $\omega$. Furthermore, for any two nodes $t_1, t_2 \in V$, the number of channels from $t_1$ to $t_2$ is denoted by $C(t_1, t_2)$.

A data unit is represented by an element of a certain base field $F$. The message generated at the source node $s$ consists of $\omega$ data units and therefore is represented by an $\omega$-dimensional row vector $x \in F^{\omega}$.

A non-source node does not necessarily receive all the information symbols to identify the value of the whole message $x$. Its encoding function simple maps the ensemble of received symbols from all its incoming channels to a symbol for each outgoing channel. A network code is specified by such an encoding mechanism for every node [10].

**Definition 1.1.** *[Local description of a network code]* Let $F$ be a finite field and $\omega$ be a positive integer. An $\omega$-dimensional $F$-valued network code on an acyclic communication network

consists of a local encoding mapping

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

for each node $t$ in the network and each channel $e \in Out(t)$.

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 $e$. It implies another equivalent definition of a network code:

**Definition 1.2.** *[Global description of a network code]* Let $F$ be a finite field and $\omega$ be a positive integer. An $\omega$-dimensional $F$-valued network code on an acyclic communication network consists of a local encoding mapping

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

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

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

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

2. For the $\omega$ imaginary channels $e$, the mappings $\widetilde{f}_e$ are the natural projections from the space $F^\omega$ to the $\omega$ different coordinates respectively.

---

□ **End of chapter.**

# Chapter 2

# Linear Network Coding


**Summary**

This chapter aims to focus on linear network coding. It has been proved in [8] that linear coding is sufficient to achieve the maximum possible rate when there is only one single source. It has further been discussed in [10] that there are four levels of strength of linear network codes, namely multicast, broadcast, dispersion and generic network code. It was proved in [10] that each of these four codes is strictly stronger than the preceding ones. Code construction algorithms for some of them will be discussed at the end of this chapter.

## 2.1 Single Source Network Coding

Given a communication network $G = (V, E)$, at least one node $s \in V$ is designated as the source node. The message $x$ is generated at the source node(s) and sent out. When there is only one source, we will refer to it as single source network coding.

It has been proved in [8] that in case of single source network coding, linear coding is sufficient to achieve the maximum possible rate. By saying linear coding, we constrain the nodes to perform only linear operations over some finite field, i.e. addition and scalar multiplication, on the symbols they received. Therefore, each node receives some information symbols from its incoming links and sends out some linear combinations of those symbols on its outgoing links. In particular, routing can be considered as a special case of linear network coding.

Furthermore, it has been proved in [8] that the maximum possible rate of sending information from source node $s$ to a non-source node $t$ is determined by the maximum flow from $s$ to $t$. We denote this quantity by $maxflow(t)$. The definitions of flow and maximum flow are as follows [5]:

**Definition 2.1.** Let $G = (V, E)$ be a directed graph and $s, t \in V$ where $s$ is the source node and $t$ is a sink node. A flow from $s$ to $t$ is a function $\mathcal{F} : E \rightarrow \mathcal{R}$ which obeys three types of constraints:

1. $\mathcal{F}(e) \leq C(e)$ for all $e \in E$,
2. $\Sigma_{d \in In(v)} \mathcal{F}(d) = \Sigma_{e \in Out(v)} \mathcal{F}(e)$ for all $v \in V \setminus \{s, t\}$,
3. $\mathcal{F}(e) \geq 0$ for all $e \in E$.

The value of a flow $\mathcal{F}$ is defined as $\Sigma_{e \in In(t)} \mathcal{F}(e)$. The maximum value of all possible flows is the maximum flow from $s$ to $t$, denoted by $maxflow(t)$.

## 2.2 Descriptions of Linear Network Codes

As discussed in the previous chapter, a network code can be specified either by a local or global description. For a linear network code, the local and global description can be simplified as follows [10]:

**Definition 2.2.** *[Local description of a linear network code]* Let $F$ be a finite field and $\omega$ be a positive integer. An $\omega$-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 of edge $(d, e)$ in the network. Meanwhile, the local encoding kernel at the node $t$ means the $|In(t)| \times |Out(t)|$ matrix $K_t = [k_{d,e}]_{d \in In(t), e \in Out(t)}$.

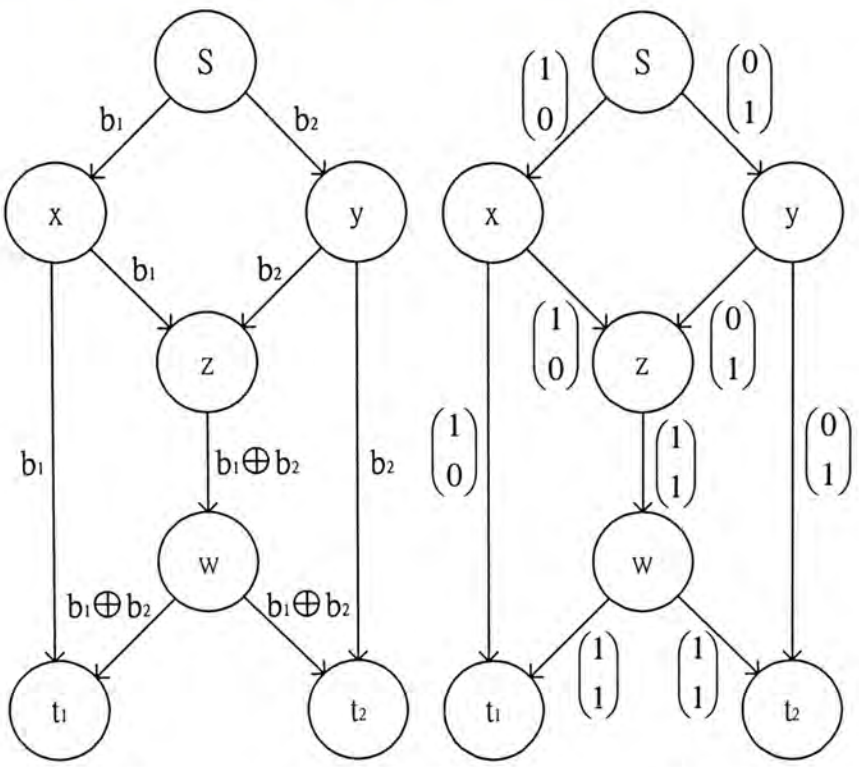
**Definition 2.3.** *[Global description of a linear network code]* Let $F$ be a finite field and $\omega$ be a positive integer. An $\omega$-dimensional $F$-valued linear network code on an acyclic communication network consists of a scalar $k_{d,e}$ for every adjacent pair of edge $(d, e)$ in the network as well as an $\omega$-dimensional column vector $f_e \in F^\omega$ for every channel $e$ such that:

1. $f_e = \sum_{d \in In(t)} k_{d,e} f_d$, where $e \in Out(t)$.
2. The vectors $f_e$ for the $\omega$ imaginary channels $e \in In(s)$ form the natural basis of the vector space $F^\omega$.

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

The message is denoted by $x$, which is a $\omega$-dimensional row vector, and the global encoding kernel for a particular channel $e$ is $f_e$, which is a $\omega$-dimensional column vector. Therefore, the

actual information symbol sent over channel $e$ is $x \cdot f_e$. A message $x = [b_1\ b_2]$ of two bits $b_1$ and $b_2$ is multicast over the network in Figure 2.1. The actual symbols sent over the channels and the global encoding kernels for the channels are shown.




(a) Actual symbols sent over the network

(b) Global encoding kernels of each channel

Figure 2.1: A message $x = [b_1\ b_2]$ of two bits $b_1$ and $b_2$ is multicast over the network

## 2.3 Desirable Properties of Linear Network Codes

From the global description of linear network codes, we can see that a particular node $t \in V$ can receive all the information symbols, and hence decode the source message $x$, if the global encoding kernels of all its incoming channels span the whole vector space $F^\omega$. In other words, node $t$ can decode the message $x$ from its received symbols if

$$\langle \{ f_d : d \in In(t) \} \rangle = F^\omega.$$

In the above expression, we adopt the notation that $\langle \cdot \rangle$ represents the linear span of a set of vectors.

Given a node $t \in V$, denote $V_t$ by

$$V_t = \langle f_d : d \in In(t) \rangle .$$

This vector subspace $V_t$ can be regarded as the "knowledge" known by node $t$. The degree of such "knowledge" can be used to classify linear network nodes into four levels of strength [10]. As suggested in [10], a linear network code qualifies as a multicast, broadcast or dispersion, respectively if the following conditions hold:

1. $dim(V_t) = \omega$ for every non-source node $t$ with $maxflow(t) \geq \omega$.

2. $dim(V_t) = min\{\omega, maxflow(t)\}$ for every non-source node $t$.

3. $dim(\langle \cup_{t \in \mathcal{P}} V_t \rangle) = min\{\omega, maxflow(\mathcal{P})\}$ for every collection $\mathcal{P}$ of non-source nodes.

Furthermore, a linear network code qualifies as a generic network code if the following condition is satisfied:
For an arbitrary set of channels $\{e_1, e_2, \ldots, e_m\}$, where each $e_j \in Out(t_j)$, the vectors $f_{e_1}, f_{e_2}, \ldots, f_{e_m}$ are linearly independent (and hence $m \leq \omega$) provided that

$$\langle \{f_d : d \in In(t_j)\} \rangle \not\subset \langle \{f_{e_k} : k \neq j\} \rangle$$

for $1 \leq j \leq m$.

It has been pointed out in [10] that a dispersion is necessarily a broadcast and a broadcast is necessarily a multicast. In [10], it has also been proved that a generic network code is necessarily a dispersion. However, the converse is not true.

**Theorem 2.1.** *Every generic linear network code is a linear dispersion.*

*Proof.* The proof can be found in [10]. □

## 2.4 Linear Network Codes Constructions

Several algorithms have been proposed to construct a linear network code on a given network [8][10][6]. Li et al. first proposed a construction algorithm in [8]. Given a positive integer $\omega$ and an acyclic network $G = (V, E)$, the algorithm is able to construct a generic network code on $G$, which is the strongest code, with certain requirement on the size of the base field. Jaggi et al. and Sanders et al. gave a refined version of the algorithm [6]. The refined algorithm would require a smaller base field and of less computational complexity. However, the refined algorithm was only capable to construct a multicast, which is the weakest code.

