On Network Coding for Communicating the Sum of Symbols from Multiple Sources

A thesis submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

by

Brijesh Kumar Rai
(Roll number: 04407004)

Advisors: Prof. Abhay Karandikar and Prof. Bikash Kumar Dey

Department of Electrical Engineering
Indian Institute of Technology Bombay
Powai, Mumbai, 400076
July 2010

Dedicated
to
Sugandha Didi and Nani

Thesis Approval Sheet

The thesis titled **"On Network Coding for Communicating the Sum of Symbols from Multiple Sources"** by **Brijesh Kumar Rai** is approved for the degree of Doctor of Philosophy.

Examiners

[illegible]
for Andrew Thangaraj .... [illegible] ....

Advisors

[illegible] .... Sikanth Kumar Dey ....

Chairman

[illegible] ....

Place :...I.I.T....Bombay

Date :..28-07-2010

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Brijesh Kumar Rai

(Signature)

BRIJESH KUMAR RAI

(Name of the student)

04407004

(Roll No.)

Date: 28-07-2010

Abstract

In this thesis, we consider a communication problem over a directed acyclic network of unit capacity links having m sources and n terminals, where each terminal requires the sum of symbols generated at all the sources. We assume that each source generates one i.i.d. random process with uniform distribution over a finite alphabet having an abelian group structure, and the different source processes are independent. We also assume that each node in the network is capable of implementing network coding. We further assume that all the links in the network are error-free and delay free. We call such a directed acyclic network as a *sum-network*.

First, we consider the problem of rate-1 linear network code solutions over finite fields. We construct two special classes of sum-networks. In the first constructed class, for any finite set of primes, there exists a sum-network which has rate-1 linear network code solutions for every code length only over finite fields of characteristics belonging to the given set. In the second constructed class, for any finite set of primes, there exists a sum-network which has rate-1 linear network code solutions for every code length only over finite fields of characteristics, which do not belong to the given set. Then we show that there exists a sum-network, where a scalar linear network code solution not only depends on the characteristic of the finite field, but also on the size of the field. We also show that there exist sum-networks where source-terminal pairs connectivity is insufficient for a rate-1 network code solution for any code length.

Next, we show the solvable equivalence of sum-networks with other type of networks. Specifically, we show that communicating the sum and communicating any linear function are solvably (respectively linear solvably) equivalent under fractional network coding (respectively fractional linear network coding) if the considered alphabet is a module over a ring and the component linear functions are invertible. Then we show that there exists a solvably (and linear solvably) equivalent sum-network for any multiple-unicast network (and more generally, for any acyclic directed network where each terminal re-

quires a subset of the symbols generated at all the sources). We also show that there exists a linear solvably equivalent multiple-unicast network for any sum-network.

As a consequence of our solvable equivalence results, some important known results, for multiple-unicast and directed acyclic networks where each terminal requires a subset of the symbols generated at all the sources, also hold for sum-networks. Specifically, we show that for any set of polynomials having integer coefficients, there exists a sum-network which is scalar linear solvable over a finite field $\mathbb{F}$ if and only if the polynomials have a common root in $\mathbb{F}$. Then we show that there exists a solvable sum-network over a finite alphabet whose reverse network is not solvable over any finite alphabet. However, we show that a sum-network and its reverse sum-network are solvably equivalent under fractional linear network coding. Similarly, we show the insufficiency of linear network coding and unachievability of the network coding capacity for sum-networks.

Finally, we consider the network coding capacity of sum-networks over a finite field. However, some of our results also hold over finite alphabets having more general algebraic structures, such as a module over a ring. The network coding capacity of a sum-network is upper bounded by the minimum of min-cut capacities of all source-terminal pairs over any alphabet. We call this upper bound the *min-cut bound.* We show that the min-cut bound is always achievable for sum-networks with $\min\{m, n\} = 1$ over sufficiently large finite fields. Moreover, scalar linear network coding is sufficient to achieve the min-cut bound. For sum-networks with $\min\{m, n\} = 2$, the network coding capacity over every finite field is known to be equal to the min-cut bound, when the min-cut bound is 1. For the min-cut bound greater than 1, we give a lower bound on the network coding capacity. For sum-networks with $\min\{m, n\} \geq 3$, we show that there exist sum-networks where the min-cut bound is not achievable over any alphabet. For this class, when the min-cut bound is 1, we give a lower bound on the network coding capacity and show that for sum-networks with $\min\{m, n\} = 3$, the lower bound is tight. We conjecture that the network coding capacity of a sum-network with $m = n = 3$ is either $0, 2/3$ or at least 1.

Contents

List of Figures

Chapter 1

Introduction

The area of Information Theory began with landmark work by Shannon [1]. On the one hand, Information Theory found applications in as diverse areas as Physics, Economics, Biology [2, 3, 4, 5, 6, 7], on the other hand, in the area of Communications Engineering, it laid bedrock for lossless and lossy data compression as well as error correcting codes for reliable communication over noisy channels. We now have near Shannon capacity achieving error correcting codes for various channel models [8, 9]. As the mode and means for communications kept enhancing, the need for a comprehensive theory to address the problems such as communication over shared channel, communication over a network also kept increasing. Attempts have been made by researchers to extend the concepts of Information Theory to address these problems. The branch of Information Theory that deals with communication networks is known as Network Information Theory [10].

While Information Theory brought many successes for point-to-point single sender-single receiver communication systems over 50 years of its emergence, it could not do the same for communication networks. The counter part of the channel capacity in the case of point-to-point single sender-single receiver is the capacity region (rate region) in the case of communication networks. In fact, full characterization of capacity region is not known even for fairly simple communication networks till date [10]. In most part of the work related to Network Information Theory, communication links have been assumed to be wireless links and thus erroneous. Please refer the survey paper [11] for more details about the development of Information Theory over 50 years of its emergence. Also refer the survey paper, "Information Theory and Communication Networks: An Unconsummated Union," by Ephremides et al. [12] for some of the issues in communication networks which have not been fully exploited.

Traditionally, information has been considered as a commodity which can only be *stored and forwarded* by the intermediate nodes (nodes other than sources and terminals) in a network. Consider a communication network having some sources, some terminals, some intermediate nodes, and error-free links such that each link can carry up to a fixed maximum amount of information flow/commodity flow. Each source generates one random process over some finite alphabet. Further, consider that there are multiple unique source-terminal pairs. A communication problem over such a network where each terminal requires generated information from its corresponding source is termed as *multicommodity flow problem*; and this network is called as a *multiple-unicast network*. In a multicommodity flow problem, the task is to determine the maximum concurrent flows in all the source-terminal pairs. Multicommodity flow problem is also termed as a *multiple-unicast problem* in the literature. The special case of a multiple-unicast network with single source and single terminal is called as a *unicast network*.

In a unicast network, the maximum amount of information flow/commodity flow from the source to the terminal is same as the min-cut capacity between the source and the terminal [13, 14]. The min-cut capacity between the source and the terminal is defined as follows: all the nodes in a network are partitioned into two subsets such that the first subset contains the source and the second subset contains the terminal. In an undirected network, the cut capacity is defined as the sum of capacities of those edges whose end nodes are in different partitions; while for the directed graphs, the cut capcity is defined as the sum of capacities of the edges going from the first partition to the second partition. Min-cut capacity is defined as the minimum of cut capacities of all possible partitions. For the *multicast networks*, where there is a single source and multiple terminals, and all the terminals require the same information generated at the source, the maximum commodity flow problem is termed as "Steiner tree packing problem", which is proven to be NP-hard problem [15].

The seminal work by Alswede et al. [16] demonstrated that mixing information at the intermediate nodes in a network may provide better throughput. Mixing of information at the intermidiate nodes and sending it to outgoing links is referred as *network coding*. For a multicast network, it has been shown that the capacity (maximum information flow) under network coding is the minimum of the min-cut capacities between the source and each of the terminals [16]. This work caught immediate attention of researchers from research areas, such as Information Theory, Graph Theory, Complexity Theory as it showed a different method to attack the problems in Network Information Theory. Since then, there has been very active research in the area of network coding for the last 10 years. In fact, it is because of the excitement generated by the concept of network coding and the hope that it may be useful in solving Network Information Theory problems that in the editorial article [17], Cai et al. commented, "while the union between networking and information theory is still not fully consummated, the two fields do seem to be dating now." A plethora of research papers have been published in the area of network coding after its emergence. There has been a number of Ph.D. thesis also in the area of network coding, such as [18, 19, 20, 21, 22, 23], just to mention a few.

The problem of function computation over a network, where each terminal requires certain function of the information generated at the sources, is the most generalized form of communication problem over a network. Every communication problem over a network can be modelled as a function computation problem by choosing appropriate functions for the terminals.

In this thesis, we consider a simple case of the general function computation problem over a network. We consider the problem of communicating the sum of symbols generated at all sources to all the terminals over a directed acyclic network using network coding. We assume that each source generates one independent and identically distributed (i.i.d.) random process with uniform distribution over a finite alphabet, having abelian group structure, and the different source processes are independent. We further assume that all the links in the network are error-free and delay free. Before giving a brief account of the related work, we present various definitions which will be used throughout the thesis.

1.1 Definitions

1.1.1 Directed Acyclic Network

A directed acyclic network $\mathcal{N}$ is represented by a directed acyclic multigraph $\mathcal{G} = (V, E)$, where V is a finite set of nodes and $E \subseteq V \times V \times \mathbb{Z}_+$ is the set of edges in the network.

For any edge $e = (v_i, v_j, \varphi) \in E$, the node v_j is called the head of the edge and the node v_i is called the tail of the edge; and are denoted as $head(e)$ and $tail(e)$ respectively. In the case of no ambiguity, we will represent an edge (v_i, v_j, φ) simply as (v_i, v_j). For each node v, $In(v) = \{e \in E \colon head(e) = v\}$ is the set of incoming edges to the node v. Similarly, $Out(v) = \{e \in E \colon tail(e) = v\}$ is the set of outgoing edges from the node v. A sequence of nodes $(v_1, v_2, \ldots, v_\varsigma)$ is called a path, denoted as $v_1 \to v_2 \to \cdots \to v_\varsigma$, if $(v_i, v_{i+1}) \in E$ for $i = 1, 2, \ldots, \varsigma - 1$.

Among the nodes, a set of nodes $S \subseteq V$ are sources and a set of nodes $T \subseteq V$ are terminals. We will denote the sources by $s_1, s_2, \ldots, s_{|S|}$ and terminals by $t_1, t_2, \ldots, t_{|T|}$, where $|.|$ denotes the cardinality of a set. We assume that a source does not have any incoming edge and a terminal does not have any outgoing edge. Each source generates a set of random processes over an alphabet. In general, each terminal in the network may have a requirement of some part of the symbols or their functions available at a specific set of sources. Each edge in the network is assumed to be capable of carrying a symbol from the alphabet in each use. Each edge is used once per unit time and is assumed to be a error-free and delay-free communication channel.

Depending on the requirement of terminals in a network, we define some classes of networks in the next section.

1.1.2 Types of Networks

Definition 1. *A directed network with some sources and some terminals where each source generates possibly multiple independent random processes and each terminal requires to recover a set of the random processes generated by the sources is called a* **Type I** *network.*

Definition 2. *A Type I network where each source generates one random process with all the source processes being independent and each terminal requires to recover one source process is called a* **Type IA** *network.*

The widely studied classes of *Multicast networks* and *Multiple-unicast networks* [16, 26, 27, 28, 40, 41] are two subclasses of Type IA networks.

Definition 3. *A directed network with some sources and some terminals where each* *source generates possibly multiple independent random processes and each terminal requires to recover a function of the source random processes is called a* **Type II** *network.*

Though Type II network is defined here for completeness, we will mostly deal with simpler Type II networks in this thesis.

Definition 4. *A Type II network where every source generates one random process and all the terminals require to recover the same function of the random processes is called a* **Type IIA network**.

As a special case, we will consider a network having m sources and n terminals where each source generates one i.i.d. random process with uniform distribution over an alphabet which is an R-module $\mathcal{A}$, where R is a ring. Let $X_i \in \mathcal{A}$ be the message generated at source s_i for $1 \leq i \leq m$. Each terminal requires a linear function of the processes of the form

$$f(X_1, X_2, \ldots, X_m) = f_1(X_1) + f_2(X_2) + \ldots + f_m(X_m), \tag{1.1}$$

where $f_1, f_2, \ldots, f_m \in \text{End}(\mathcal{A})$, $\text{End}(\mathcal{A})$ denotes the set of all R-endomorphism of $\mathcal{A}$. Such a network will be called a **linear-network**.

Example 1. An example of $\text{End}(\mathcal{A})$ is a function $f : X \longrightarrow MX$, where $X \in \mathcal{A} = R^l$ and $M \in R^{l\times l}$. For such a function, a linear-network is defined as a network where all the terminals in the network require a function $f(X_1, X_2, \ldots, X_m) = M_1X_1 + M_2X_2 + \ldots + M_mX_m$, where $X_1, X_2, \ldots, X_m \in R^l$ and $M_1, M_2, \ldots, M_m \in R^{l\times l}$.

Since a vector space over a field $\mathbb{F}$ is a special case of R-module $\mathcal{A}$, a linear network is defined over a vector space. Similarly, any ring and any field are special cases of an R-module. So, a linear network is defined over these algebraic structure also. Moreover, any abelian group is a module over the integer ring. Thus, a linear network is defined over a $\mathbb{Z}$-module, where $\mathbb{Z}$ is the ring of integers.

As a further special case, a linear network where functions $f_1, f_2, \ldots, f_m$ in (1.1) are identity maps will be called as a **sum-network**. In other words, a sum-network is a network where all the terminals require to recover $X_1 + X_2 + \ldots + X_m$, where $X_1, X_2, \ldots, X_m \in \mathcal{A}$.