The algorithm proposed by Li et al. [8] constructs a generic linear network code by assigning a vector to each channel. It is proved in [10] that a base field $F$ is sufficient for this algorithm if

$$|F| > \begin{pmatrix} |E| + \omega - 1 \\ \omega - 1 \end{pmatrix}.$$

Meanwhile, the refined algorithm proposed by Jaggi et al. and Sanders et al. only requires the base field $F$ to have more than $\eta$

elements, where $\eta$ is the number of receiver nodes with $maxflow$ bigger than or equal to $\omega$, i.e.,

$$|F| > |\{t \in V : maxflow(t) \geq \omega\}|.$$

---

□ **End of chapter.**

# Chapter 3

# Node-Based Characterization of Linear Network Codes


**Summary**

It has been suggested in [10] that a linear network code can be specified by an assignment of $\omega$-dimensional vectors to the channels. In this chapter, we give a necessary condition from a node-based perspective for the existence of a linear network code.


## 3.1 Channel-based characterization

As discussed in the previous chapter, a linear network code has both local and global descriptions. We will focus on global description in this chapter. Let $F$ be a finite field and $\omega$ be a

positive integer. The global description of an $\omega$-dimensional $F$-valued linear network code on an acyclic communication network $G = (V, E)$ can be interpreted as an assignment of $\omega$-dimensional vector $f_e \in F^\omega$ to each channel $e \in E$ such that the actual information sent over channel $e$ is $x \cdot f_e$, where $x$ is an $\omega$-dimensional row vector representing the message.

In other words, we can construct a linear network code by assigning vectors inside the vector space $F^\omega$ to the channels subject to one constraint: For every node $t \in V$ and its outgoing channel $e \in Out(t)$,

$$f_e \in \langle f_d : d \in In(t) \rangle .$$

This constraint is to abide the law of information flow.

## 3.2 A Necessary Condition for the Existence of Linear Network Codes

Besides considering linear network codes from an assignment of vectors to the edges, we can view a linear network code from a node-based perspective. Let the vector subspace known by a node $t \in V$ be $V_t$, i.e.,

$$V_t = \langle f_d : d \in In(t) \rangle .$$

In particular, for the source node $s$, we have

$$V_s = F^{\omega}.$$

Furthermore, define two functions, $h_1$ and $h_2$, from $V$ to the power set of $V, \mathcal{P}(V)$, such that $u \in h_1(t) \subset V$ if there exists an edge from $u$ to $t$ and $v \in h_2(t) \subset V$ if there exists an edge from $t$ to $v$. In particular, we define

$$h_1(s) = \emptyset.$$

**Lemma 3.1.** For any finite acyclic directed graph $G = (V, E)$, there exists at least one node $a \in V$ such that $h_1(a) = \{s\}$, where $s \in V$ denotes the source node.

*Proof.* As the graph is connected, we can assume $h_1(a) \neq \emptyset$ for all $a \in V \setminus \{s\}$. Assume the contrary, i.e.,

$$\{a : h_1(a) = \{s\}\} = \emptyset.$$

Then fix a particular node $a_1 \in V$. Since $h_1(a_1) \neq \{s\}$, we can always find another node $a_2 \in h_1(a_1) \setminus \{s\}$. Similarly we can find another node $a_3 \in h_1(a_2) \setminus \{s\}$. By continuing the process, we can obtain a sequence of nodes $\{a_i\}$, where

$$a_{i+1} \in h_1(a_i) \setminus \{s\}.$$

Since there is a finite number of nodes, the set of all $a_i$'s cannot be infinite. Hence, there exists two distinct positive integer $j$ and $j'$ such that $a_j = a_{j'}$. Note that the reverse of the sequence $\{a_i\}$ actually defines a path in the graph. Therefore the fact that $a_j = a_{j'}$ implies a directed cycle in the graph, contradicting the assumption that the graph is acyclic. Thus we can conclude that

$$\{a : h_1(a) = \{s\}\} \neq \emptyset.$$

□

**Theorem 3.1.** Given a linear network code specified by $f_e$ for all $e \in E$ and for each node $t \in V \setminus \{s\}$, let

$$h_1(t) = \{u_1, u_2, \ldots, u_p\}$$

and

$$C(u_i, t) = n_i$$

where $C(u_i, t)$ denotes the capacity of edge $(u_i, t)$. Then

$$V_t \subset \langle \{V_{u_1}, V_{u_2}, \ldots, V_{u_p}\} \rangle$$

and for any subset $\mathcal{I} \subset \{1, 2, \ldots, p\}$,

$$\sum_{i \in \mathcal{I}} n_i \geq dim(V_t) - dim\left(V_t \cap \langle \cup_{j \notin \mathcal{I}} V_{u_j} \rangle\right).$$

In particular, if we set $\mathcal{I}$ to be $\{1, 2, \ldots, p\}$, then we would obtain

$$dim(V_t) \leq \sum_{i=1}^{p} n_i.$$

*Proof.* By Lemma 3.1, there exists at least one node $a \in V$ such that

$$h_1(a) = \{s\}.$$

Obviously, as

$$V_s = F^{\omega},$$

we have

$$V_a \subset V_s.$$

Moreover, since

$$V_a = \{f_d : d \in In(a)\},$$

we have

$$dim(V_a) \leq C(s, a).$$

Hence the condition holds for all nodes $a \in V$ with $h_1(a) = \{s\}$.

For a general node $t \in V$ in the network, let

$$h_1(t) = \{u_i : i = 1, 2, \ldots, p\}$$

for some positive integer $p$, and

$$C(u_i, t) = n_i$$

where $n_i$ is a positive integer for $i = 1, 2, \ldots, p$ (Figure 3.1).

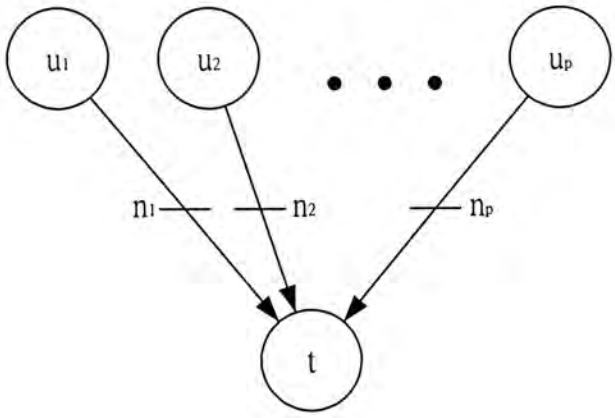



Figure 3.1: There are $p$ nodes in $h_1(t)$.

For any integer $i = 1, 2, \ldots, p$, let

$$V_i = \langle \{ f_e : e \in Out(u_i) \cap In(t) \} \rangle .$$

Then,

$$V_i \subset V_{u_i}$$

and

$$V_t = \left\langle \bigcup_i V_i \right\rangle .$$

Hence,

$$V_t \subset \left\langle \bigcup_i V_{u_i} \right\rangle .$$

Also, we have

$$dim(V_i) \leq n_i .$$

Moreover, for any subset $\mathcal{I} \subset \{1, 2, \ldots, p\}$,

$$\begin{aligned} dim(V_t) &= dim\left(\langle \cup_i V_i \rangle \cap V_t\right) \\ &= dim\left(\left(\langle \cup_{i\in\mathcal{I}} V_i \rangle \cap V_t\right) \cup \left(\langle \cup_{j\notin\mathcal{I}} V_j \rangle \cap V_t\right)\right) \\ &\leq dim\left(\langle \cup_{i\in\mathcal{I}} V_i \rangle \cap V_t\right) + dim\left(\langle \cup_{j\notin\mathcal{I}} V_j \rangle \cap V_t\right) \\ &= dim\left(\langle \cup_{i\in\mathcal{I}} V_i \rangle\right) + dim\left(\langle \cup_{j\notin\mathcal{I}} V_j \rangle \cap V_t\right) \\ &\leq \sum_{i\in\mathcal{I}} n_i + dim\left(\left\langle \cup_{j\notin\mathcal{I}} V_{u_j} \right\rangle \cap V_t\right). \end{aligned}$$

Therefore,

$$\sum_{i\in\mathcal{I}} n_i \geq dim(V_t) - dim\left(\left\langle \cup_{j\notin\mathcal{I}} V_{u_j} \right\rangle \cap V_t\right).$$

□

## 3.3 Insufficiency of the condition

The condition given in the last section is a necessary but not sufficient condition for a linear network code. Consider the example in Figure 3.2. It is easy to check that the assigned vector subspaces to the nodes satisfy the condition mentioned in the previous section. Nevertheless, there exists no legitimate linear network code with such vector subspace assignment. Since node $z$ has to receive the vector subspace $V_z$, it must receive two linearly independent vectors, in which one of them must be