A sum-network is defined over a vector space, a ring, a field, and a $\mathbb{Z}$-module. In this thesis, in the context of sum-networks, we will assume that the alphabet has an abelian group structure. Further, in most works in this thesis, we will assume that the albhabet has additional abgebraic structures together with the abelian group structure.

In this thesis, we focus on directed acyclic sum-networks. When we mention sum-network, it will mean directed acyclic sum-network thoughout the thesis.

1.1.3 Network Coding

As mentioned previously, mixing of information at the intermidiate nodes and sending it to outgoing links is referred as network coding. For any network, we assume that each source generates one random process and each terminal has a requirement of only one source process or a function of the source processes. Any network can be analyzed like this by adding artificial sources and terminals in the original network. A network code for a network is an assignment of an edge function for each edge and a decoding function for each terminal. The most general form of network coding can be captured by a (k, l) fractional network code over a finite alphabet $\mathcal{A}$. In a (k, l) fractional network code over $\mathcal{A}$, an edge function f_e for an edge e is defined as

$$f_e \colon \mathcal{A}^k \to \mathcal{A}^l, \quad \text{if} \;\; \mathit{tail}(e) \in S \tag{1.2}$$

and

$$f_e \colon \mathcal{A}^{l|In(v)|} \to \mathcal{A}^l, \quad \text{if} \;\; \mathit{tail}(e) \notin S. \tag{1.3}$$

A decoding function g_v for a terminal v is defined as

$$g_v \colon \mathcal{A}^{l|In(v)|} \to \mathcal{A}^k. \tag{1.4}$$

A (k, l) *fractional network code solution over a finite alphabet* $\mathcal{A}$ is a (k, l) fractional network code over $\mathcal{A}$ which fulfills the requirement of each terminal in the network k times in l times uses of the network. The ratio k/l is the *rate* of a (k, l) fractional network code. When $k = l$, a (k, k) fractional network code is referred as *k-length network code* and k is referred as *code length* of the network code. When $k = l = 1$, a $(1, 1)$ fractional network code is referred as *scalar network code*. When $k = l$, a (k, k) fractional network code solution is referred as *k-length network code solution*. When $k = l = 1$, a $(1, 1)$ fractional network code solution is referred as *scalar network code solution*.

In the context of sum-networks, for clarity, we describe the concept of a (k, l) fractional network code solution as follows. Consider a sum-network having m sources and n terminals. Let k symbols generated from a finite alphabet, having an abelian group structure, at each source be grouped to form m k-length vectors $X_i = [X_{i1} X_{i2} \dots X_{ik}]^T$ for $1 \leq i \leq m$, where T denotes transpose. Let R_j be the message recovered by terminal t_j for $1 \leq j \leq n$. A (k, l) fractional network code solution for the sum-network would mean that all the n terminals receive $R_j = [\sum_{i=1}^{m} X_{i1} \quad \sum_{i=1}^{m} X_{i2} \dots \sum_{i=1}^{m} X_{ik}]^T$ for $1 \leq j \leq n$.

In the following, we describe the concept of *linear network coding*.

1.1.3.1 Linear Network Code

A *linear network code* is a network code where all the edge functions and decoding functions are linear. In most part of the literature related to network coding, linear network codes are defined over finite fields [26, 27, 28, 29]. However, linear network codes can be defined over more general algebraic structures. For example, Dougherty et al. [34] considered linear network codes over R-modules. We describe linear network codes over R-modules in terms of linear functions as follows.

Let Y_e denotes the message transmitted through an edge $e \in E$ and R_v denotes the message recovered by the terminal $v \in T$. Let the alphabet be a finite module $\mathcal{A}$ over a ring R, the message carried by an edge e using a (k, l)-fractional linear network code is of the form

$$Y_e = \sum_{j:X_j \text{ generated at } tail(e)} \alpha_{j,e} X_j \qquad \text{if } tail(e) \in S, \tag{1.5}$$

where $X_j = \mathcal{A}^k$, $Y_e = \mathcal{A}^l$ and $\alpha_{j,e} \in R^{l \times k}$.

$$Y_e = \sum_{e':head(e')=tail(e)} \beta_{e',e} Y_{e'} \qquad \text{if } tail(e) \notin S, \tag{1.6}$$

where $Y_e, Y_{e'} = \mathcal{A}^l$ and $\beta_{e',e} \in R^{l \times l}$.

The message recovered by a terminal v is

$$R_v = \sum_{e \in In(v)} \gamma_e Y_e, \tag{1.7}$$

where $\gamma_e \in R^{k \times l}$ and $R_v \in \mathcal{A}^k$.
$\alpha_{j,e}$, $\beta_{e',e}$ and γ_e are called the local coding coefficients.

A (k, l) *fractional linear network code solution* for a sum-network is defined similar to the (k, l) fractional network code solution except that now edge functions and decoding functions are linear as described in (1.5), (1.6) and (1.7). The ratio k/l is the rate of a (k, l) fractional linear network code. When $k = l$, a (k, k) fractional linear network code is referred as *k-length linear network code* and k is referred as code length of the linear network code. When $k = l = 1$, a $(1, 1)$ fractional linear network code is referred as *scalar linear network code.* When $k = l$, a (k, k) fractional linear network code solution is referred as *k-length linear network code solution.* When $k = l = 1$, a $(1, 1)$ fractional linear network code solution is referred as *scalar linear network code solution.*

For a non-linear network code solution for a sum-network, the module structure of the alphabet is irrelevant except for the (abelian) group structure which is necessary for defining the problem (recovery of the *sum* at the terminals).

By a linear network code over a finite ring R, we will mean that the finite alphabet is a finite ring R and the local coding coefficients are also from R.

A rate r is said to be *achievable* under a class of network codes over a finite alphabet $\mathcal{A}$ if there exists a (k, l) fractional network code solution over $\mathcal{A}$ in that class such that $k/l \geq r$.

1.1.3.2 Network Coding Capacity

The *network coding capacity* of a network over a finite alphabet $\mathcal{A}$ is defined to be the suppremum of the achievable rates under the class of all network codes over $\mathcal{A}$.

It has been shown in [24] that *network coding capacity* is independent of the alphabet for Type I networks. However, for communicating the *arithmetic sum* of symbols generated at all sources to every terminal, network coding capacity may depend on the alphabet [25]. For the class of linear network codes over a finite alphabet $\mathcal{A}$ (such as a module over a ring), network coding capacity is referred as *linear network coding ca-*

pacity over $\mathcal{A}$. It has been shown in [34] that the linear network coding capacity may depend on the alphabet.

1.1.3.3 Path Gain

Given a linear network code for a network, $\prod_{i=1}^{\varsigma-2} \beta_{(v_i,v_{i+1}),(v_{i+1},v_{i+2})}$ is called the path gain of the path $v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_\varsigma$ where $\beta_{(v_i,v_{i+1},(v_{i+1},v_{i+2})}$ for $1 \leq i \leq \varsigma - 2$ are the local coding coefficients of the successive edge pairs.

1.1.3.4 Solvability

A network is said to be *solvable over a finite alphabet* $\mathcal{A}$ if it has a rate-1 network code solution over $\mathcal{A}$ for some code length. For a network, existence of a rate-1 network code solution over $\mathcal{A}$ for some code length is termed as *solvability over* $\mathcal{A}$ of the network.

A network is said to be *solvable* if it has a rate-1 network code solution over some alphabet for some code length. For a network, existence of a rate-1 network code solution over some alphabet for some code length is termed as *solvability* of the network.

A network is said to be *scalar solvable over a finite alphabet* $\mathcal{A}$ if it has a scalar network code solution over $\mathcal{A}$. For a network, existence of a scalar network code solution over $\mathcal{A}$ is termed as *scalar solvability over* $\mathcal{A}$ of the network.

A network is said to be *scalar solvable* if it has a scalar network code solution over some alphabet. For a network, existence of a scalar network code solution over some alphabet is termed as *solvability* of the network.

Similarly *linear solvable over* $\mathcal{A}$, *linear solvability over* $\mathcal{A}$, *linear solvable*, *linear solvability*, *scalar linear solvable over* $\mathcal{A}$, *scalar linear solvability over* $\mathcal{A}$, *scalar linear solvable*, and *scalar linear solvability* are defined for the case of linear network codes.

We note that a scalar network code solution (scalar linear network code solution respectively) over a finite alphabet implies a k-length network code solution (k-length linear network code solution respectively) over the same finite alphabet for every code length k. However, a k-length network code solution for k greater than 1 (respectively k-length linear network code solution for k greater than 1) over a finite alphabet does not imply a scalar linear network code solution over the same finite alphabet [32].

We also note that a Type I network having k-length network code solution over a finite alphabet $\mathcal{A}$, using non-linear network coding, has a scalar network code solution over any finite alphabet of size $|\mathcal{A}^k|$. However, this is true for sum-networks only when both $\mathcal{A}$ and the finite alphabet of size $|\mathcal{A}^k|$ have an algebraic structure where the operation "sum" is well defined.

1.1.3.5 Solvably Equivalent Networks

Two networks are said to be *solvably equivalent under* (k, l) *fractional network coding* when the first network has a (k, l) fractional network code solution over a finite alphabet if and only if the other network has a (k, l) fractional network code solution over the same alphabet.

Two networks are said to be *solvably equivalent* when the first network has a rate-1 network code solution over a finite alphabet for some code length if and only if the other network has a rate-1 network code solution over the same finite alphabet for the same code length.

Two networks are said to be *scalar solvably equivalent* when the first network is scalar solvable over some finite alphabet if and only if the other network is scalar solvable over the same finite alphabet.

Similarly, the terms *linear solvably equivalent under* (k, l) *fractional network coding*, *linear solvably equivalent*, and *scalar linear solvably equivalent* are defined for the case of linear network codes.

1.1.3.6 Reversibility

Given a network $\mathcal{N}$, its *reverse network* $\mathcal{N}'$ is defined as the network with the same set of vertices, the edges reversed keeping their capacities same, and the roles of sources and terminals interchanged.

A network $\mathcal{N}$ is said to be *reversible under* (k, l) *fractional network coding* when the network $\mathcal{N}$ has a (k, l) fractional network code solution over some finite alphabet if and only if its reverse network $\mathcal{N}'$ has a (k, l) fractional network code solution over the same finite alphabet. Existence of a reversible network under (k, l) fractional network coding is termed as *reversibility under* (k, l) *fractional network coding* for the network.

A network $\mathcal{N}$ is said to be *reversible* when the network $\mathcal{N}$ has a rate-1 network code solution over some finite alphabet for some code length if and only if its reverse network $\mathcal{N}'$ has a rate-1 network code solution over the same finite alphabet for the same code length. Existence of a reversible network is termed as *reversibility* for the network.

A network $\mathcal{N}$ is said to be *reversible under scalar network coding* when the network $\mathcal{N}$ has a scalar network code solution over some finite alphabet if and only if its reverse network $\mathcal{N}'$ has a scalar network code solution over the same finite alphabet. Existence of a reversible network under scalar network coding is termed as *scalar reversibility* for the network.

Similarly, the terms *linearly reversible under* (k, l) *fractional linear network coding*, *linear reversibility under* (k, l) *fractional linear network coding*, *linearly reversible*, *linear reversibility*, *linearly reversible under scalar network coding*, and *scalar linear reversibility* are defined for the case of linear network codes.

1.2 Related Work

As we have mentioned earlier, the area of network coding began with seminal work by Ahlswede et al. [16]. It has been shown by Li et al. [26] that scalar linear network coding over a "sufficiently" large finite field is sufficient to achieve the capacity of a multicast network. Koetter and Médard [27] have proposed an algebraic formulation of the linear network coding problem and related the network coding problem with determining roots of a set of polynomials. Jaggi et al. [28] have given a polynomial time algorithm to construct a linear network code for a multicast network. Tracy Ho. et al. [29] have shown that even when the local coding coefficients are chosen randomly and in a distributed fashion, the multicast capacity can be achieved with high probability. It is an open problem to find the capacity of an arbitrary multicast network where some or all nodes are not capable to implement network coding.

Subsequent to the above works, some authors have considered more general networks than multicast networks [30, 31, 32, 34, 34]. However, the requirements of the terminals have been restricted largely to subsets of the source symbols/processes. In the following, we outline some major results known till date.

Rasala Lehman et al. [30] have given a classification of the complexity of network coding problems and have shown that there exists an instance of Type I network where determining a scalar linear network code solution is an NP-hard problem. Langberg et al. [31] have shown that that there exists an instance of Type I network where determining a fractional linear network code solution is also an NP-hard problem.

It has been shown in [32] by construction that for Type I networks in general, scalar linear network coding is not sufficient in the sense that there exists a rate-1 linear network code solution of code length greater than 1 though there does not exists a scalar linear network code solution over any finite alphabet. Similar result has also been reported by Riis [33], showing the insufficiency of scalar linear network coding over binary alphabet field for an example acyclic directed network.

It has been shown in [34] that even fractional linear network coding is insufficient for Type I networks. They have also shown the insufficiency of fractional linear network coding over more general alphabets such as a module over a ring as well as for more general linear network codes [35]. It is an open problem to find necessary and sufficient conditions for linear solvabilty of Type I networks. It has been shown in [36] that for every set of integer polynomial equations, there exists a directed acyclic network which is scalar linear solvable over a finite field if and only if the set of polynomial equations has a solution over the same finite field. It has been shown in [24] that network coding capacity is independent of the alphabet for Type I networks. Dougherty et al. [38] have shown that there exists an instance of network coding problem where network coding capacity is unachievable.

It has been shown in [39, 40] that if a multiple-unicast network is scalar linear solvable over a finite alphabet then its reverse multiple-unicast network is also scalar linear solvable over the same finite alphabet. It has also been shown in [40] that there exists a multiple-unicast network which is scalar solvable over binary alphabet using non-linear network coding but its reverse multiple-unicast network is not scalar solvable over the binary alphabet. It has been shown in [41] that there exists a scalar solvable multiple-unicast network whose reverse multiple-unicast network is not scalar solvable over any finite alphabet. Necessary and sufficient conditions for scalar (linear) solvability of multiple-unicast networks are not yet known. For further information related to network coding, we refer interested readers to the following books [42, 43, 44, 45].

In this thesis, we consider a directed acyclic network with some sources and some terminals where each terminal needs to recover the sum of the symbols generated by the sources. In general, recovery of a wide class of functions of the source symbols may be of relevance in various applications. For instance, a sink node in a sensor network may require the average temperature (or other parameters) sensed by the sensors; or a network management node may require the average or maximum traffic at different parts of the network that are measured by different local nodes.

The problem of function computation over a network in general has been addressed in different contexts and in different flavours in the past. There are information theoretic works to characterize the achievable rate-region of various small networks with correlated sources [46, 47, 48, 49, 50]. A large body of work exists in determining the scaling laws for communication requirements in computing functions over large networks (see, for example, [51, 52, 53, 54, 55]).

Recently, there has been some work in the context of network coding for function computation over a network. Ramamoorthy [56] has considered the problem of computing sum of symbols generated at sources by every terminal in a directed acyclic network network. In such a network with only two sources or only two terminals, he has shown that this problem is scalar linear solvable over every finite field if and only if every source has a path to every terminal. There has also been some work by other authors in parallel to the work in this thesis. Langberg et al. [57] have shown that for a directed acyclic network having 3 sources and 3 terminals, if every source is connected with every terminal by at least two distinct paths then it is possible to communicate the sum of symbols generated at all sources to every terminal using scalar linear network coding. Appuswami et al. [25, 58] have considered the problem of communicating more general functions, for example, arithmetic sum (unlike sum as in finite fields), to one terminal.

1.3 Motivation for the Thesis

In any theoretical work, the motivation for it is driven by intellectual curiosity to study structures, to generalize existing results, and to find analogous results in different problem set-ups. The motivation for the work in this thesis has been the same. Specifically, the initial work in this thesis began with an attempt to generalize the work by Ramamoorthy [56]. This is the first paper where a function (*sum of the source symbols*) computation problem is addressed *using network coding*. The result in [56] is specific for directed acyclic networks having only two sources or two terminals. During the course of investigation, it turned out that the result in [56] could not be generalized for directed acyclic networks having more than two sources and more than two terminals. We then investigated some structural properties of directed acyclic networks in the context of communicating the sum of symbols generated at all sources to every terminal. Some of the results in this thesis have been obtained as a result of that investigation. The motivation for further work in this thesis has been to investigate analogous results to the existing results for Type I and Type IA networks. The structural properties of directed acyclic networks, in the context of communicating the sum of symbols generated at all sources to every terminal, are not yet completely understood, and there are still a lot of interesting open problems, to work on, along the lines of the work in this thesis. We would like to explore these open problems in future.

1.4 Contributions of the Thesis

This thesis presents the following contributions.

- We construct two special classes of sum-networks. In the first class, for any finite set of primes, we prove that there exists a sum-network which has a rate-1 linear network code solution for every code length only over finite fields of characteristics belonging to the given set. In the second class, for any finite set of primes, we prove that there exists a sum-network which has a rate-1 linear network code solution for every code length only over finite fields of characteristics which do not belong to the given set. These results show that unlike sum-networks with two sources or two terminal, where a rate-1 linear network code solution can be found over every field (follows from [56]), a rate-1 linear network code solution for sum-networks with more than two sources and more than two terminal may depend on the characteristic of the field.

- It is known that for sum-networks having two sources or two terminal, source-terminal pairs connectivity is a necessary as well as sufficient condition for a rate-1 linear network code solution. We show that there exist sum-networks, with more than two sources and more than two terminals, where it is not possible to communicate the sum of symbols generated at all sources to every terminal using a rate-1 network code of any code length even though there is a path from every source to every terminal. Thus the result for sum-networks with two sources and two terminals can not be generalized for sum-networks having more than two sources and more than two terminals.

- We show that the problem of communicating the sum of symbols generated at all the sources to all the terminals is solvably equivalent (as well as linear solvably equivalent) to communicating any linear function of the symbols if the considered alphabet is a module over a ring and the component linear functions are invertible. It is not known whether a sum-network and its corresponding linear-network, where some of the component linear functions are not invertible, are solvably equivalent.

- We show by explicit construction that for any given directed acyclic Type I network, there exists a sum-network which is solvable (linear solvable respectively) over a finite alphabet if and only if the original network is solvable (linear solvable respectively) over the same finite alphabet. Further, if the Type I network is a multiple-unicast network, then the reverse of the equivalent sum-network is also solvably equivalent (linear solvably equivalent respectively) to the reverse of the multiple-unicast network.

- We show, again by construction, that for any sum-network, there exists a linear solvably equivalent multiple-unicast network. It remains an open problem to prove or disprove that for any sum-network, there exists a solvably equivalent multiple-unicast network.

- Using our construction and an existing result for Type I networks, we prove that for any set of polynomials having integer coefficients there exists a sum-network which is scalar linear solvable over a finite field $\mathbb{F}$ if and only if the polynomials have a common root over $\mathbb{F}$.

- We propose a (k, n) fractional linear network code construction for the reverse network of any network from a (k, n) fractional linear network code for the original network and show that this construction provides a (k, n) fractional linear network code solution for the reverse sum-network if and only if the original code provides a (k, n) fractional linear solution to the original sum-network. This code turns out to be the same as the *dual code* defined in [39] for the special case of of reversibility under scalar linear network coding. However, our treatment is more general and description is more elementary.

- When non-linear codes are permitted, we show that there exists a sum-network which is solvable over a finite alphabet even though the reverse network is not solvable over any finite alphabet.

- We show that linear network coding is not sufficient by constructing a solvable sum-network which does not have a rate-1 linear network code solution over any alphabet for any code length.

- We show that there exists a sum-network whose network coding capacity is not achievable.

- We investigate the network coding capacity of sum-networks. The exact characterization of the network coding capacity seems to be difficult. We present some bounds in terms of number of sources, number of terminals and minimum of min-cut capacities of all source-terminal pairs. We show that, in some special cases, our given bounds are tight.

1.5 Organization of the Thesis

The rest of the thesis is organized as follows. In Chapter 2, we consider the problem of rate-1 linear network code solutions over a finite field. First, we construct sum-networks where rate-1 linear network code solutions depend only on the characteristic of the finite field. Then we show that there exists a sum-network where a scalar linear network code solution not only depends on the characteristic of the finite field but also on the field size. We also show that source-terminal connectivity is insufficient for the rate-1 network code solution for sum-networks.

In Chapter 3, we show solvable equivalence of sum-networks with other type of networks. Specifically, we show that communicating the sum and communicating any linear function of the sources are solvably equivalent under fractional network coding if the considered alphabet is a module over a ring and the component linear functions are invertible. Then we show that there exists a solvably (and linear solvably) equivalent sum-network for any multiple-unicast network (and more generally, for any Type I networks). We also show that there exists a linear solvably equivalent multiple-unicast network for every sum-network.

In Chapter 4, first, we show that for any set of polynomials having integer coefficients, there exists a sum-network which is scalar linear solvable over a finite field $\mathbb{F}$ if and only if the polynomials have a common root in $\mathbb{F}$. Then we show that there exists a solvable sum-network whose reverse network is not solvable. However, we show that a sum-network and its reverse sum-network are solvably equivalent under fractional linear network coding. We also show the insufficiency of linear network coding and unachievability of the network coding capacity for sum-networks.

In Chapter 5, we consider the network coding capacity of sum-networks over a finite field. We present some bounds on the network coding capacity in terms of number of sources, number of terminals, and minimum of min-cut capacities of all source-terminal pairs. In some cases, we show that the presented bounds are tight. We conjecture that the network coding capacity of a sum-network with three sources and three terminals is either $0, 2/3$ or at least 1.

We summarize the thesis with directions for future work in Chapter 6.

Chapter 2

Rate-1 Network Code Solutions

In this chapter, we consider the problem of rate-1 network code solutions of sum-networks over a finite field. We will be dealing mainly with rate-1 linear network code solutions except in the Section 2.4 where we remove linearity restriction and consider non-linear rate-1 network code solutions also. Recall that by solvability (linear solvability respectively) of a network, we mean that the network has a rate-1 network code solution (rate-1 linear network code solution respectively) over some finite alphabet for some code length. It has been shown in [56] that sum-networks with either two sources or two terminals, the sum of symbols generated at all sources can be communicated to every terminal using a scalar linear network code over every finite field if and only if there is at-least one path from every source to every terminal. Note that the sum of symbols generated at all sources can be communicated to every terminal using some non-zero rate network code only if there is at least one path from every source to every terminal. The result in [56] is a very strong result as it proves that the solvability of a sum-network having either two sources or two terminals does not depend on the alphabet field, but only depends on the source-terminal connectivity. The main contribution of this chapter is the construction of sum-networks which are solvable only over finite fields of specific characteristics.

We first address the problem of rate-1 linear network code solutions of sum-networks having more than two sources and more than two terminals. We present two special classes of sum-networks. In the first constructed class, for any finite set of prime numbers, there exists a sum-network which has rate-1 linear network code solutions for every code length only over finite fields of characteristics from the given set. In the second constructed class, for any finite set of prime numbers, there exists a sum-network which has rate-1 linear network code solutions for every code length only over finite fields of characteristics other than those from the given set. In the light of these results, a natural question arises whether all rate-1 linear network code solutions of a sum-network depend only on the characteristic of the finite field and not on the size of the finite field. We answer this question in negation by constructing a sum-network where a scalar linear network code solution not only depends on the characteristic of the finite field but also on the size of the field.

Next, we deal with the question whether having a path from every source to every terminal is sufficient for the linear solvability of sum-networks having more than two sources and more than two terminals. We answer this question in negation by constructing sum-networks which do not have a rate-1 linear network code solution over any field for any code length even though there is a path from every source to every terminal. Interestingly, we show that such sum-networks exist for every $m > 2$ and $n > 2$, where m is the number of sources and n is the number of terminals. Thus the result in [56] does not generalize for sum-networks having more than two sources and more than two terminals in the case of linear network coding.

Finally, we show that having a path from every source to every terminal is not sufficient even for the solvability of sum-networks having more than two sources and more than two terminals. In this case also, there exist sum-networks for all $m > 2$ and $n > 2$ which do not have a rate-1 network code solution over any finite alphabet for any code length. Thus it follows that the result in [56] does not generalize even by using non-linear network codes.

We conclude this chapter by giving a summary of the results in this chapter and posing some open problems based on the work in this chapter.

In this chapter, we denote the finite field alphabet by $\mathbb{F}_q$ where q is the size of the finite field. We drop the subscript q and denote the finite field by $\mathbb{F}$ in case we do not specify the size of the finite field. Further, p, possibly with subscripts, will denote a positive prime integer.

2.1 Linear Network Code Solutions over Finite Fields of Characteristics from a Finite Set of Prime Numbers

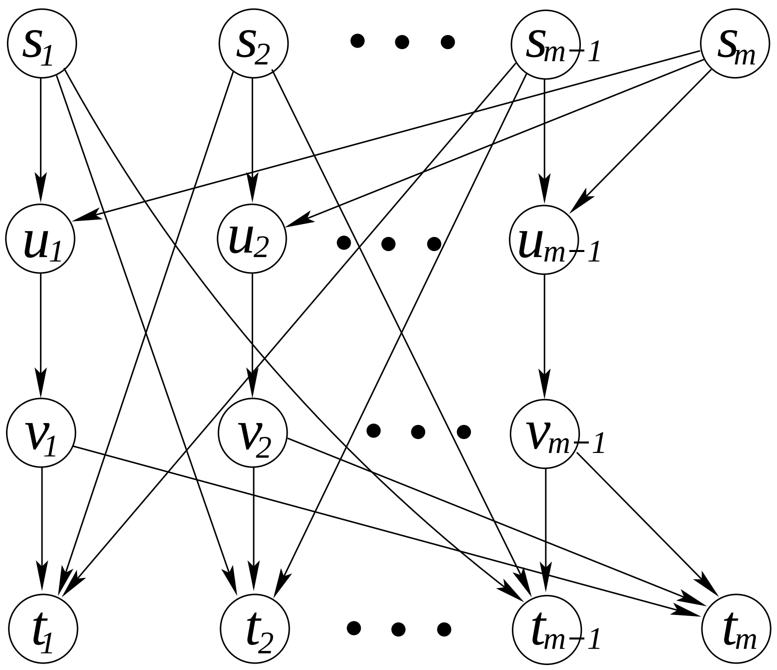

Figure 2.1: The network $\mathcal{S}_m$ where $m = p_1 p_2 \dots p_\lambda + 2$. The sum-network $\mathcal{S}_m$ has rate-1 linear network code solutions for every code length only over finite fields having characteristics from the set $\{p_1, p_2, \dots, p_\lambda\}$.

We show that for any finite set of positive prime numbers $\mathcal{P}$, there exists a sum-network of unit-capacity edges which has a rate-1 linear network code solution for every code length over a finite field if and only if the characteristic of the finite field belongs to $\mathcal{P}$. We prove this result by constructing a network $\mathcal{S}_m$ shown in Fig. 2.1.