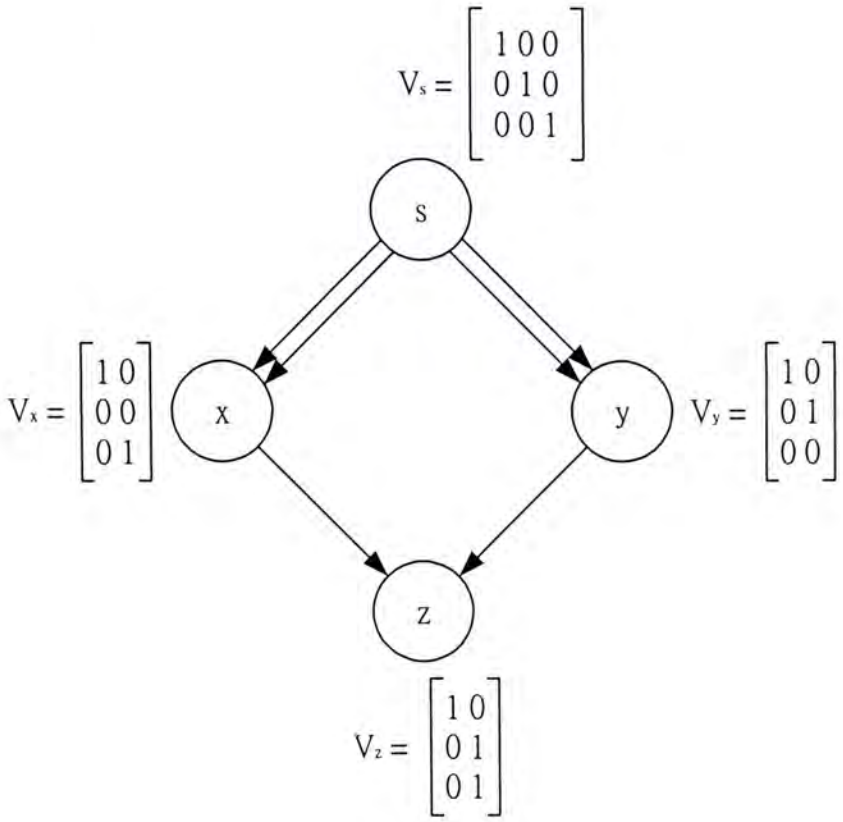



Figure 3.2: Insufficiency of node-based characterization.

received from node $x$ and the other must be received from node $y$. Observe that

$$V_x \cap V_z = \left\langle \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T \right\rangle$$

and

$$V_y \cap V_z = \left\langle \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T \right\rangle.$$

It means that nodes $x$ and $y$ must co-operate such that node $z$ can receive the vector $\begin{bmatrix} 0 & 1 & 1 \end{bmatrix}^T$. However, in doing so, the vector $\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T$ cannot be sent to node $z$. Hence, we cannot

find a linear network code which would result in such vector subspace assignment. Therefore, the condition is a necessary but not sufficient condition.

---

□ **End of chapter.**

# Chapter 4

# Relation between Various Classes of Linear Network Codes


**Summary**

This chapter aims to describe the relation between the four linear network codes as discussed in the previous chapter, namely multicast, broadcast, dispersion and generic network code. Though it has been proved in [10] that each of these four codes is strictly stronger than the preceding ones, we observed that certain transformation can be performed on the network $G$ such that a weaker network code on the transformed network $\overline{G}$ is equivalent to a stronger code on the original network $G$.

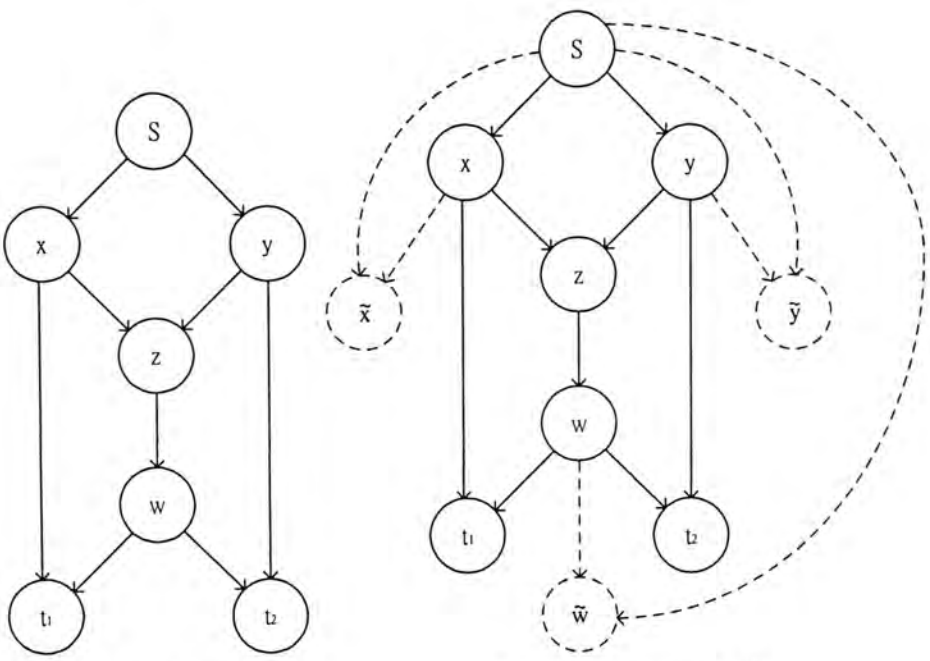
In [10], it has been proved that there exist four levels of linear network codes: multicast, broadcast, dispersion and generic network code. Each of these four codes is strictly stronger than the preceding ones. Nevertheless, it has also been pointed out in [10] that by transforming the original network $G$ into another network $\widetilde{G}$, a linear multicast on $\widetilde{G}$ incorporates a linear broadcast on $G$. Similarly, by installing new nodes and channels to the network $G$, a linear broadcast on the new network incorporates a linear dispersion on the original network. In this chapter, we further propose another transformation to establish the relation between linear dispersion and generic network code.

## 4.1 Relation between Multicast and Broadcast

### 4.1.1 Auxiliary Graph

For every acyclic digraph $G = (V, E)$, we derive an auxiliary graph $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ from $G$ through the following mechanism: Let $\omega$ be a positive integer. For every non-source node $t \in V$ where $maxflow(t) < \omega$, we install a new node $\tilde{t}$ and $\omega$ incoming channels to this new node, among which $maxflow(t)$ of them

are from $t$ and the remaining $\omega - maxflow(t)$ are from the source node $s$. This node has no outgoing channels. This newly constructed acyclic digraph is designated as the auxiliary graph $\widetilde{G}$ of the original graph $G$. An example of auxiliary graph is shown in Figure 4.1.




(a) Original graph $G$ (b) Auxiliary graph $\widetilde{G}$

Figure 4.1: An example of auxiliary graph. The dotted nodes and edges are added to form the auxiliary graph.

**Theorem 4.1.** A linear multicast on the auxiliary graph $\widetilde{G}$ incorporates a linear broadcast on the original graph $G$.

*Proof.* Let $\omega$ be a positive integer. Suppose we are given an

acyclic digraph $G = (V, E)$ and a linear multicast on its auxiliary graph $\widetilde{G} = (\widetilde{V}, \widetilde{E})$.

For every node $t \in V$, we consider two cases:

1. Case 1: If $maxflow(t) \geq \omega$, then by the property of the given linear multicast, we have

$$dim(V_t) = \omega.$$

2. Case 2: If $maxflow(t) < \omega$, then, by the construction of auxiliary graph $\widetilde{G}$, there exists a node $\tilde{t} \in \widetilde{V}$ such that there are $\omega$ incoming channels to $\tilde{t}$ among which $maxflow(t)$ of them are from $t$ and the remaining $\omega - maxflow(t)$ are from source node $s$. It is easy to check that

$$maxflow\left(\tilde{t}\right) = \omega.$$

Hence, by the property of the given linear multicast, we can conclude that

$$dim(V_{\tilde{t}}) = \omega.$$

Since there are only $\omega$ channels terminating at $\tilde{t}$, the global encoding kernels of each of the incoming channels must be linearly independent. Also, as $maxflow(t)$ of them are from $t$, we can conclude that

$$dim(V_t) \geq maxflow(t).$$

It has been proved in [8] that the maximum information rate is upper bounded by the maximum flow. Hence,

$$dim(V_t) \leq maxflow(t).$$

Therefore, we have

$$dim(V_t) = maxflow(t).$$

Combining both cases, we can conclude that for every node $t \in V$,

$$dim(V_t) = min\{\omega, maxflow(t)\}.$$

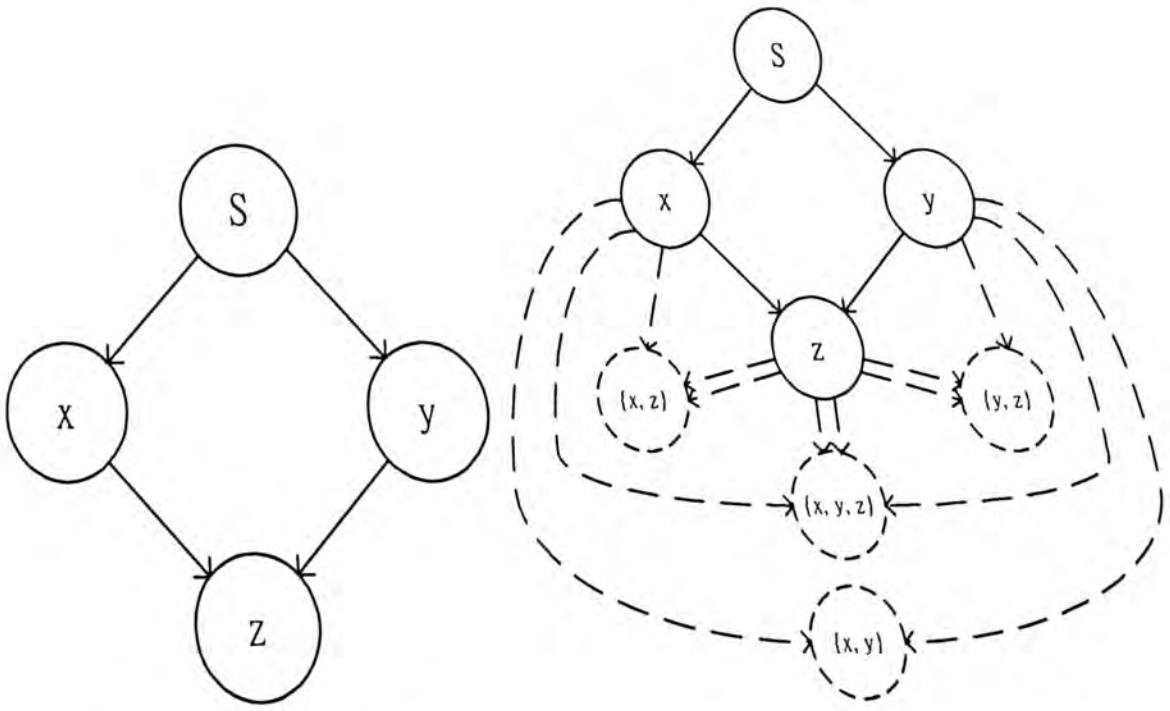
By removing the installed nodes $\tilde{t}$ and their incoming channels from $\widetilde{G}$, we can obtain a linear broadcast on $G$. □

## 4.2 Relation between Broadcast and Dispersion

### 4.2.1 Expanded Graph

For every acyclic digraph $G = (V, E)$, we expand this graph to form an expanded $\widehat{G} = (\widehat{V}, \widehat{E})$ from $G$ through the following mechanism: Let $\omega$ be a positive integer. For every non-empty collection $\mathcal{P}$ of non-source nodes where $|\mathcal{P}| > 1$, we install a new node $\hat{t}_{\mathcal{P}}$ and there exist $maxflow(\mathcal{P})$ incoming channels to this new node. This new node has no outgoing channels. For each

node $t \in \mathcal{P}$, there exist $maxflow(t)$ channels from $t$ to $\hat{t}_{\mathcal{P}}$. This newly constructed acyclic digraph is designated as the expanded graph $\widehat{G}$ of the original graph $G$. An example of expanded graph is shown in Figure 4.2.