The network $\mathcal{S}_m \triangleq (V(\mathcal{S}_m), E(\mathcal{S}_m))$ has four layers of vertices $V(\mathcal{S}_m) = S \cup U \cup V \cup T$. The first layer of nodes are the m sources $S \triangleq \{s_1, s_2, \dots, s_m\}$. The second and third layers have $m-1$ nodes each, and they are denoted as $U \triangleq \{u_1, u_2, \dots, u_{m-1}\}$ and $V \triangleq \{v_1, v_2, \dots, v_{m-1}\}$ respectively. The last layer consists of the m terminals $T \triangleq \{t_1, t_2, \dots, t_m\}$. For every $i = 1, 2, \dots, m-1$, there is an unit capacity edge from s_i to u_i, from u_i to v_i, and from v_i to t_i. That is, $(s_i, u_i), (u_i, v_i), (v_i, t_i) \in E(\mathcal{S}_m)$ for each

$i = 1, 2, \ldots, m-1$. For every $i, j = 1, 2, \ldots, m-1$, $i \neq j$, there is an unit capacity edge from s_i to t_j. Finally, for every $i = 1, 2, \ldots, m-1$, there is an unit capacity edge from s_m to u_i and from v_i to t_m. So, the set of edges is given by

$$\begin{aligned} E(\mathcal{S}_m) &= \cup_{i=1}^{m-1}\{(s_i, u_i), (u_i, v_i), (v_i, t_i)\} \\ &\quad \cup\{(s_i, t_j) : i, j = 1, 2, \ldots, m-1, i \neq j\} \\ &\quad \cup\{(s_m, u_i) : i = 1, 2, \ldots, m-1\} \\ &\quad \cup\{(v_i, t_m) : i = 1, 2, \ldots, m-1\}. \end{aligned}$$

Theorem 1. *For any finite, possibly empty, set $\mathcal{P} = \{p_1, p_2, \ldots, p_\lambda\}$ of positive prime numbers, there exists a sum-network of unit-capacity edges which, for any $k \geq 1$, has a k-length linear network code solution over a finite field if and only if the characteristic of the finite field belongs to $\mathcal{P}$.*

Proof. Define $m = p_1 p_2 \ldots p_\lambda + 2$, where the empty product is assumed to be 1. We prove that the network $\mathcal{S}_m$ satisfies the condition in the theorem. First, note that every source-terminal pair in the network $\mathcal{S}_m$ is connected, which is obviously a necessary condition for being able to communicate the sum of symbols generated at all sources to each terminal over any field using any non-zero rate network code.

First we prove that if, for any code length k, it is possible to communicate the sum of symbols generated at all sources using a k-length linear network code over $\mathbb{F}_q$ to all the terminals in $\mathcal{S}_m$, then the characteristic of $\mathbb{F}_q$ must be in $\mathcal{P}$. Let the message carried by an edge e be denoted by Y_e. For $i = 1, 2, \ldots, m$, the message vector generated by the source s_i be denoted by $X_i \in \mathbb{F}_q^k$. For $i = 1, 2, \ldots, m$, terminal t_i computes a linear combination R_i of the received vectors.

Local coding coefficients used at different layers in the network are denoted by different symbols for clarity. The message vectors carried by different edges and the corresponding local coding coefficients are as below.

$$Y_{(s_i,t_j)} = \alpha_{i,j} X_i \text{ for } 1 \leq i, j \leq m-1, i \neq j, \tag{2.1a}$$

$$Y_{(s_i,u_i)} = \alpha_{i,i} X_i \text{ for } 1 \leq i \leq m-1, \tag{2.1b}$$

$$Y_{(s_m,u_i)} = \alpha_{m,i} X_m \text{ for } 1 \leq i \leq m-1, \tag{2.1c}$$

$$Y_{(u_i,v_i)} = \beta_{i,1} Y_{(s_i,u_i)} + \beta_{i,2} Y_{(s_m,u_i)} \text{ for } 1 \leq i \leq m-1. \tag{2.1d}$$

Here all the coding coefficients $\alpha_{i,j}, \beta_{i,j}, \gamma_{i,j}$ are $k \times k$ matrices over $\mathbb{F}_q$, and the message vectors X_i and the messages $Y_{(.,.)}$ carried by the links are length-k vectors over $\mathbb{F}_q$.

The message vectors computed at terminals are as follows.

$$R_i = \sum_{\substack{j=1 \\ j \neq i}}^{m-1} \gamma_{j,i} Y_{(s_j,t_i)} + \gamma_{i,i} Y_{(v_i,t_i)} \text{ for } 1 \leq i \leq m-1, \tag{2.2a}$$

$$R_m = \sum_{j=1}^{m-1} \gamma_{j,m} Y_{(v_j,t_m)}. \tag{2.2b}$$

Without loss of generality (w.l.o.g.), we assume that

$$Y_{(v_i,t_i)} = Y_{(v_i,t_m)} = Y_{(u_i,v_i)}.$$

By assumption, each terminal decodes the sum of all the sources. That is,

$$R_i = \sum_{j=1}^{m} X_j \text{ for } i = 1, 2, \ldots, m, \tag{2.3}$$

for all values of $X_1, X_2, \ldots, X_m \in \mathbb{F}_q^k$.

From (2.1) and (2.2), we have

$$R_i = \sum_{\substack{j=1 \\ j \neq i}}^{m-1} \gamma_{j,i} \alpha_{j,i} X_j + \gamma_{i,i} \beta_{i,1} \alpha_{i,i} X_i + \gamma_{i,i} \beta_{i,2} \alpha_{m,i} X_m, \tag{2.4}$$

for $i = 1, 2, \ldots, m-1$, and

$$R_m = \sum_{j=1}^{m-1} \gamma_{j,m} \beta_{j,1} \alpha_{j,j} X_j + \sum_{j=1}^{m-1} \gamma_{j,m} \beta_{j,2} \alpha_{m,j} X_m. \tag{2.5}$$

Since (2.3) is true for all values of $X_1, X_2, \ldots, X_m \in \mathbb{F}_q^k$, (2.4) and (2.5) imply

$$\gamma_{j,i} \alpha_{j,i} = I \text{ for } 1 \leq i, j \leq m-1, i \neq j, \tag{2.6}$$

$$\gamma_{i,i} \beta_{i,1} \alpha_{i,i} = I \text{ for } 1 \leq i \leq m-1, \tag{2.7}$$

$$\gamma_{i,i} \beta_{i,2} \alpha_{m,i} = I \text{ for } 1 \leq i \leq m-1, \tag{2.8}$$

$$\gamma_{i,m} \beta_{i,1} \alpha_{i,i} = I \text{ for } 1 \leq i \leq m-1, \tag{2.9}$$

$$\sum_{i=1}^{m-1} \gamma_{i,m} \beta_{i,2} \alpha_{m,i} = I, \tag{2.10}$$

where I denotes the $k \times k$ identity matrix over $\mathbb{F}_q$. All the coding matrices in (2.6), (2.7), (2.8), (2.9) are invertible since the right hand side of the equations are the identity matrices. Equations (2.7) and (2.8) imply $\beta_{i,1}\alpha_{i,i} = \beta_{i,2}\alpha_{m,i}$ for $1 \le i \le m-1$. So, (2.10) gives

$$\sum_{i=1}^{m-1} \gamma_{i,m}\beta_{i,1}\alpha_{i,i} = I. \tag{2.11}$$

Now, using (2.9), we get

$$\begin{aligned} \sum_{i=1}^{m-1} I &= I, && (2.12)\\ \Rightarrow \quad (m-1)I &= I, && (2.13)\\ \Rightarrow \quad (m-2)I &= \mathbf{0}. && (2.14) \end{aligned}$$

This is true if and only if the characteristic of $\mathbb{F}_q$ divides $m-2$. So, the sum of symbols generated at all sources can be communicated to all the terminals in $\mathcal{S}_m$ by a k-length linear network code only if the characteristic of $\mathbb{F}_q$ belongs to $\mathcal{P}$.

Now, if the characteristic of $\mathbb{F}_q$ belongs to $\mathcal{P}$, then for any code length k, every coding matrix in (1.2)-(1.6) can be chosen to be the identity matrix. The terminals then can recover the sum of symbols generated at all sources by taking the sum of the incoming messages, i.e., by taking $\gamma_{j,i} = I$ for $1 \le j \le m-1$ and $1 \le i \le m$ in (1.5) and (1.7). This completes the proof. ■

Corollary 2. *For any prime number p, the sum-network network $\mathcal{S}_{p+2}$ has a rate-1 linear network code for every code length only over finite fields of characteristic p.*

When the set $\mathcal{P}$ is empty, Theorem 1 gives, as a special case, a network where the the sum of symbols generated at all sources can not be communicated to every terminal using a rate-1 linear network code over any field for any code length even though every source-terminal pair is connected.

Corollary 3. *In the network $\mathcal{S}_3$ (see Fig. 2.2), the sum of symbols generated at all sources can not be communicated to every terminal using a rate-1 linear network code over any finite field.*

It was proved in [56] that if $m < 3$ or $n < 3$, then any network where every source-terminal pair is connected allows the sum of symbols generated at all sources to be communicated to every terminal by scalar linear network coding. The network $\mathcal{S}_3$, shown in Fig. 2.2, is an example of a sum-network with $m, n > 2$ where the sum of the sources can not be communicated to the terminals using any rate-1 linear network code even though every source-terminal pair is connected. This example was also found independently by Rammoorthy and Langberg [59].

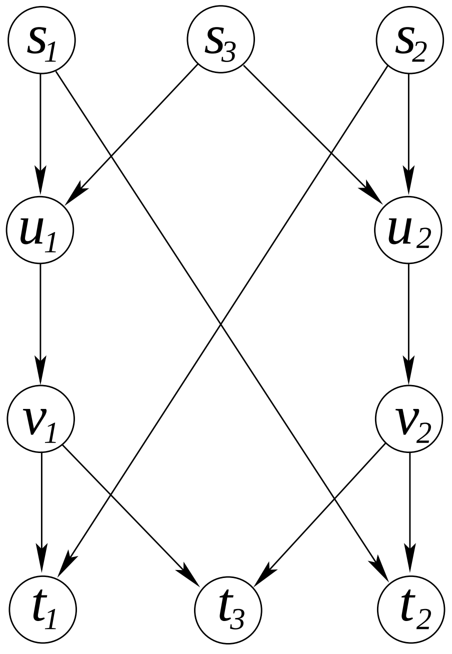

Figure 2.2: A sum-network $\mathcal{S}_3$ which does not have any rate-1 linear network code solution over any field for any code length even though every source-terminal pair is connected.

Now we define a method of combining two networks to obtain a larger network. We call this method *crisscrossing*. Let $\mathcal{N}_1$ be a directed acyclic network with some sources $S_1 \subseteq V(\mathcal{N}_1)$ and some terminals $T_1 \subseteq V(\mathcal{N}_1)$. Similarly let $\mathcal{N}_2$ be a directed acyclic network with some sources $S_2 \subseteq V(\mathcal{N}_2)$ and some terminals $T_2 \subseteq V(\mathcal{N}_2)$. We assume that the nodes of $\mathcal{N}_1$ and $\mathcal{N}_2$ are labeled such that $V(\mathcal{N}_1) \cap V(\mathcal{N}_2) = \phi$. The crisscrossed network $\mathcal{N}_1 \bowtie \mathcal{N}_2$ has the node set $V(\mathcal{N}_1 \bowtie \mathcal{N}_2) = V(\mathcal{N}_1) \cup V(\mathcal{N}_2)$, and the edge set $E(\mathcal{N}_1 \bowtie \mathcal{N}_2) = E(\mathcal{N}_1) \cup E(\mathcal{N}_2) \cup (S_1 \times T_2) \cup (S_2 \times T_1)$. That is, other than the edges of $\mathcal{N}_1$ and $\mathcal{N}_2$, their crisscross has edges from the sources of $\mathcal{N}_1$ to the terminals of $\mathcal{N}_2$, and from the sources of $\mathcal{N}_2$ to the terminals of $\mathcal{N}_2$.

We illustrate the idea of crisscrossing with an example. A complete bipartite graph $K_{m,n}$ has m nodes in one partition and n nodes in the other. We assume that the m nodes in one partition are sources and the n nodes in the other partition are the

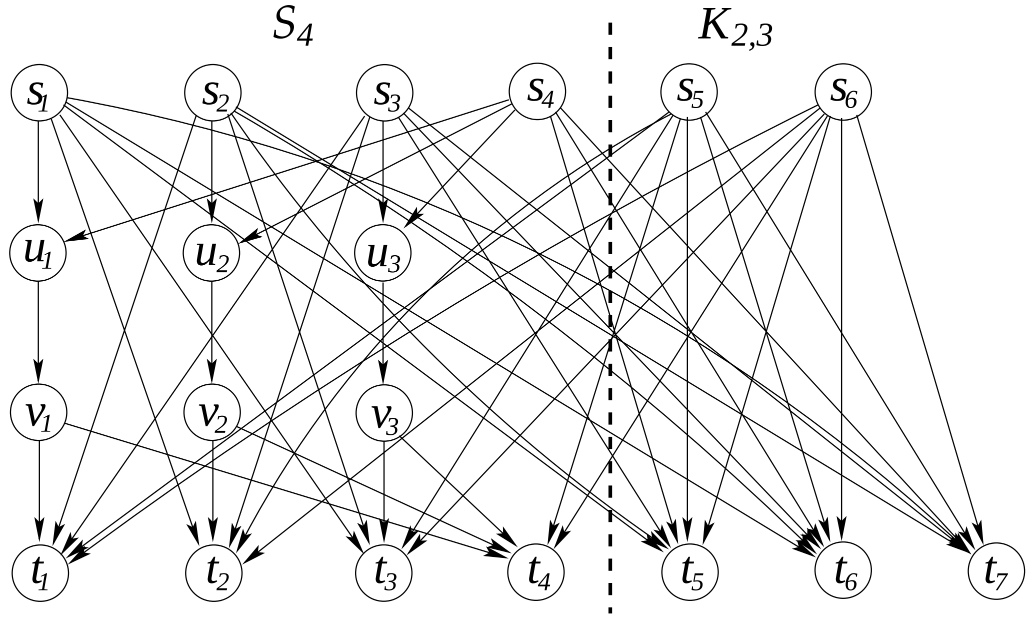

Figure 2.3: The network $\mathcal{S}_4 \bowtie K_{2,3}$.

terminals in the corresponding network. All the edges are assumed to be unit capacity edges which are directed from the source partition to the terminal partition. Clearly, in the network $K_{m,n}$, each source can broadcast its message to all the terminals and each terminal can thus recover any function of the source messages per network use. In particular, each terminal can recover the sum of the source messages per network use. In other words, sum-network $K_{m,n}$ is scalar linear solvable over every finite field. The crisscross of the networks $\mathcal{S}_4$ and $K_{2,3}$ is shown in Fig. 2.3.