(a) Original graph $G$ (b) Expanded graph $\widehat{G}$

Figure 4.2: An example of expanded graph. The dotted nodes and edges are added.

**Theorem 4.2.** A linear broadcast on the expanded graph $\widehat{G}$ incorporates a linear dispersion on the original graph $G$.

*Proof.* Let $\omega$ be a positive integer. Suppose we are given an acyclic digraph $G = (V, E)$ and a linear broadcast on its expanded graph $\widehat{G} = (\widehat{V}, \widehat{E})$.

For every collection of non-source nodes $\mathcal{P} \subset V$ such that

$|\mathcal{P}| > 1$, there exists a corresponding node $\hat{t}_{\mathcal{P}} \in \widehat{V}$. It is easy to check that

$$maxflow(\hat{t}_{\mathcal{P}}) = maxflow(\mathcal{P}).$$

If $|\mathcal{P}| > 1$, then, by the property of the given linear broadcast,

$$\begin{aligned} dim\left(V_{\hat{t}_{\mathcal{P}}}\right) &= maxflow(\hat{t}_{\mathcal{P}}) \\ &= maxflow(\mathcal{P}). \end{aligned}$$

Besides, for each $t \in V \subset \widehat{V}$, we have

$$dim\left(V_t\right) = maxflow(t).$$

Hence, for every collection of non-source nodes $\mathcal{P} \subset V$, we have

$$dim\left(\langle \cup_{t\in\mathcal{P}} V_t \rangle\right) = maxflow(\mathcal{P}).$$

After removing the installed nodes $\hat{t}_{\mathcal{P}}$ and their incoming channels, it would result a linear network code which is guaranteed to be a dispersion. □

## 4.3 Relation between Dispersion and Generic Network Code

### 4.3.1 Edge Disjoint Path

**Definition 4.1.** A *walk* [4] in a graph $G = (V, E)$, is a sequence of vertices and edges of $G$ of the form:

$$(v_1, (v_1, v_2), v_2, (v_2, v_3), v_3, \ldots, v_{n-1}, (v_{n-1}, v_n), v_n),$$

where $v_i \in V$ and $(v_{i-1}, v_i) \in E$ for $i = 1, 2, \ldots, n$. A walk is called a *path* if $v_i \neq v_j$ for $i \neq j$. For simplicity, a path can be written as a sequence of edges only.

Given a path $P = \{(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n)\}$ on a acyclic directed graph $G$, we define a relation $\prec_P$ on $P$ such that for any $(v_i, v_{i+1}), (v_j, v_{j+1}) \in P$, $(v_i, v_{i+1}) \prec_P (v_j, v_{j+1})$ if there exists a path in $G$ from $v_i$ to $v_j$. Likewise, we denote $(v_i, v_{i+1}) \succ_P (v_j, v_{j+1})$ if there exists a path in $G$ from $v_j$ to $v_i$. Obviously, this relation $\preceq_P$ is a partial ordering on the set of edges in path $P$.

We say two paths $P_1$ and $P_2$ intersect with each other if

$$P_1 \cap P_2 \neq \emptyset.$$

An edge $e$ is called a junction edge between two intersecting paths $P_1$ and $P_2$ if

$$e \in P_1 \cap P_2.$$

**Lemma 4.1.** For any acyclic digraph $G = (V, E)$, let $s \in V$ be the source node and $t \in V$ be an arbitrary node such that

$$maxflow(t) = n.$$

Then there exist $n$ edge-disjoint paths (EDP) from $s$ to $t$.

*Proof.* The proof can be found in [8]. □

By Lemma 4.1, although we are guaranteed to find $n$ edge-disjoint paths from source node $s$ to node $t$ if $maxflow(t) = n$, the choice of such $n$ edge-disjoint paths may not be unique. In Fig. 4.3, the maximum flow to node $t$ is two. There exist two edge-disjoint paths from $s$ to $t$. One of them must be the path $((s,x),(x,a),(a,t))$. For the other path, there are two choices, which are

$$((s,y),(y,b),(b,t))$$

and

$$((s,y),(y,c),(c,t)).$$

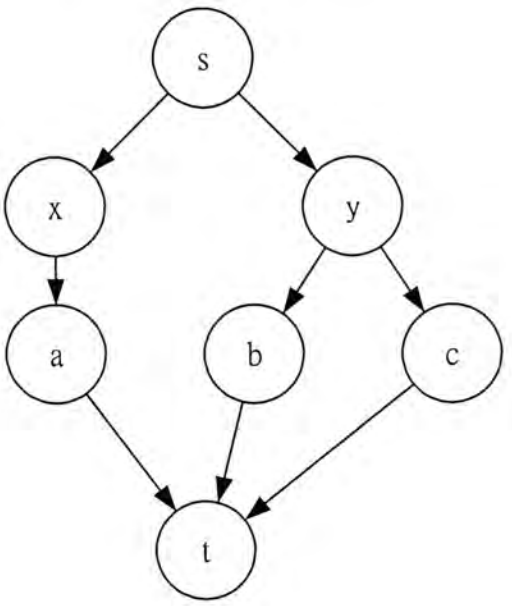



Figure 4.3: There are two choices of choosing the edge-disjoint paths.

If there is only one choice to choose such edge-disjoint paths, the graph is referred to as an unique graph [2]. Network coding on unique graphs has been studied in [2].

### 4.3.2 Path Rearrangement

For any acyclic digraph $G = (V, E)$, let $s$ be the source node and $A$ and $B$ be two sets of nodes such that

$$A \subset B \subset V.$$

Assume

$$maxflow(A) = \alpha,$$

and

$$maxflow(B) = \beta.$$

It is possible to find $\alpha$ EDP's from source node $s$ to $A$ and $\beta$ EDP's from $s$ to $B$. Let us assume $\beta > \alpha$. For simplicity, denote the $\alpha$ paths to $A$ by $P_1^A, P_2^A, \ldots, P_\alpha^A$ and the $\beta$ paths to $B$ by $P_1^B, P_2^B, \ldots, P_\beta^B$. We will refer to the paths $P_1^A, P_2^A, \ldots, P_\alpha^A$ as $A$-paths and $P_1^B, P_2^B, \ldots, P_\beta^B$ as $B$-paths.

**Lemma 4.2.** For any fixed $i = 1, 2, \ldots, \alpha$, there exists an integer $1 \leq j \leq \beta$ such that

$$P_i^A \cap P_j^B \neq \emptyset.$$

In other words, each $A$-path must intersect with some $B$-path(s).

*Proof.* Assume the contrary, i.e., for some $i$, $P_i^A$ is edge-disjoint with $P_j^B$ for all $j = 1, 2, \ldots, \beta$. Then the set of paths,

$$\{P_j^B : j = 1, 2, \ldots, \beta\} \cup \{P_i^A\},$$

accounts for $\beta + 1$ edge-disjoint paths from source node $s$ to the set $B$. Hence,

$$maxflow(B) = \beta + 1,$$

which is a contradiction. □

An edge on $P_i^A$ which is a junction edge between $P_i^A$ and $P_k^B$ for some $k$ will be referred to as a junction edge of $P_i^A$. Likewise, an edge on $P_j^B$ which is a junction edge between $P_j^B$ and $P_l^A$ for some $l$ will be referred to as a junction edge of $P_j^B$.

Suppose there are $K$ junction edges of $P_j^B$. Let this set be $h_j^B(1), h_j^B(2), \ldots, h_j^B(K)$ such that

$$h_j^B(\lambda) \prec_{P_j^B} h_j^B(\mu)$$

if $\lambda < \mu$. Let $\gamma_j(\cdot)$ be a function such that

$$h_j^B(k) \in P_{\gamma_j(k)}^A.$$

Since all $P^A$'s are edge-disjoint, $\gamma_j(k)$ is well-defined.

**Definition 4.2.** Consider the set of paths $\{P_i^A\}$ and $\{P_j^B\}$ as described above. A set of edges $\{c_i : i = 1, 2, \ldots, \alpha\}$ is a set of critical edges of $\{P_i^A\}$ with respect to $\{P_j^B\}$ if the following conditions are satisfied:

1. For each $i$, $c_i \in P_i^A \cap P_j^B$ for some $j = 1, 2, \ldots, \beta$.

2. There exists no integer $j$ such that $c_i, c_{i'} \in P_j^B$ for $i \neq i'$.

3. For $i \neq i'$ where

$$c_i \in P_i^A \cap P_j^B$$

and

$$c_{i'} \in P_{i'}^A \cap P_{j'}^B,$$

then there exists no edge $e \in P_i^A \cap P_{j'}^B$ such that $c_{i'} \succ_{P_{j'}^B} e \succ_{P_i^A} c_i$.

In condition 3 above, by symmetry, we can exchange the roles of $i$ and $i'$ so that there also exists no $e \in P_{i'}^A \cap P_j^B$ such that $c_i \succ_{P_j^B} e \succ_{P_{i'}^A} c_{i'}$. This is illustrated in Fig. 4.4.

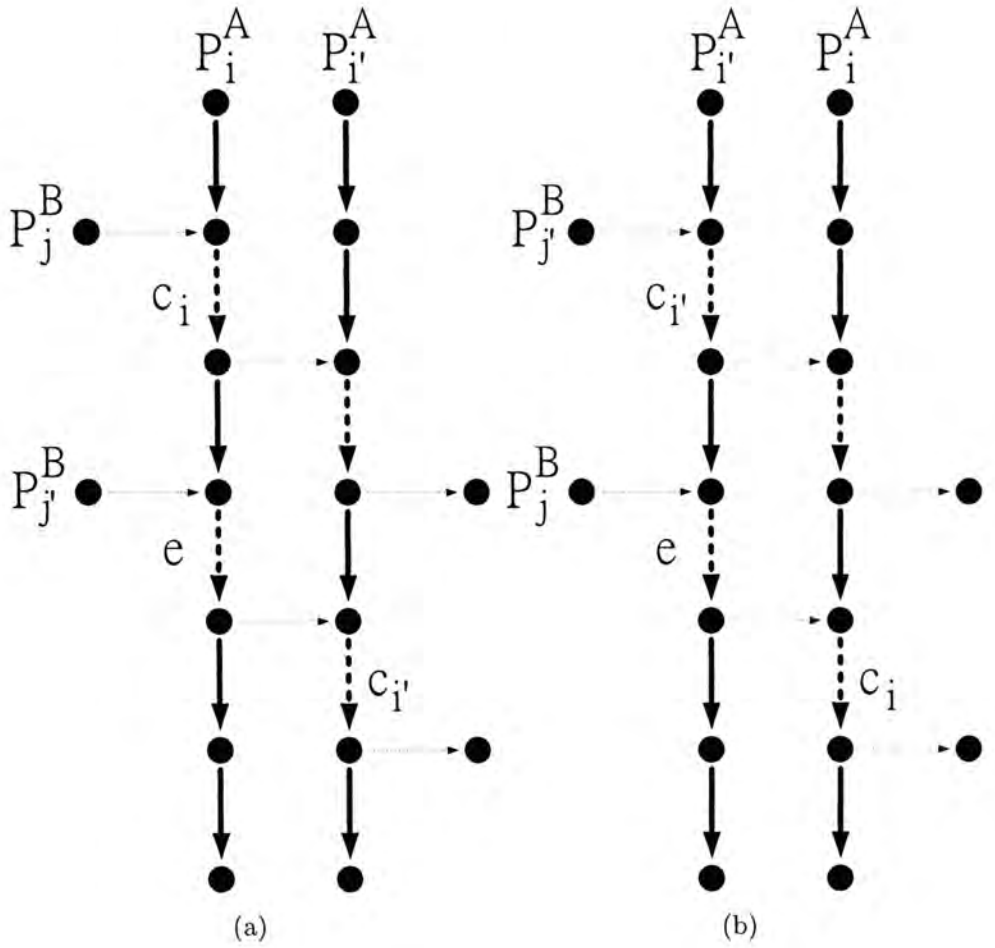



Figure 4.4: In the figures, the thick edges form two $A$-paths and the dotted edges form two $B$-paths. The edge $e$ cannot exist if $c_i$ and $c_{i'}$ are critical.

**Lemma 4.3.** Consider the set of paths $\{P_i^A\}$ and $\{P_j^B\}$ as described above. Then there exists a set of critical edges of $\{P_i^A\}$ with respect to $\{P_j^B\}$.

This lemma will be proved by an implicit construction of the set of edge $\{c_i\}$ in Algorithm 4.1.

**Algorithm 4.1.** The following algorithm prescribes a mechanism to find a set of critical edges $c_i$'s for each $P_i^A$.

{

*for* $(i = 1; i \leq \alpha; i++)$

{

$n_i := 1;$

$c_i :=$ *the* $n_i$*-th most downstream junction edge of* $P_i^A$*;*

}

*//This is just the initialization and the* $c_i$*'s will be updated.*

*while* ($\exists$ *a pair* $(c_i, c_j)$ *on some path* $P_k^B$)

{

*if* $(c_j \succ_{P_k^B} c_i)$

{

$n_j++;$

$c_j :=$*the* $n_j$*-th most downstream junction edge of* $P_j^A$*;*

} *else* {

$n_i++;$

$c_i :=$*the* $n_i$*-th most downstream junction edge of* $P_i^A$*;*

}

}

}

For the time being, let us assume that the algorithm always terminates and denote the terminal value of $c_i$ by $c_i^*$ for $1 \leq i \leq \alpha$.

**Lemma 4.4.**

$$c^*_{\gamma_j(1)} \succeq_{P^A_{\gamma_j(1)}} h^B_j(1).$$

*Proof.* Assume the contrary, i.e.,

$$c^*_{\gamma_j(1)} \prec_{P^A_{\gamma_j(1)}} h^B_j(1).$$

According to the algorithm, $c_{\gamma_j(1)}$ must equal $h^B_j(1)$ at some point. As another edge is subsequently assigned to $c_{\gamma_j(1)}$, there must exist a junction edge $e$ of $P^B_j$ such that

$$e \prec_{P^B_j} h^B_j(1).$$

By the definition of $h^B_j(1)$, such a junction edge $e$ cannot exist, and this leads to a contradiction. □

**Lemma 4.5.** For all $k \geq 2$, if

$$c^*_{\gamma_j(k)} \prec_{P^A_{\gamma_j(k)}} h^B_j(k),$$

then

$$c^*_{\gamma_j(l)} = h^B_j(l)$$

for some $l < k$.

*Proof.* We will prove the lemma by induction on $k$. For $k = 2$, i.e., if

$$c^*_{\gamma_j(2)} \prec_{P^A_{\gamma_j(2)}} h^B_j(2),$$

we will consider three cases:

Case I:

$$c^*_{\gamma_j(1)} = h^B_j(1).$$

The proof is immediate.

Case II:

$$c^*_{\gamma_j(1)} \prec_{P^A_{\gamma_j(1)}} h^B_j(1).$$

This is impossible as is proved in Lemma 4.4.

Case III:

$$c^*_{\gamma_j(1)} \succ_{P^A_{\gamma_j(1)}} h^B_j(1).$$

From the given condition, $c_{\gamma_j(2)}$ must equal $h^B_j(2)$ at some point and later be re-assigned. Nevertheless, in this case such a re-assignment cannot happen.

Combining three cases, the lemma is seen to be true for $k = 2$.

Assume it is true for some $k \geq 2$. Given that

$$c^*_{\gamma_j(k+1)} \prec_{P^A_{\gamma_j(k+1)}} h^B_j(k+1),$$

then $c_{\gamma_j(k+1)}$ must equal $h_j^B(k+1)$ at some point and be re-assigned later. There are two possible cases:

Case 1:

$$c^*_{\gamma_j(l)} \succ_{P_j^B} h_j^B(l)$$

for all $l < k+1$. This is obviously impossible because if so, $c_{\gamma_j(k+1)}$ cannot be re-assigned.

Case 2: There exists some integers $l' < k+1$, such that

$$c^*_{\gamma_j(l')} \preceq_{P_j^B} h_j^B(l').$$

Let $l_m$ be the smallest such $l'$. If

$$c^*_{\gamma_j(l_m)} = h_j^B(l_m),$$

then the proof is done. Otherwise, by Lemma 4.4,

$$l_m \geq 2.$$

Then by induction hypothesis,

$$c^*_{\gamma_j(l)} = h_j^B(l)$$

for some $l < l_m < k+1$. This completes the proof of the lemma.

□

**Justification of Algorithm 4.1.** We first need to show that the algorithm will terminate at some point.

We are going to illustrate the algorithm using a table as follows. There are $\alpha$ rows in the table, where each row corresponds to a path $P_i^A$. Every time the while-loop is executed, the $c_i$'s are assigned with some values that forms a new column of the table on the right. The algorithm will end when all the entries on the last column are on different $P_j^B$'s.

| | | | |
|---|---|---|---|
| $P_1^A$ | $e_1$ | $e_1$ | $\cdots$ |
| $P_2^A$ | $e_2$ | $e_2$ | $\cdots$ |
| $P_3^A$ | $e_2$ | $e_3$ | $\cdots$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |
| $P_\alpha^A$ | $e_\alpha$ | $e_\alpha$ | $\cdots$ |

To prove that the algorithm will terminate, we observe that every time the while-loop is executed, the edge corresponding to one of the $A$-paths is replaced by another upstream edge on the same path, while the edges correspond to all the other $A$-paths remain unchanged. Hence, the table can only have at most $\sum_i |P_i^A|$ columns.