Let $\mathcal{P}(\mathcal{N})$ denotes the set of characteristics of fields such that the sum-network $\mathcal{N}$ has a rate-1 linear network code solution for every code length only over a finite field having characteristic from the set $\mathcal{P}(\mathcal{N})$. We have the following results.

Theorem 4. *For any two sum-networks $\mathcal{N}_1$ and $\mathcal{N}_2$,*

$$\mathcal{P}(\mathcal{N}_1 \bowtie \mathcal{N}_2) = \mathcal{P}(\mathcal{N}_1) \cap \mathcal{P}(\mathcal{N}_2).$$

Proof. For any two networks $\mathcal{N}_1$ and $\mathcal{N}_2$, and for any field $\mathbb{F}$, the crisscrossed sum-network $\mathcal{N}_1 \bowtie \mathcal{N}_2$ has a rate-1 linear network code solution for every code length over $\mathbb{F}$ if and only if the individual sum-networks $\mathcal{N}_1$ and $\mathcal{N}_2$ have a rate-1 linear network code solution for every code length over $\mathbb{F}$. So the result follows. ■

Corollary 5. *For any sum-network $\mathcal{N}$, and any positive integers m, n,*

$$\mathcal{P}(\mathcal{N} \bowtie K_{m,n}) = \mathcal{P}(\mathcal{N}).$$

For any positive integer m, let $\Pi(m)$ denotes the set of prime factors of m. Then Theorem 1 states that $\mathcal{P}(\mathcal{S}_m) = \Pi(m-2)$. Theorem 4 then gives

Corollary 6. *For any two positive integers m and n,*

$$\mathcal{P}(\mathcal{S}_m \bowtie \mathcal{S}_n) = \Pi(\gcd(m-2, n-2)).$$

Theorem 4 together with the two corollaries allow construction of a large class of sum-networks having arbitrary finite $\mathcal{P}(\mathcal{N})$.

2.2 Linear Network Code Solutions over Finite Fields of Characteristics from a Co-finite Set of Prime Numbers

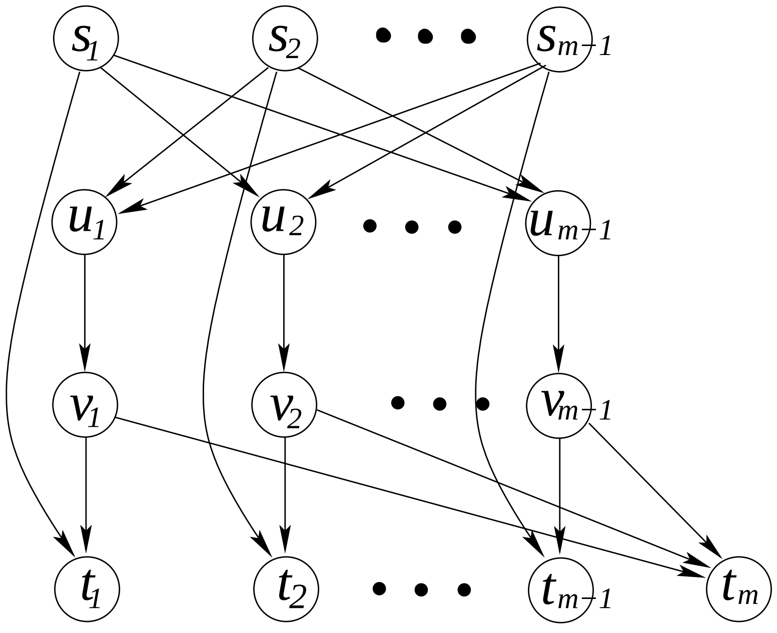

Figure 2.4: The sum-network $\mathcal{S}_m^*$ where $m = p_1 p_2 \dots p_\lambda + 2$. The sum-network $\mathcal{S}_m^*$ has rate-1 linear network code solutions for every code length only over finite fields whose characteristics do not belong the set $\{p_1, p_2, \dots, p_\lambda\}$.

We show that for any finite set of positive prime numbers $\mathcal{P}$, there exists a sum-network of unit-capacity edges which has a rate-1 linear network code solution for every code length over a finite field if and only if the characteristic of the finite field does not belong to $\mathcal{P}$. We prove this result by constructing a network $\mathcal{S}_m^*$ shown in Fig. 2.1. The description of the network $\mathcal{S}_m^*$ is as follows. For $m \geq 3$, the network $\mathcal{S}_m^* \triangleq (V(\mathcal{S}_m^*), E(\mathcal{S}_m^*))$ has four layers of vertices $V(\mathcal{S}_m^*) = S \cup U \cup V \cup T$. The first layer of nodes are the $m-1$ sources $S \triangleq \{s_1, s_2, \ldots, s_{m-1}\}$. The second and third layers have $m-1$ nodes each, and they are denoted as $U \triangleq \{u_1, u_2, \ldots, u_{m-1}\}$ and $V \triangleq \{v_1, v_2, \ldots, v_{m-1}\}$ respectively. The last layer consists of the m terminals $T \triangleq \{t_1, t_2, \ldots, t_m\}$. For every $i = 1, 2, \ldots, m-1$, there is an unit capacity edge from s_i to t_i, u_i to v_i, v_i to t_i, and from v_i to t_m. That is, $(s_i, t_i), (u_i, v_i), (v_i, t_i), (v_i, t_m) \in E(\mathcal{S}_m^*)$ for each $i = 1, 2, \ldots, m-1$. Also for every $i, j = 1, 2, \ldots, m-1$, $i \neq j$, there is an unit capacity edge from s_i to u_j. So, the set of edges is given by

$$\begin{aligned} E(\mathcal{S}_m^*) = \;& \cup_{i=1}^{m-1} && \{(s_i, t_i), (u_i, v_i), (v_i, t_i), (v_i, t_m)\} \\ & \cup && \{(s_i, u_j) : i, j = 1, 2, \ldots, m-1, i \neq j\}. \end{aligned}$$

Theorem 7. *For any finite set $\mathcal{P} = \{p_1, p_2, \ldots, p_\lambda\}$ of positive prime numbers, there exists a sum-network of unit-capacity edges which, for any $k \geq 1$, has a k-length linear network code solution over a finite field if and only if the characteristic of the finite field does not belong to $\mathcal{P}$.*

Proof. Consider the network $\mathcal{S}_m^*$ shown in Fig. 2.4 for $m = p_1 p_2 \ldots p_\lambda + 2$. We will show that this network satisfies the condition of the theorem.

We note that every source-terminal pair in the network $\mathcal{S}_m^*$ is connected which is a necessary condition for being able to communicate the sum of the source messages to each terminal node over any field using any non-zero rate network code.

We now prove that if it is possible to communicate the sum of the symbols generated at all sources using a k-length linear network code over $\mathbb{F}_q$ to all the terminals in $\mathcal{S}_m^*$, then the characteristic of $\mathbb{F}_q$ must not divide $m-2$. Let the message carried by an edge e is denoted by Y_e. For $i = 1, 2, \ldots, m-1$, the message vector generated by the source s_i is denoted by $X_i \in \mathbb{F}_q^k$. Each terminal t_i computes a linear combination R_i of the received vectors.

The message vectors carried by different edges and the corresponding local coding coefficients are as below.

$$\begin{aligned}
Y_{(s_i,t_i)} &= \alpha_{i,i}X_i \text{ for } 1 \le i \le m-1, && (2.15\text{a})\\
Y_{(s_i,u_j)} &= \alpha_{i,j}X_i \text{ for } 1 \le i,j \le m-1, i \neq j, && (2.15\text{b})\\
Y_{(u_i,v_i)} &= \sum_{\substack{j=1\\ j\neq i}}^{m-1} \beta_{j,i}Y_{(s_j,u_i)} \text{ for } 1 \le i \le m-1, && (2.15\text{c})
\end{aligned}$$

and

$$\begin{aligned}
R_i &= \gamma_{i,1}Y_{(s_i,t_i)} + \gamma_{i,2}Y_{(v_i,t_i)} \text{ for } 1 \le i \le m-1, && (2.16\text{a})\\
R_m &= \sum_{j=1}^{m-1} \gamma_{j,m}Y_{(v_j,t_m)}. && (2.16\text{b})
\end{aligned}$$

Here all the coding coefficients $\alpha_{i,j}, \beta_{i,j}, \gamma_{i,j}$ are $k \times k$ matrices over $\mathbb{F}_q$, and the message vectors X_i and the messages carried by the links $Y_{(.,.)}$ are length-k vectors over $\mathbb{F}_q$.

W.l.o.g., we assume that $Y_{(v_i,t_i)} = Y_{(v_i,t_m)} = Y_{(u_i,v_i)}$ and $\alpha_{i,i} = \alpha_{i,j} = I$ for $1 \le i,j \le m-1, i \neq j$, where I denotes the $k \times k$ identity matrix.

By assumption, each terminal decodes the sum of all the source messages. That is,

$$R_i = \sum_{j=1}^{m-1} X_j \text{ for } i = 1,2,\ldots,m \tag{2.17}$$

for all values of $X_1, X_2, \ldots, X_{m-1} \in \mathbb{F}_q^k$.

From (2.15) and (2.16), we have

$$R_i = \sum_{\substack{j=1\\ j\neq i}}^{m-1} \gamma_{i,2}\beta_{j,i}X_j + \gamma_{i,1}X_i \tag{2.18}$$

for $i = 1,2,\ldots,m-1$, and

$$\begin{aligned}
R_m &= \sum_{i=1}^{m-1} \gamma_{i,m} \left(\sum_{\substack{j=1\\ j\neq i}}^{m-1} \beta_{j,i}X_j \right)\\
&= \sum_{j=1}^{m-1} \left(\sum_{\substack{i=1\\ i\neq j}}^{m-1} \gamma_{i,m}\beta_{j,i} \right) X_j. && (2.19)
\end{aligned}$$

Since (2.17) is true for all values of $X_1, X_2, \ldots, X_m \in \mathbb{F}_q^k$, (2.18) and (2.19) imply

$$\gamma_{i,2}\beta_{j,i} = I \text{ for } 1 \leq i, j \leq m-1, i \neq j, \tag{2.20}$$

$$\gamma_{i,1} = I \text{ for } 1 \leq i \leq m-1, \tag{2.21}$$

$$\sum_{\substack{i=1 \\ i \neq j}}^{m-1} \gamma_{i,m}\beta_{j,i} = I \text{ for } 1 \leq j \leq m-1. \tag{2.22}$$

All the coding matrices in (2.20), (2.21) are invertible since the right hand side of the equations are the identity matrices. Equation (2.20) imply $\beta_{j,i} = \beta_{o,i}$ for $1 \leq i, j, o \leq m-1$, $j \neq i \neq o$. So, let us denote all the equal co-efficients $\beta_{j,i}; 1 \leq j \leq m-1, j \neq i$ by β_i. Then (2.22) can be rewritten as

$$\sum_{\substack{i=1 \\ i \neq j}}^{m-1} \gamma_{i,m}\beta_i = I \text{ for } 1 \leq j \leq m-1. \tag{2.23}$$

Equation (2.23) implies

$$\gamma_{i,m}\beta_i = \gamma_{j,m}\beta_j \text{ for } 1 \leq i, j \leq m-1, i \neq j.$$

Then (2.23) gives

$$(m-2)\gamma_{1,m}\beta_1 = I$$
$$\Rightarrow \gamma_{1,m}\beta_1 = (m-2)^{-1}I. \tag{2.24}$$

Equation (2.24) implies that the matrix $\gamma_{1,m}\beta_1$ is a diagonal matrix and all the diagonal elements are equal to $(m-2)^{-1}$. But the inverse of $(m-2)$ exists over the finite field if and only if the characteristic of the field does not divide $(m-2)$. So, the sum of the source messages can be communicated in $\mathcal{S}_m^*$ by a k-length linear network code over F_q only if the characteristic of $\mathbb{F}_q$ does not divide $(m-2)$.

Now, if the characteristic of $\mathbb{F}_q$ does not divide $(m-2)$, then for any code length k, every coding matrix in (2.15a)-(2.15c) can be chosen to be the identity matrix. The terminals $t_1, t_2, \cdots, t_{m-1}$ then can recover the sum of the source messages by taking the sum of the incoming messages, i.e., by taking $\gamma_{i,1} = \gamma_{i,2} = I$ for $1 \leq i \leq m-1$ in (2.16a). Terminal t_m recovers the sum of the source messages by taking $\gamma_{i,m}, 1 \leq i \leq m-1$ in (2.16b) as diagonal matrices having diagonal elements as inverse of $(m-2)$. The inverse of $(m-2)$ exists over F_q because the characteristic of F_q does not divide $(m-2)$. This completes the proof. ■

Corollary 8. *For any prime number p, the sum-network $\mathcal{S}^*_{p+2}$ has a rate-1 linear network code solution for every code length only over fields whose characteristic is different from p.*

Remark 1. *We note that the alphabet field in Theorems 1 and 7 may also be an infinite field of non-zero characteristic. In particular, the theorems also apply to the field of rationals $F_q(X)$ over F_q. So, the sum-network in Theorem 1 is also solvable using linear convolutional network code over F_q if and only if the characteristic of F_q is in $\mathcal{P}$ and the sum-network in Theorem 7 is also solvable using linear convolutional network code over F_q if and only if the characteristic of F_q is not in $\mathcal{P}$ respectively. Please refer [35] for the definition of convolutional network code.*

2.3 Dependency on the Alphabet Size under Scalar Linear Network Coding

In this section, we show that there exists a sum-network where a scalar linear network code solution not only depends on the characteristic of the alphabet field but also on the alphabet size. First we construct a sum-network $\mathcal{X}_3$ which does not admit a scalar linear network code solution over $\mathbb{F}_2$, while it admits a scalar linear network code solution over every other finite field. Then we show that sum-network $\mathcal{S}_4 \bowtie \mathcal{X}_3$ has a scalar linear network code solution if and only if the characteristic of the finite field is 2 and the size of the finite field is 2^k, where k is greater than 1. The network $\mathcal{X}_3$ is shown in Fig. 2.5. The sum-network $\mathcal{X}_3$ has a set of vertices $V(G_1) \triangleq \cup_{i=1}^{3}\{s_i, u_i, v_i, t_i\}$. The sum-network $\mathcal{X}_3$ has a set of unit capacity edges $E(G_1) \triangleq \{(u_i, v_i)|i = 1, 2, 3\} \cup \{(s_i, u_j), (v_i, t_j)|i, j = 1, 2, 3, i \neq j\}$. The nodes s_1, s_2, s_3 are the sources and the nodes t_1, t_2, t_3 are the terminals in the network. The symbols generated at the sources are denoted by X, Z, and W respectively. R_i is the recovered symbols at the terminal t_i for $1 \leq i \leq 3$.