It remains to show that the $c_i^*$'s found by the algorithm have the properties stated in Definition 4.2. According to the algorithm, the $c_i^*$'s are selected from the set of all junction edges

of the paths $P_i^A$'s. Therefore, each $c_i^*$ must be a junction edge between $P_i^A$ and $P_j^B$ for some integer $j$ so that Condition 1 in Definition 4.2 is satisfied. Condition 2 is also satisfied because there exists at most one $c_i^*$ on each $P_j^B$'s, or else the while-loop will continue to execute. To prove that $\{c_i^*\}$ forms a set of critical edges, it remains to show that they satisfy Condition 3, i.e., for any two $c_i^*$ and $c_{i'}^*$ where

$$c_i^* \in P_i^A \cap P_j^B$$

and

$$c_{i'}^* \in P_{i'}^A \cap P_{j'}^B,$$

there exists no edge $e \in P_i^A \cap P_{j'}^B$ such that

$$c_{i'}^* \succ_{P_{j'}^B} e \succ_{P_i^A} c_i^*.$$

This will be shown by contradiction. Assume the contrary, i.e., for some $i \neq i'$, where

$$c_i^* \in P_i^A \cap P_j^B$$

and

$$c_{i'}^* \in P_{i'}^A \cap P_{j'}^B,$$

there exists an edge $e \in P_i^A \cap P_{j'}^B$ such that

$$c_{i'}^* \succ_{P_{j'}^B} e \succ_{P_i^A} c_i^*.$$

Since

$$e \succ_{P_i^A} c_i^*,$$

by Lemma 4.4,

$$e = h_{j'}^B(k)$$

for some $k \geq 2$. Then by Lemma 4.5, there exists an integer $l \leq k$ such that

$$c^*_{\gamma_{j'}(l)} = h_{j'}^B(l).$$

This implies that $c^*_{\gamma_{j'}(l)}$ is on $P_{j'}^B$. Since

$$c_{i'}^* \succ_{P_{j'}^B} e,$$

$c_{i'}^*$ is also on $P_{j'}^B$. This is a contradiction because when the algorithm terminates, two critical edges cannot be on the same $B$-path. □

**Algorithm 4.2.** Given the EDP's $P_1^A, P_2^A, \ldots, P_\alpha^A$ to $A$ and EDP's $P_1^B, P_2^B, \ldots, P_\beta^B$ to $B$, where $A \subset B$, the following algorithm constructs EDP's $\overline{P}_1^A, \overline{P}_2^A, \ldots, \overline{P}_\alpha^A$ to $A$ and EDP's $\overline{P}_1^B$, $\overline{P}_2^B$, $\ldots$,$\overline{P}_\beta^B$ to $B$ such that the former is a subset of the latter.

{

*for* $(i = 1; i \leq \alpha; i++)$ {

$\overline{P}_i^A := P_i^A$;

}

*for* $(j = 1; j \leq \beta; j++)$ {

$\overline{P}_j^B := P_j^B$;

}

*//This is just the initialization.*

*//The* $\overline{P}_i^A$ *and* $\overline{P}_j^B$ *will be updated later.*

*for* $(i = 1; i \leq \alpha; i++)$

{

$c :=$ *the critical edge of* $P_i^A$;

$P_j^B :=$ *the* $P^B$ *which contains* $c$;

$\overline{P}_i^A := \{e_B \in P_j^B : e_B \prec_{P_j^B} c\} \cup \{c\} \cup \{e_A \in P_i^A : c \prec_{P_i^A} e_A\}$;

$\overline{P}_j^B := \overline{P}_i^A$;

}

}

The output of the algorithm is the set of $\alpha$ newly rearranged paths from source node $s$ to $A$, $\{\overline{P}_i^A : i = 1, 2, \ldots, \alpha\}$ and the set of $\beta$ newly rearranged paths from source node $s$ to $B$, $\{\overline{P}_j^B : j = 1, 2, \ldots, \beta\}$. We claim that the set of rearranged paths from $s$ to $A$ is a subset of the set of rearranged paths from $s$ to $B$ and every $\overline{P}_j^B$ is edge-disjoint with each other. Mathematically, we can write

$$\left\{\overline{P}_i^A : i = 1, 2, \ldots, \alpha\right\} \subset \left\{\overline{P}_j^B : j = 1, 2, \ldots, \beta\right\}, \tag{4.1}$$

and, for any $j_1 \neq j_2$,

$$\overline{P}_{j_1}^B \cap \overline{P}_{j_2}^B = \emptyset. \tag{4.2}$$

*Justification.* We first prove equation (4.1). According to the last line of the algorithm, each $\overline{P}_i^A$ is assigned to be $\overline{P}_j^B$. Therefore the proof is trivial.

We now prove equation (4.2). Referring to the construction of the $\overline{P}_j^B$'s, we can see that either one of the following must be true:

I. $\overline{P}_j^B = P_{j'}^B$ for some positive integer $j' \leq \beta$.

II. $\overline{P}_j^B = \{e_B \in P_{j'}^B : e_B \prec c\} \cup \{c\} \cup \{e_A \in P_{i'}^A : c \prec e_A\}$ for some positive integers $i' \leq \alpha$ and $j' \leq \beta$ and some edge $c \in E$. The edge $c$ must be the critical edge of $P_{i'}^A$.

For a particular $\overline{P}_j^B$, we say it is of Type I if it satisfies Condition I. Otherwise, it is of Type II (if it satisfies Condition II). For any two distinct $\overline{P}_{j_1}^B$ and $\overline{P}_{j_2}^B$, we consider the following three mutually exclusive cases:

**Case 1: Both $\overline{P}_{j_1}^B$ and $\overline{P}_{j_2}^B$ are of Type I.**

If both of them are of type I, then there exist two positive integers $j_1', j_2' \leq \beta$ such that

$$\overline{P}_{j_1}^B = P_{j_1'}^B,$$

and

$$\overline{P}_{j_2}^B = P_{j_2'}^B.$$

Since $P_{j_1'}^B$ and $P_{j_2'}^B$ are edge-disjoint, $\overline{P}_{j_1}^B$ and $\overline{P}_{j_2}^B$ must be edge-disjoint too.

**Case 2: Both $\overline{P}_{j_1}^B$ and $\overline{P}_{j_2}^B$ are of Type II.**

If both of them are of type II, then there exists two positive integers $i_1', i_2' \leq \alpha$ and $j_1', j_2' \leq \beta$ and two edges $c_1, c_2 \in E$ such that

$$\overline{P}_{j_1}^B = \{e_B \in P_{j_1'}^B : e_B \prec c_1\} \cup \{c_1\} \cup \{e_A \in P_{i_1'}^A : c_1 \prec e_A\},$$

and

$$\overline{P}^B_{j_2} = \{e_B \in P^B_{j'_2} : e_B \prec c_2\} \cup \{c_2\} \cup \{e_A \in P^A_{i'_2} : c_2 \prec e_A\}.$$

The $c_1$ and $c_2$ are the critical edges of $P^A_{i'_1}$ and $P^A_{i'_2}$, respectively. Since $P^B_{j'_1}$ and $P^B_{j'_2}$ must be edge-disjoint, $\{e_B \in P^B_{j'_1} : e_B \prec c_1\}$ must be edge-disjoint with $\{e_B \in P^B_{j'_2} : e_B \prec c_2\}$. Also,

$$c_2 \notin \{e_B \in P^B_{j'_1} : e_B \prec c_1\}.$$

Moreover, $\{e_B \in P^B_{j'_1} : e_B \prec c_1\}$ must be edge-disjoint with $\{e_A \in P^A_{i'_2} : c_2 \prec e_A\}$, or otherwise, there would be an edge $e' \in P^B_{j'_1} \cap P^A_{i'_2}$ such that

$$c_2 \prec_{P^A_{i'_2}} e' \prec_{P^B_{j'_1}} c_1,$$

which is a contradiction to the fact that $c_1$ and $c_2$ are critical. Since $P^A_{i'_1}$ must be edge-disjoint with $P^A_{i'_2}$, we can conclude that $\{e_A \in P^A_{i'_1} : c_1 \prec e_A\}$ and $\{e_A \in P^A_{i'_2} : c_2 \prec e_A\}$ are edge-disjoint, and hence,

$$c_1 \neq c_2$$

and

$$c_1 \notin \{e_A \in P^A_{i'_2} : c_2 \prec e_A\}.$$

Thus, each component of $\overline{P}^B_{j_1}$ is edge-disjoint with all of the components of $\overline{P}^B_{j_2}$. By symmetry, each component of $\overline{P}^B_{j_2}$ is

edge-disjoint with all of the components of $\overline{P}^B_{j_1}$ too. Hence, $\overline{P}^B_{j_1}$ and $\overline{P}^B_{j_2}$ are edge-disjoint.

**Case 3: One of them is of Type I while the other is of Type II.**

Without loss of generosity, suppose $\overline{P}^B_{j_1}$ is of Type I and $\overline{P}^B_{j_2}$ is of Type II. Then there exists a positive integer $j'_1 \leq \beta$ such that

$$\overline{P}^B_{j_1} = P^B_{j'_1}.$$

Besides, there exist positive integers $i'_2 \leq \alpha$ and $j'_2 \leq \beta$ such that

$$\overline{P}^B_{j_2} = \{e_B \in P^B_{j'_2} : e_B \prec c_2\} \cup \{c_2\} \cup \{e_A \in P^A_{i'_2} : c_2 \prec e_A\},$$

where $c_2 \in E$ is the critical edge of $P^A_{i'_2}$. Observe that $\{e_A \in P^A_{i'_2} : c_2 \prec e_A\}$ must be edge-disjoint with $P^B_{j'_2}$, or otherwise $c_2$ cannot be the critical edge of $P^A_{i'_2}$. Obviously, $P^B_{j'_1}$ and $P^B_{j'_2}$ are edge-disjoint. We can conclude that $\overline{P}^B_{j_1}$ is edge-disjoint with $\overline{P}^B_{j_2}$.

Therefore, in all three cases, any two paths in the set $\{\overline{P}^B_j : j = 1, 2, \ldots, \beta\}$ are edge-disjoint. This proves equation (4.2) and hence justifies Algorithm 4.2. □

### 4.3.3 Extended Graph

For every acyclic digraph $G = (V, E)$, we extend this graph to form an extended graph $G' = (V', E')$ from $G$ through the following mechanism: For every channel $e = (a, b) \in E$ where $a, b \in V$, we segment it into two sub-edges, $(a, c)$ and $(c, b)$, and install a new node $c$ between them. This newly constructed acyclic digraph is designated as the extended graph $G'$ of the original graph $G$.