Now we prove the following result.

Theorem 9. *The sum-network $\mathcal{X}_3$ is scalar linear solvable over all fields other than $\mathbb{F}_2$.*

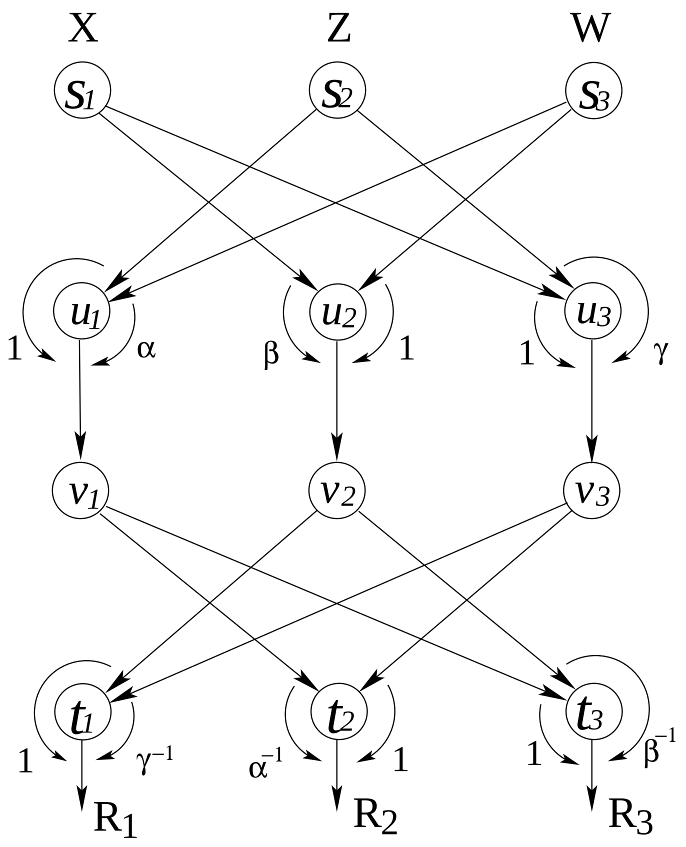

Figure 2.5: The network $\mathcal{X}_3$. $\mathcal{X}_3$ has a scalar linear network code solution over every finite field over than $\mathbb{F}_2$.

Proof. The message carried by different edges and the corresponding local coding coefficients are as below.
W.l.o.g, we assume

$$Y_{(s_1,u_2)} = Y_{(s_1,u_3)} = X, \tag{2.25a}$$

$$Y_{(s_2,u_1)} = Y_{(s_2,u_3)} = Z, \tag{2.25b}$$

$$Y_{(s_3,u_1)} = Y_{(s_3,u_2)} = W, \tag{2.25c}$$

and

$$Y_{(u_1,v_1)} = Y_{(s_2,u_1)} + \alpha Y_{(s_3,u_1)}, \tag{2.26a}$$

$$Y_{(u_2,v_2)} = Y_{(s_3,u_2)} + \beta Y_{(s_1,u_2)}, \tag{2.26b}$$

$$Y_{(u_3,v_3)} = Y_{(s_1,u_3)} + \gamma Y_{(s_2,u_3)}, \tag{2.26c}$$

where $\alpha, \beta, \gamma \in \mathbb{F}_q$.

Also, w.l.o.g, we assume that

$$Y_{(u_1,v_1)} = Y_{(v_1,t_2)} = Y_{(v_1,t_3)}, \tag{2.27a}$$

$$Y_{(u_2,v_2)} = Y_{(v_2,t_1)} = Y_{(v_2,t_3)}, \tag{2.27b}$$

$$Y_{(u_3,v_3)} = Y_{(v_3,t_1)} = Y_{(v_3,t_2)}. \tag{2.27c}$$

Since there is only one path $s_2 \to u_3 \to v_3 \to t_1$ from source s_2 to terminal t_1 and also one path $s_3 \to u_2 \to v_2 \to t_1$ from source s_3 to t_1 with path gains γ and 1 respectively, the recovered symbol R_1 at t_1 must be

$$R_1 = Y_{(v_2,t_1)} + \gamma^{-1} Y_{(v_3,t_1)}. \tag{2.28a}$$

Similarly, the recovered symbols R_2 and R_3 must be

$$R_2 = Y_{(v_3,t_2)} + \alpha^{-1} Y_{(v_1,t_2)}, \tag{2.28b}$$

$$R_3 = Y_{(v_1,t_3)} + \beta^{-1} Y_{(v_2,t_3)}. \tag{2.28c}$$

The coding coefficients are depicted in Fig. 2.5 for clarity.

From (2.25), (2.26), (2.27) and (2.28) it follows that

$$R_1 = (\beta + \gamma^{-1})X + Z + W, \tag{2.29a}$$

$$R_2 = X + (\gamma + \alpha^{-1})Z + W, \tag{2.29b}$$

$$R_3 = X + Z + (\alpha + \beta^{-1})W. \tag{2.29c}$$

Note that (2.29) requires that the coding coefficients α, β and γ be non-zero. This requirement can also be seen as natural since if any of these coefficients is zero, then a particular source-terminal pair will be disconnected.

Since all the terminals must recover the sum of symbols generated at all sources, i.e., $R_1 = R_2 = R_3 = X + Z + W$, we have

$$\beta + \gamma^{-1} = \gamma + \alpha^{-1} = \alpha + \beta^{-1} = 1. \tag{2.30}$$

Now, over the binary field the values of α, β and γ must all be 1. Putting $\alpha = \beta = \gamma = 1$ in (2.30), we have $1 = 0$. This gives a contradiction. Thus it follows that, it is not possible to communicate the sum of symbols generated at all sources to every terminal in this network over the binary field $\mathbb{F}_2$ using scalar linear network coding.

Now, we consider any other finite field $\mathbb{F}_q$ $(q \neq 2)$. We show that over $\mathbb{F}_q$, the conditions in (2.30) are satisfied for some choice of α, β, and γ.

Since $q > 2$, let $\alpha \in \mathbb{F}_q$ be any element other than 0 and 1. Also, take $\gamma = 1 - \alpha^{-1}$ and $\beta = (1 - \alpha)^{-1}$. Clearly, they satisfy (2.30). Hence, it is possible to communicate the sum of symbols generated at all sources to every terminal over $\mathbb{F}_q$. ■

Remark 2. *We note that though the sum of the symbols generated at all sources can not be communicated to every terminal in this network by a scalar network code over $\mathbb{F}_2$, it is possible to do so by a k-length linear network code over $\mathbb{F}_2$ using any code length $k > 1$. This follows because it is possible to communicate the sum of the symbols generated at all sources to all the terminals over the extension field $\mathbb{F}_{2^k}$ using a scalar linear network code.*

Now we have the following result for the sum-network $\mathcal{S}_4 \bowtie \mathcal{X}_3$.

Theorem 10. *The sum-network $\mathcal{S}_4 \bowtie \mathcal{X}_3$ has a scalar linear network code solution if and only if the field size is 2^k for $k > 1$.*

Proof. The theorem follows from Theorem 1, Theorem 9 and Theorem 4. ■

2.4 Insufficiency of Source-Terminal Pairs Connectivity for Solvability

We have already shown a sum-network $\mathcal{S}_3$ which does not have any rate-1 linear network code solution over any finite field for any code length even though there is a path from every source to every terminal. Now we present some other examples where source-terminal pairs connectivity is not sufficient for any rate-1 linear network code solution over any finite field for any code length.

Example 2. For any positive $m, n > 3$, one can construct a network by crisscrossing $\mathcal{S}_3$ with a complete bipartite network $K_{m-3,n-3}$ to get a sum-network with m sources and n terminals which does not have any rate-1 linear network code solution over any field for any code length even though every source-terminal pair is connected. The sum-network $\mathcal{S}_3 \bowtie K_{m-3,n-3}$ is shown in Fig. 2.6.

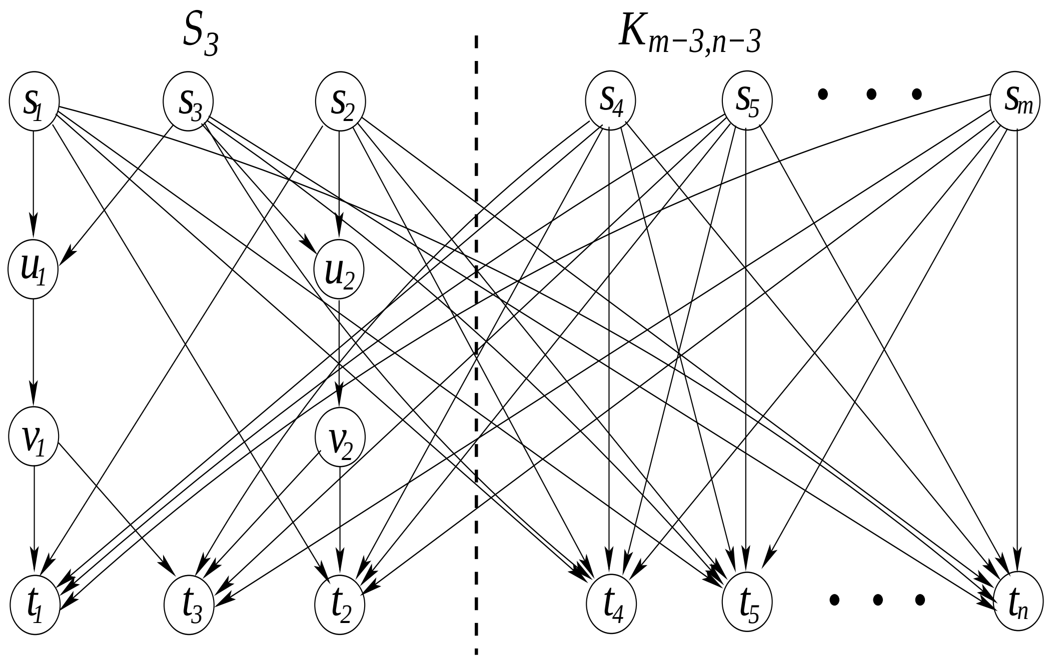

Figure 2.6: The sum-network $\mathcal{S}_3 \bowtie K_{m-3,n-3}$ which does not admit a rate-1 linear network code solution over any field for any code length.

Example 3. A sum-network constructed by crisscrossing sum-networks with disjoint $\mathcal{P}$ does not have a rate-1 linear network code solution over any finite field for any code length. For instance, for any number $m \geq 3$, the sum-network $\mathcal{S}_m \bowtie \mathcal{S}_m^*$ does not have a rate-1 linear network code solution over any finite field for any code length.

Now we remove linearity restriction. We are interested in the question whether having a path from every source to every terminal is sufficient for a rate-1 network code solution over any finite field for any code length for a sum-network. Langberg et al. [59, 57] have shown that the network $\mathcal{S}_3$ shown in Fig. 2.2 does not have a rate-1 network code solution for code length $k = 1$ (scalar network code solution) over any finite field. In fact, as we prove in Chapter 5, the network coding capacity of sum-network $\mathcal{S}_3$ is 2/3 over every finite field.

The sum-network $\mathcal{S}_3'$ shown in Fig. 2.7 is also such a network which has network coding capacity 2/3 over every finite field. We prove this result also in Chapter 5. These results show that there exist sum-networks where having a path from every source to every terminal is not sufficient for a rate-1 network code solution over any finite field for any code length. In fact, such sum-networks exist for the entire class of sum-networks having more than two sources and more than two terminals. Sum-networks constructed for any arbitrary number of sources m and any arbitrary number of terminals n by crisscrossing $\mathcal{S}_3$ or $\mathcal{S}'_3$ with $K_{m-3,n-3}$ do not have a rate-1 network code solution over any finite field for any code length.

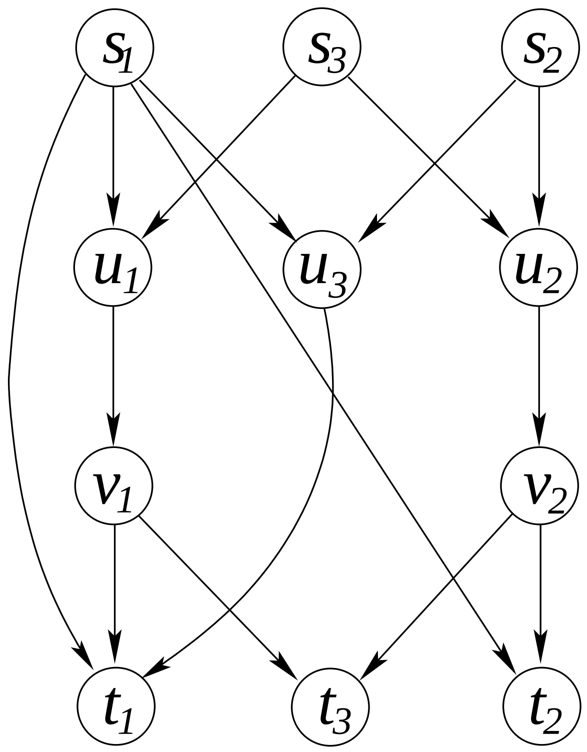

Figure 2.7: The sum-network $\mathcal{S}'_3$ which does not admit a rate-1 network code solution over any alphabet for any code length.