For a digraph $G = (V, E)$ and its extended graph $G' = (V', E')$, we define two functions

$$g_1 : E \to E'$$

and

$$g_2 : E \to E'$$

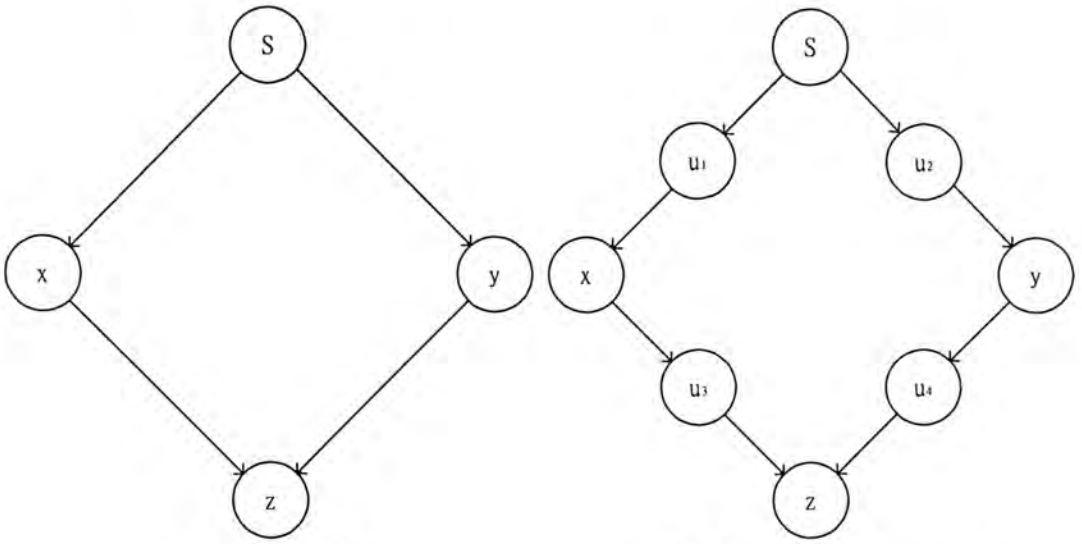
such that for all $a, b \in V$ and $c \in V' \setminus V$ such that $(a, c), (c, b) \in E'$ and $(a, b) \in E$,

$$g_1((a, b)) = (a, c),$$

and

$$g_2((a, b)) = (c, b).$$

An example of extended graph is shown in Figure 4.5.




(a) Original graph $G$ (b) Extended graph $G'$

Figure 4.5: An example of extended graph. The nodes $u_1, u_2, u_3$ and $u_4$ are inserted and the edges are segmented into two.

**Theorem 4.3.** An $\omega$-dimensional linear dispersion on the extended graph $G'$ implies an $\omega$-dimensional generic linear network code on the original graph $G$.

*Proof.* Suppose we are given an acyclic digraph $G = (V, E)$ and an $\omega$-dimensional linear dispersion on its extended graph $G' = (V', E')$ specified by $\overline{f}_{e'} \in F^\omega$ for each $e' \in E'$. Without loss of generosity, we can assume that

$$\overline{f}_{g_1(e)} = \overline{f}_{g_2(e)}$$

for all $e \in E$. We first prove a property of such a dispersion on $G'$ that will enable us to construct a generic linear network

code on $G$. Let $\{e'_1, e'_2, \ldots, e'_m\}$ be an arbitrary set of edges in $G'$ with $m \leq \omega$, such that

$$e'_j \in Out(t_j),$$

and

$$e'_j \in In(u_j).$$

(c.f. Fig. 4.6)

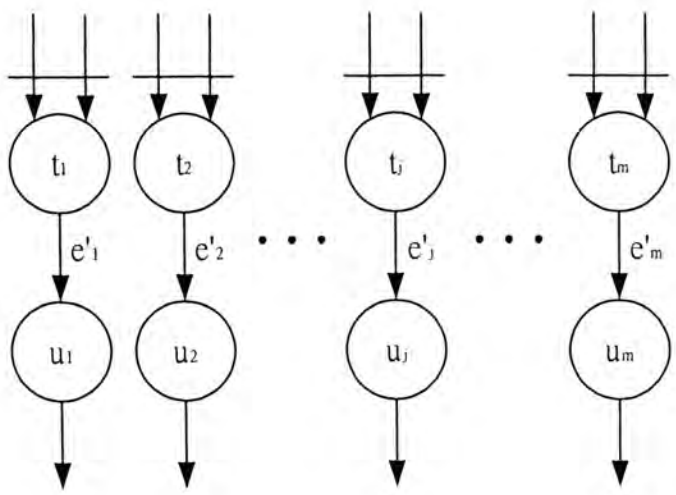



Figure 4.6: We are given a dispersion on $G'$

Given that

$$\left\langle \left\{ \overline{f}_d : d \in In(t_j) \right\} \right\rangle \not\subset \left\langle \left\{ \overline{f}_{e'_k} : k \neq j \right\} \right\rangle$$

for $1 \leq j \leq m$, we now prove the linear independence of $\overline{f}_{e'_1}, \overline{f}_{e'_2}, \ldots, \overline{f}_{e'_m}$. Since

$$V_{t_j} = \left\langle \left\{ \overline{f}_d : d \in In(t_j) \right\} \right\rangle \not\subset \left\langle \left\{ \overline{f}_{e'_k} : k \neq j \right\} \right\rangle$$

for $1 \leq j \leq m$, there exists a vector $w \in V_{t_j}$ such that

$$w \notin \langle \{\overline{f}_{e'_k} : k \neq j\} \rangle .$$

Therefore, we have

$$dim \langle V_{t_j} \cup \{\overline{f}_{e'_k} : k \neq j\} \rangle > dim \langle \{\overline{f}_{e'_k} : k \neq j\} \rangle \qquad (4.3)$$

Furthermore, as it is a linear dispersion,

$$\begin{aligned} & dim \left( \langle \{\overline{f}_{e'_k} : k \neq j\} \rangle \right) \\ = \; & dim \left( \langle \cup_{k \neq j} V_{u_k} \rangle \right) \\ = \; & maxflow \left( \{u_k : k \neq j\} \right), \end{aligned}$$

and

$$\begin{aligned} & dim \left( \langle V_{t_j} \cup \{\overline{f}_{e'_k} : k \neq j\} \rangle \right) \\ = \; & min(\omega, maxflow(\{t_j\} \cup \{u_k : k \neq j\})). \end{aligned}$$

We will prove that $\overline{f}_{e'_j} \notin \langle \{\overline{f}_{e'_k} : k \neq j\} \rangle$. Assume the contrary is true, i.e.,

$$\overline{f}_{e'_j} \in \langle \{\overline{f}_{e'_k} : k \neq j\} \rangle .$$

Then

$$\begin{aligned} & maxflow(\{u_k : k = 1, 2, \ldots, m\}) \\ = \; & dim \left( \langle \cup_k V_{u_k} \rangle \right) \end{aligned}$$

$$\begin{aligned} &= dim\left(\left\langle\left\{\overline{f}_{e'_k} : k = 1, 2, \ldots, m\right\}\right\rangle\right) \\ &= dim\left(\left\langle\left\{\overline{f}_{e'_k} : k \neq j\right\}\right\rangle\right) \\ &= maxflow(\{u_k : k \neq j\}). \end{aligned} \tag{4.4}$$

Let

$$maxflow(\{u_k : k = 1, 2, \ldots, m\}) = \alpha$$

and

$$maxflow(\{t_j\} \cup \{u_k : k \neq j\}) = \beta.$$

Hence, by (4.4),

$$maxflow(\{u_k : k \neq j\}) = \alpha.$$

Obviously, we have

$$\{u_k : k \neq j\} \subset \{t_j\} \cup \{u_k : k \neq j\}.$$

Hence, we have

$$\beta \geq \alpha.$$

Assume

$$\beta > \alpha,$$

or equivalently,

$$\beta - \alpha > 0.$$

Then by Algorithm 4.2, we can choose $\beta$ EDP's from $s$ to $\{t_j\} \cup \{u_k : k \neq j\}$ in which $\alpha$ of them terminate at $\{u_k : k \neq j\}$ and $\beta - \alpha$ of them terminate at $t_j$. As a path from $s$ to $t_j$ implies a path from $s$ to $u_j$, there exists one path (among those $\beta - \alpha$ paths) from $s$ to $u_j$ which is edge-disjoint with the $\alpha$ edge-disjoint paths from $s$ to $\{u_k : k \neq j\}$. Therefore, there are at least $\alpha + 1$ edge-disjoint paths from $s$ to $\{u_k : k = 1, 2, \ldots, m\}$. It implies that

$$maxflow(\{u_k : k = 1, 2, \ldots, m\}) \geq \alpha + 1,$$

which is clearly a contradiction. Hence we conclude that

$$\beta = \alpha,$$

that is,

$$\begin{aligned} & maxflow(\{t_j\} \cup \{u_k : k \neq j\}) \\ &= maxflow(\{u_k : k = 1, 2, \ldots, m\}) \\ &= maxflow(\{u_k : k \neq j\}) \end{aligned}$$

and hence

$$\begin{aligned} & dim\left(\langle V_{t_j} \cup \{\overline{f}_{e'_k} : k \neq j\}\rangle\right) \\ &= min(\omega, maxflow(\{t_j\} \cup \{u_k : k \neq j\})) \end{aligned}$$

$$\begin{aligned} &\leq maxflow(\{t_j\} \cup \{u_k : k \neq j\}) \\ &= maxflow(\{u_k : k \neq j\}) \\ &= dim\left(\left\langle\left\{\overline{f}_{e'_k} : k \neq j\right\}\right\rangle\right), \end{aligned}$$

which is a contradiction to (4.3). Therefore, we conclude that $\overline{f}_{e'_j} \notin \left\langle\left\{\overline{f}_{e'_k} : k \neq j\right\}\right\rangle$. Therefore all $\overline{f}_{e'}$'s are linearly independent.

Finally, we construct a generic linear network code on $G = (V, E)$ by letting $f_e = \overline{f}_{e'}$ for all $e \in E$ where $e' = g_1(e)$. Hence we conclude that a linear dispersion on the extended graph incorporates a generic linear network code on the original graph. □

---

□ **End of chapter.**

# Chapter 5

# Upper Bound on the Size of the Base Field


**Summary**

This chapter aims to describe the relation between the size of the base field and the four classes of linear network codes as discussed in the previous chapters. Although the upper bound on the size of the base field for a generic network code has been given in [10], we can obtain another upper bound through analyzing the relation between the four classes of linear network codes.