2.5 Summary

In this chapter, we have constructed sum-networks which have a rate-1 linear network code solution for every code length only over finite fields whose characteristics belong to a given finite or co-finite set of prime numbers. We have then shown that the existence of a sum-network where a scalar linear network code solution not only depends on the characteristic of the finite field but also depends on the alphabet size. Finally, we have shown that source-terminal connectivity is insufficient for sum-networks to have a rate-1 network code solution over any finite field for any code length.

2.6 Open Questions

We pose the following questions based on the work in Sections 2.1, 2.2, and 2.3 of this chapter.

- Does there exist a sum-network which has rate-1 linear network code solutions only over finite fields of characteristics from a given set of prime numbers and only for every non-zero integer multiple of a chosen code length k?
- Does there exist a sum-network which has rate-1 linear network code solutions only over finite fields of characteristics other than from a given set of prime numbers and only for every non-zero integer multiple of a chosen code length k?

These questions can be generalized as

- Does there exist a sum-network which has (k, l) fractional linear network code solutions only over finite fields of characteristics from a given set of prime numbers for every choice of $k, l \in \mathbb{Z}_+$?
- Does there exist a sum-network which has (k, l) fractional linear network code solutions only over finite fields of characteristics other than from a given set of prime numbers for every choice of $k, l \in \mathbb{Z}_+$?

We have constructed 3 sum-networks $\mathcal{X}_3$, $\mathcal{S}_3$, and $\mathcal{S}'_3$ having 3 sources and 3 terminals. The sum-network $\mathcal{X}_3$ does not have a scalar linear network code solution over $\mathbb{F}_2$ while it has a scalar linear network code solution over every other finite field. Sum-networks $\mathcal{S}_3$ and $\mathcal{S}'_3$ do not have a rate-1 linear network code solution over any finite field for any code length. Moverover, the sum-networks $\mathcal{S}_3$ and $\mathcal{S}'_3$ also do not have a non-linear rate-1 network code solution. Recently, it is shown in [57] that a sum-network having 3 sources and 3 terminals has a scalar linear network code solution over every finite field if there are two disjoint paths from every source to every terminal. While having two paths from every source to every terminal is a sufficient condition for a scalar linear network code solution over every finite field, it is not a necessary condition. We can easily notice various trivial instances where this is not a necessary condition, for example, any sum-network having exactly one path from every source to every terminal. Based on these observations, we pose the following questions for a sum-network having 3 sources and 3 terminals.

- What is the necessary and sufficient condition for a sum-network having 3 sources and 3 terminals to have a scalar linear solution over every finite field?
- What is the necessary and sufficient condition for linear solvability for a sum-network having 3 sources and 3 terminals?
- What is the necessary and sufficient condition for the solvability for a sum-network having 3 sources and 3 terminals?

These question can be generalized for sum-networks with $m \geq 3, n \geq 3$.

- What is the necessary and sufficient condition for a sum-network with $m \geq 3, n \geq 3$ to have a scalar linear solution over every finite field?
- What is the necessary and sufficient condition for linear solvability for a sum-network with $m \geq 3, n \geq 3$?
- What is the necessary and sufficient condition for the solvability for a sum-network with $m \geq 3, n \geq 3$?

The above questions can be posed for specific finite fields also, namely,

- What is the necessary and sufficient condition for a sum-network with $m \geq 3, n \geq 3$ to have a scalar linear solution over a specific finite field $\mathbb{F}$?
- What is the necessary and sufficient condition for linear solvability over a specific finite field $\mathbb{F}$ for a sum-network with $m \geq 3, n \geq 3$?
- What is the necessary and sufficient condition for the solvability over a specific finite field $\mathbb{F}$ for a sum-network with $m \geq 3, n \geq 3$?

Chapter 3

Solvably Equivalent Networks

In this chapter, we investigate solvable equivalence of sum-networks with other classes of networks. First, we show that linear-networks and sum-networks defined in Chapter 1 are solvably equivalent under fractional network coding if the component linear functions are invertible, i.e., if the component linear functions are invertible, the sum of symbols generated at all sources can be communicated to every terminal using a (k, l) fractional network code over a finite alphabet if and only if such a linear function of symbols generated at all sources can be communicated to every terminal using a (k, l) fractional network code over the same finite alphabet. Then we construct a solvably equivalent (also linear solvably equivalent) sum-network of any given multiple-unicast network. We also show that the reverse sum-network of the constructed sum-network is solvably equivalent (also linear solvably equivalent) to the corresponding reverse multiple-unicast network. Next, we construct a solvably equivalent (also linear solvably equivalent) sum-network of a directed acyclic Type I network. Finally, we construct a linear solvably equivalent multiple-unicast network of a sum-network.

3.1 Equivalence of Linear-Networks and Sum-Networks

In this section, we show that linear-networks and sum-networks are solvably equivalent under fractional network coding if the component linear functions are invertible.

We recall the definition of a linear-network form Chapter 1: consider a network having m sources and n terminals where each source generates one i.i.d. random process with uniform distribution over an alphabet which is an R-module $\mathcal{A}$, where R is a ring. Let $X_i \in \mathcal{A}$ be the message generated at source s_i for $1 \leq i \leq m$. Each terminal requires a linear function of the processes of the form

$$f(X_1, X_2, \ldots, X_m) = f_1(X_1) + f_2(X_2) + \ldots + f_m(X_m), \tag{3.1}$$

where $f_1, f_2, \ldots, f_m \in \text{End}(\mathcal{A})$, $\text{End}(\mathcal{A})$ denotes the set of all R-endomorphism of $\mathcal{A}$. Such a network is called a linear-network.

First, we claim that if the sum of symbols generated at all sources can be communicated to every terminal using a (k, l) fractional network code over $\mathcal{A}$ then any linear function of the symbols generated at all sources, as mentioned in (3.1), can also be communicated to every terminal using a (k, l) fractional network code. Clearly, if the sum of symbols generated at all sources can be communicated to every terminal using a (k, l) fractional network code over $\mathcal{A}$, then any linear function of the symbols can also be communicated to every terminal using the same network code if each source computes its respective component linear function (see Fig. 3.1).

Next, we claim that if all the component linear functions in (3.1) are invertible, then the sum of symbols generated at all sources can be communicated to every terminal using a (k, l) fractional network code over $\mathcal{A}$ if the linear function of symbols generated at all sources can be communicated to every terminal using a (k, l) fractional network code over $\mathcal{A}$. This is because, given a (k, l) fractional network code to communicate the linear function of symbols generated at all sources, each source can compute the inverse of its respective component linear function thereby enabling essentially the same (k, l) fractional network code solution to communicate the sum of symbols generated at all sources to every terminal (see Fig. 3.2).

We note that the equivalence between communicating the sum of symbols and communicating a linear function of symbols holds also under linear network coding.

We also note that it is not clear whether the existence of the inverses of all the component functions is a necessary condition for the equivalence between communicating the sum and communication a linear function. We have not investigated the possibility of equivalence between the problem of communicating the sum of symbols and the problem of communicating any linear function of symbols where some of component functions do not have inverses. It might be possible to prove equivalence in this case by

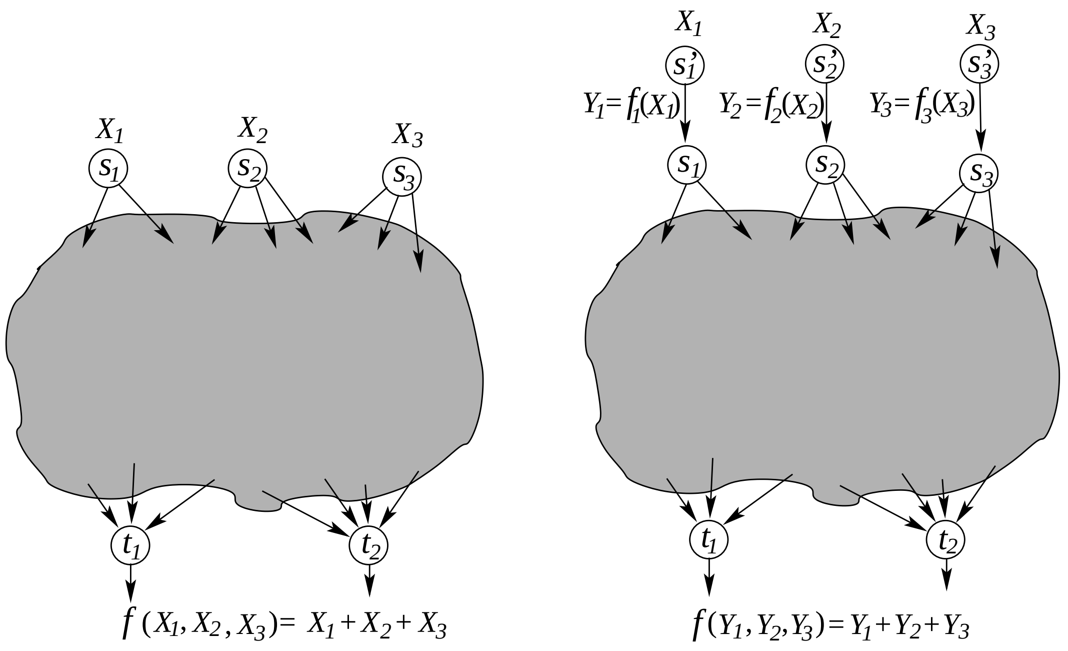

Figure 3.1: A linear function of the symbols can be communicated to every terminal by computing component functions at the sources and by using the same (k, l) fractional network code to communicate the sum of symbols as illustrated in the figure.

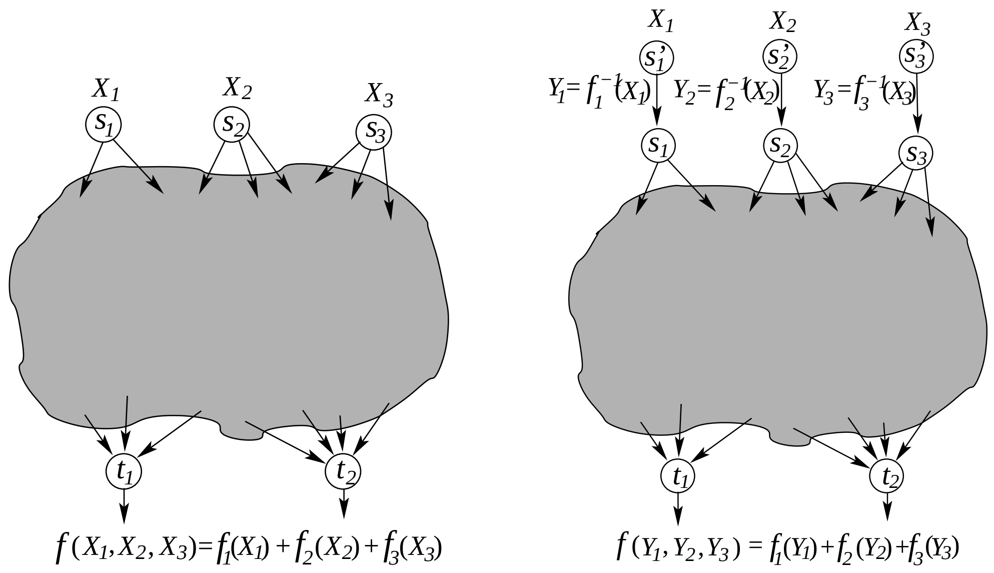

Figure 3.2: The sum of symbols can be communicated to every terminal by computing inverses of component functions at the sources, provided inverse of each component function exists, and by using the same (k, l) fractional network code to communicate the linear function of symbols as illustrated in the figure.

changing the entire network code instead of only computing the respective component functions/inverses of the component functions at the sources.

3.2 C_1 : Construction of a Solvably Equivalent Sum-Network to a given Multiple-Unicast Network

Consider a generic multiple-unicast network $\mathcal{N}_1$ shown in Fig. 3.3. $\mathcal{N}_1$ has m sources $w_1, w_2, \ldots, w_m$ and m corresponding terminals $z_1, z_2, \ldots, z_m$ respectively. Fig. 3.4 shows a sum-network $\mathcal{N}_2$ of which $\mathcal{N}_1$ is a part. In this network, there are $m+1$ sources $s_1, s_2, \ldots, s_{m+1}$ and $2m$ terminals $\{t_{L_i}, t_{R_i} | 1 \leq i \leq m\}$. The reverse networks of $\mathcal{N}_1$ and $\mathcal{N}_2$ are denoted by $\mathcal{N}_1'$ and $\mathcal{N}_2'$ (shown in Fig. 3.5) respectively. The set of vertices and edges of $\mathcal{N}_2$ are respectively

$$\begin{aligned} V(\mathcal{N}_2) &= v(\mathcal{N}_1) \cup \{s_1, s_2, \ldots, s_{m+1}\} \\ &\quad \cup \{u_1, u_2, \ldots, u_m\} \cup \{v_1, v_2, \ldots, v_m\} \\ &\quad \cup \{t_{L_1}, t_{L_2}, \ldots, t_{L_m}\} \cup \{t_{R_1}, t_{R_2}, \ldots, t_{R_m}\} \end{aligned}$$

and

$$\begin{aligned} E(\mathcal{N}_2) &= E(\mathcal{N}_1) \cup \{(s_i, w_i) | i = 1, 2, \ldots, m\} \\ &\quad \cup \{(s_i, u_j) | i, j = 1, 2, \ldots, m, i \neq j\} \\ &\quad \cup \{(s_{m+1}, u_j) | j = 1, 2, \ldots, m\} \\ &\quad \cup \{(s_i, t_{R_i}) | i = 1, 2, \ldots, m\} \\ &\quad \cup \{(u_i, v_i) | i = 1, 2, \ldots, m\} \\ &\quad \cup \{(v_i, t_{L_i}) | i = 1, 2, \ldots, m\} \\ &\quad \cup \{(v_i, t_{R_i}) | i = 1, 2, \ldots, m\} \\ &\quad \cup \{(z_i, t_{L_i}) | i = 1, 2, \ldots, m\}. \end{aligned}$$

The following theorem shows that the networks $\mathcal{N}_1$ and $\mathcal{N}_2$, as well as their reverse networks, are solvably equivalent in a very strong sense. Here $\mathbb{F}$ denotes a field and G denotes an abelian group.