## 5.1 Base Field Size Requirement

### 5.1.1 Linear Multicast

A linear network code construction algorithm has been given by [6]. This algorithm constructs an $\omega$-dimensional $F$-valued linear multicast on an acyclic network when $|F| > \eta$, the number of non-source nodes $t$ in the network with $maxflow(t) \geq \omega$. It implies that a base field with at least $\eta + 1$ elements is sufficient for constructing a linear multicast. In other words, the upper bound on the minimum possible size of the base field $F$ is $\eta + 1$.

### 5.1.2 Linear Broadcast

Consider a network $G = (V, E)$ and its auxiliary network $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ as defined in Chapter 4. Suppose we construct a linear multicast on $\widetilde{G}$ using the algorithm devised in [6]. The upper bound on the minimum possible size of the base field for this multicast is $\tilde{\eta} + 1$, which is the number of non-source nodes $\tilde{t} \in \widetilde{V}$ in the network $\widetilde{G}$ with $maxflow(\tilde{t}) \geq \omega$. We can easily check that

$$\tilde{\eta} = |V| - 1.$$

Since we have proved that a linear multicast on $\widetilde{G}$ incorporates a linear broadcast on $G$, the base field size requirement for

a multicast on $\widetilde{G}$ is equivalent to the requirement for a broadcast on $G$. Hence we can conclude that the minimum possible size of the base field for constructing a linear broadcast is upper bounded by $|V|$.

### 5.1.3 Linear Dispersion

Consider a network $G = (V, E)$ and its expanded network $\widehat{G} = (\widehat{V}, \widehat{E})$ as defined in Chapter 4. Suppose we construct a linear broadcast on $\widehat{G}$. The upper bound on the minimum possible size of the base field for this broadcast is $|\widehat{V}|$. We can easily check that

$$|\widehat{V}| \leq 2^{|V|-1}.$$

Since we have proved that a linear broadcast on $\widehat{G}$ incorporates a linear dispersion on $G$, the base field size requirement for a broadcast on $\widehat{G}$ is equivalent to the requirement for a dispersion on $G$. Hence we can conclude that the minimum possible size of the base field for constructing a linear dispersion is upper bounded by $2^{|V|-1}$.

### 5.1.4 Generic Network Code

Consider a network $G = (V, E)$ and its extended network $G' = (V', E')$ as defined in Chapter 4. Suppose we construct a linear dispersion on $G'$. The upper bound on the minimum possible size of the base field for this dispersion is $2^{|V'|-1}$. We can easily check that

$$2^{|V'|-1} = 2^{|V|+|E|-1}.$$

Since we have proved that a linear dispersion on $G'$ incorporates a generic network code on $G$, the base field size requirement for a dispersion on $G'$ is equivalent to the requirement for a generic network code on $G$. Hence we can conclude that the minimum possible size of the base field for constructing a generic network code is upper bounded by $2^{|V|+|E|-1}$.

The upper bounds are summarized in the following table.

| Linear Network Code | Upper Bound |
|---|---|
| Multicast | $\eta$ |
| Broadcast | $\|V\|$ |
| Dispersion | $2^{\|V\|-1}$ |
| Generic Network Code | $2^{\|V\|+\|E\|-1}$ |

Table 5.1: The upper bound of linear network codes.

## 5.2 Upper Bounds Comparison for Generic Network Code

As we proved in the previous section, the upper bound on the minimum possible size of the base field for constructing a generic network code is $2^{|V|+|E|-1}$. This quantity only depends on $|V|$ and $|E|$. In other words, this bound only depends on the topology of the network.

Nevertheless, another upper bound has been proposed in [10]. An algorithm was proposed in [10] to construct an $\omega$-dimensional $F$-valued linear generic network code when the base field $F$ contains more than $\begin{pmatrix} |E|+\omega-1 \\ \omega-1 \end{pmatrix}$ elements. This bound depends on both $|E|$ and $\omega$, i.e., it depends on the topology of the network and the dimension of the network code.

The upper bound imposed on the base field size depends on the network code construction mechanism. In order to minimize the required field size, we need to analyze the network topology and the dimension of the network code before we can decide which algorithm would be used to construct the network code.

---

□ **End of chapter.**

# Chapter 6

# Future Work

Compared with multi-source network coding, single-source network coding is much better understood. Much work has been done on single-source and linear network coding. Yet, there are still some future works.

For a linear broadcast, every non-source node would be able to receive some dimensions of the original message. If the maximum flow of the node $t$ is large enough, i.e.,

$$maxflow(t) \geq \omega,$$

it would receive the complete message. Otherwise, it would receive some linear combinations of the message. For example, in the classical example of network coding (Figure 6.1), all three nodes $x$, $y$ and $w$ have maximum flow being 1, and therefore they all receive one dimension of the original message.

If we re-design the network code such that node $w$ can receive

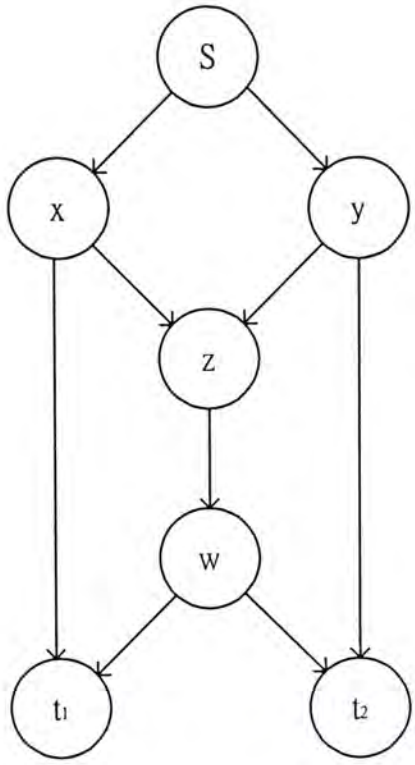



Figure 6.1: Nodes $x$, $y$ and $w$ have maximum flow being one.

$b_1$ and nodes $t_1$ and $t_2$ remain to receive both $b_1$ and $b_2$, then we can arrive at the solution as shown in Figure 6.2.

In a linear broadcast, we in general cannot control which dimensions/linear combinations of the source message each node will be receiving. This problem is highly related to the multi-source network coding problem.

In Chapter 3, we have derived a necessary condition for the existence of linear network codes from a node-based perspective. If we can further derive a sufficient condition based on the same

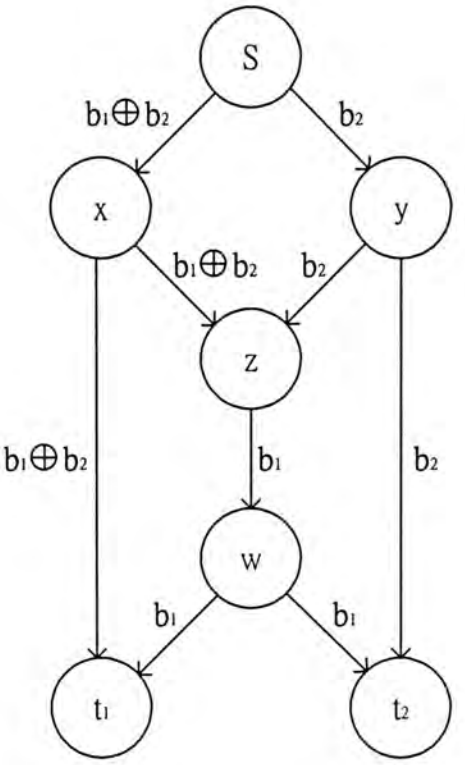



Figure 6.2: Re-design of network code to let node $w$ receive $b_1$.

formulation, we will have a complete node-based characterization of linear network code and hence a linear network code can be described as an assignment of vector subspaces to the nodes. One of the possible future work on linear network coding is to devise an algorithm which can construct a linear network code by assigning vector subspaces to the nodes, instead of assigning vectors to the channels. One common step in many construction algorithms currently available [1][6][10] is to assign a vector within some vector subspace to a channel. If there exists an al-

gorithm which can construct a linear network code by assigning vector subspaces to the nodes, it is expected that there would be a similar step in the algorithm, i.e., there would be a step in which a vector subspace is assigned to a node, in which the vector subspace must satisfy some properties. In choosing this vector subspaces, we may be able to deliberately select a particular vector subspace for a particular node such that a particular vector is inside this subspace. Hence, to certain extent, we can control the dimensions which the nodes receive.

---

□ **End of chapter.**

# Bibliography

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, 46:1204–1216, July 2000.

[2] K. K. Chi, S. L. Che, Y. Sun, and X. M. Wang. Network coding and analysis of field size for a kind of multicast network. Preprint, 2003.

[3] R. Dougherty, C. Freiling, and K. Zeger. Linearity and solvability in multicast networks. *IEEE Transactions on Information Theory*, 50:2243–2256, October 2004.

[4] L. R. Foulds. *Graph Theory Applications.* New York: Springer-Verlag, 1991.

[5] J. L. Gross and J. Yellen. *Handbook of Graph Theory.* Boca Raton:CRC Press, 2004.

[6] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen. Polynomial time algorithms for multicast network code construction. *IEEE Transactions on Information Theory*, 51:1973–1982, June 2005.

[7] R. Koetter and M. Medard. An algebraic approach to network coding. *IEEE/ACM Transactions on Networking*, 11:782–795, October 2003.

[8] S.-Y. R. Li, R. W. Yeung, and N. Cai. Linear network coding. *IEEE Transactions on Information Theory*, 49:371–381, October 2003.

[9] R. W. Yeung. *A First Course in Information Theory*. New York: Kluwer Academic/Plenum Publishers, 2002.

[10] R. W. Yeung, S.-Y. R. Li, N. Cai, and Z. Zhang. Theory of network coding. to appear in *Foundations and Trends in Communications and Information Theory*.

CUHK Libraries
004359081