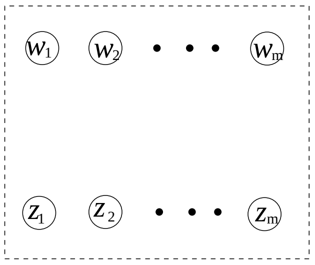

Figure 3.3: A multiple-unicast network $\mathcal{N}_1$

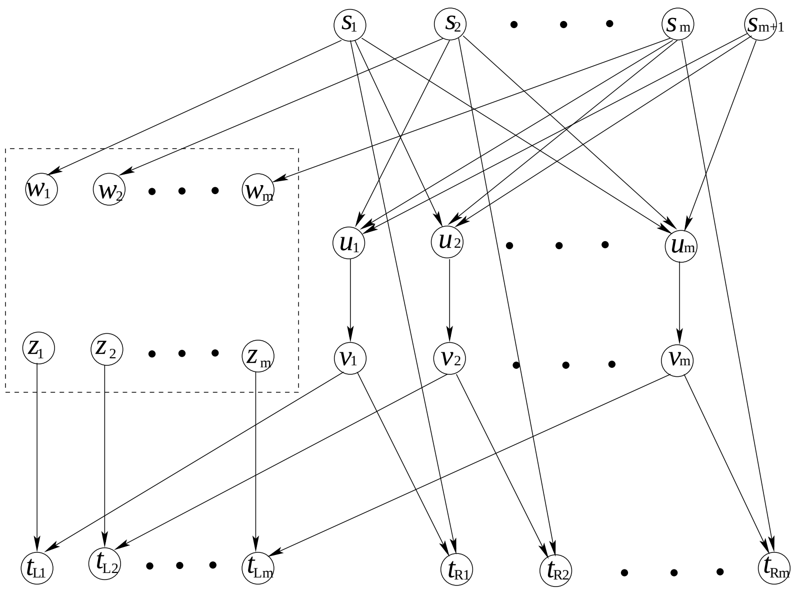

Figure 3.4: The construction (C_1) of a sum-network $\mathcal{N}_2$ from the multiple-unicast network $\mathcal{N}_1$

Theorem 11. *(i) The sum-network $\mathcal{N}_2$ has a k-length linear network code solution over $\mathbb{F}$ if and only if the multiple-unicast network $\mathcal{N}_1$ has a k-length linear network code solution over $\mathbb{F}$.*
(ii) The sum-network $\mathcal{N}_2$ has a k-length network code solution over G if and only if the multiple-unicast network $\mathcal{N}_1$ has a k-length network code solution over G.
(iii) The reverse sum-network $\mathcal{N}_2'$ has a k-length linear network code solution over $\mathbb{F}$ if and only if the reverse multiple-unicast network $\mathcal{N}_1'$ a k-length linear network code solution over $\mathbb{F}$.
(iv) The reverse sum-network $\mathcal{N}_2'$ has a k-length network code solution over G if and only if the reverse multiple-unicast network $\mathcal{N}_1'$ has a k-length network code solution over G.

Proof of part (i) First, we prove that if the multiple-unicast network $\mathcal{N}_1$ has a k-length linear network code solution over $\mathbb{F}$ then the sum-network $\mathcal{N}_2$ also has a k-length linear code solution over $\mathbb{F}$. Let us assume that k generated symbols at the source s_i be grouped to form a k-length vector $X_i \in \mathbb{F}^k$ for $1 \leq i \leq m$. Let us consider a k-length linear network code solution of $\mathcal{N}_1$ over $\mathbb{F}$. Using such a network code, for every $i = 1, \ldots, m$, z_i can recover X_i and forward through the edge (z_i, t_{L_i}). We now extend the code for $\mathcal{N}_1$ to a k-length linear network code for $\mathcal{N}_2$ by taking all the local coding coefficients and decoding coefficients at the terminals in the rest of the network

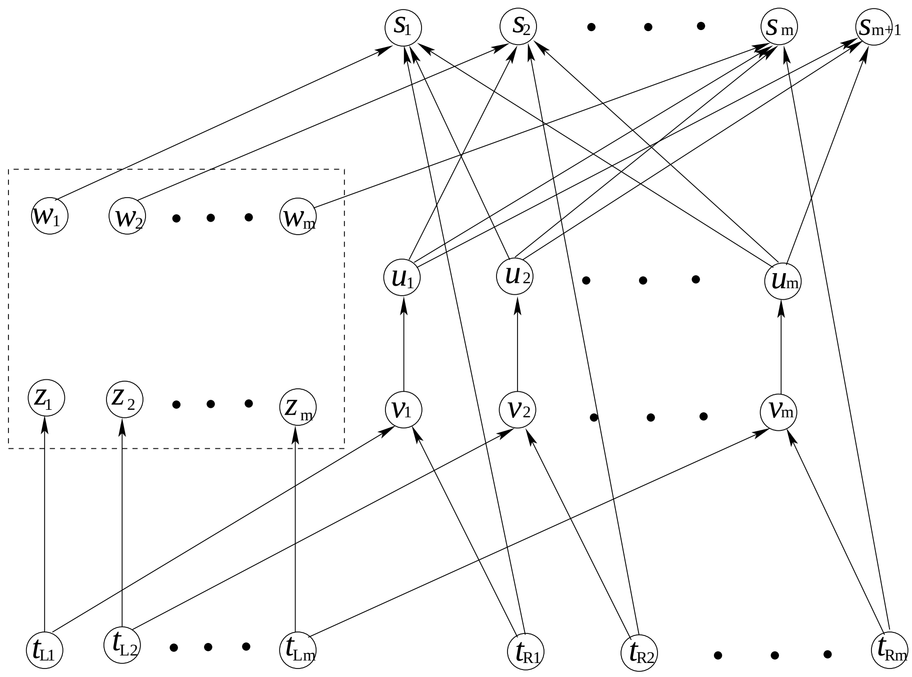

Figure 3.5: The reverse network $\mathcal{N}_2'$ of $\mathcal{N}_2$

to be $k \times k$ identity matrices. Clearly this gives a required solution for $\mathcal{N}_2$.

Now, we prove the converse. We assume that the edge (z_i, t_{L_i}) carries a linear combination of the source symbol vectors $(X_1, X_2, \ldots, X_m)$, i.e., $Y_{(z_i,t_{L_i})} = \sum_{j=1}^{m} \beta_j^i X_j$ for $1 \leq i \leq m$ where $\beta_j^i \in \mathbb{F}^{k\times k}$. We note that s_{m+1} has no edge coming to the network $\mathcal{N}_1$. So, each edge (z_i, t_{L_i}), for $1 \leq i \leq m$, can not have any contribution from the source symbol vector X_{m+1}.

We denote the message carried by the edge e by Y_e as in (1.6) and (1.5) of Chapter 1. For brevity, we denote the decoded messages at the terminal nodes t_{L_i} and t_{R_i} for $i = 1, \ldots, m$ by R_{L_i} and R_{R_i} respectively.

W.l.o.g., we assume that

$$Y_{(u_i,v_i)} = Y_{(v_i,t_{L_i})} = Y_{(v_i,t_{R_i})} \text{ for } 1 \leq i \leq m,$$

$$\text{and } Y_{(s_i,w_i)} = X_i \text{ for } 1 \leq i \leq m.$$

The message carried by different edges and the corresponding local coding coefficients are as follows.

$$\begin{aligned}
Y_{(s_{m+1},u_j)} &= \alpha_{m+1,j}X_{m+1} \text{ for } 1 \leq j \leq m, && \text{(3.2a)}\\
Y_{(s_i,u_j)} &= \alpha_{i,j}X_i \text{ for } 1 \leq i,j \leq m, i \neq j, && \text{(3.2b)}\\
Y_{(s_i,t_{R_i})} &= \alpha_{i,i}X_i \text{ for } 1 \leq i \leq m, && \text{(3.2c)}\\
Y_{(u_i,v_i)} &= \sum_{\substack{j=1\\ j\neq i}}^{m+1} \beta_{j,i}Y_{(s_j,u_i)} \text{ for } 1 \leq i \leq m, && \text{(3.2d)}\\
Y_{(z_i,t_{L_i})} &= \sum_{j=1}^{m} \beta_j^i X_j \text{ for } 1 \leq i \leq m. && \text{(3.2e)}
\end{aligned}$$

The decoded messages at the terminals are as follows.

$$\begin{aligned}
R_{L_i} &= \gamma_{i,L}Y_{(z_i,t_{L_i})} + \gamma'_{i,L}Y_{(v_i,t_{L_i})} \text{ for } 1 \leq i \leq m, && \text{(3.3a)}\\
R_{R_i} &= \gamma_{i,R}Y_{(s_i,t_{R_i})} + \gamma'_{i,R}Y_{(v_i,t_{R_i})} \text{ for } 1 \leq i \leq m. && \text{(3.3b)}
\end{aligned}$$

Here all the coding coefficients are $k \times k$ matrices over $\mathbb{F}$, and the messages carried by the edges $Y_{(.,.)}$ are length k vectors over $\mathbb{F}$.

W.l.o.g., we assume that

$$\alpha_{m+1,i} = \alpha_{i,j} = I \text{ for } 1 \leq i,j \leq m.$$

where I denotes the $k \times k$ identity matrix.

From (3.2) and (3.3), we have

$$\begin{aligned}
R_{L_i} &= \gamma_{i,L}\sum_{j=1}^{m} \beta_j^i X_j + \gamma'_{i,L}\sum_{\substack{j=1\\ j\neq i}}^{m+1} \beta_{j,i}X_j \text{ for } 1 \leq i \leq m, && \text{(3.4a)}\\
R_{R_i} &= \gamma_{i,R}X_i + \gamma'_{i,R}\sum_{\substack{j=1\\ j\neq i}}^{m+1} \beta_{j,i}X_j \text{ for } 1 \leq i \leq m. && \text{(3.4b)}
\end{aligned}$$

By assumption all the terminal nodes can recover the sum of source symbol vectors, i.e. $R_{L_i} = R_{R_i} = \sum_{j=1}^{m+1} X_j$ for $1 \leq i \leq m$. So (3.4) implies

$$\gamma'_{i,L}\beta_{m+1,i} = I \text{ for } 1 \leq i \leq m, \tag{3.5a}$$

$$\gamma_{i,L}\beta_i^i = I \text{ for } 1 \leq i \leq m, \tag{3.5b}$$

$$\gamma_{i,L}\beta_j^i + \gamma'_{i,L}\beta_{j,i} = I \text{ for } i,j = 1,2,\ldots,m, j \neq i, \tag{3.5c}$$

$$\gamma_{i,R} = I \text{ for } 1 \leq i \leq m, \tag{3.5d}$$

$$\gamma'_{i,R}\beta_{m+1,i} = I \text{ for } 1 \leq i \leq m, \tag{3.5e}$$

$$\gamma'_{i,R}\beta_{j,i} = I \text{ for } i,j = 1,2,\ldots,m, j \neq i. \tag{3.5f}$$

All the coding matrices in (3.5a), (3.5b), (3.5d), (3.5e) and (3.5f) are invertible since the right hand side of the equations are the identity matrices. Eq. (3.5a) and (3.5e) together imply

$$\gamma'_{i,L} = \gamma'_{i,R} \text{ for } 1 \leq i \leq m. \tag{3.6}$$

By (3.5f) and (3.6), we have

$$\gamma'_{i,L}\beta_{j,i} = I \text{ for } 1 \leq i,j \leq m, j \neq i. \tag{3.7}$$

By (3.5c) and (3.7), we have

$$\gamma_{i,L}\beta_j^i = \mathbf{0} \text{ for } 1 \leq i,j \leq m,\ j \neq i, \tag{3.8}$$

where $\mathbf{0}$ denotes the all-zero $k \times k$ matrix.

Since $\gamma'_{i,L}$ is invertible by (3.5b), $\beta_j^i = 0$ for $1 \leq i,j \leq m,\ j \neq i$. By (3.5b), $\beta_i^i = 0$ is an invertible matrix for $1 \leq i \leq m$. So, we conclude that for every $i = 1,\ldots,m$, $Y_{z_i,t_{L_i}} = \beta_i^i X_i$, i.e., the edge $Y_{(z_i,t_{L_i})}$ carries only a scaled version of X_i. This implies that for every $i = 1,\ldots,m$, node z_i recovers X_i; which in turn implies that the multiple-unicast network $\mathcal{N}_1$ has a k-length linear network code solution over $\mathbb{F}$. This completes the proof of Part (i).

Proof of part (ii) Now we consider the case when nodes are allowed to do non-linear network coding, i.e., nodes can send any function of the incoming symbols on an outgoing edge. Let us assume that k generated symbols at the source s_i be grouped to form a k-length message vector $X_i \in G^k$ for $1 \leq i \leq m$. For the forward part, let us assume that $\mathcal{N}_1$ has a k-length network code solution over G. Using such a network code, for every $i = 1, \ldots, m$, z_i can recover X_i and forward through the edge (z_i, t_{L_i}). We now extend the code for $\mathcal{N}_1$ to a network code for $\mathcal{N}_2$ where each intermediate/terminal node adds all the incoming messages to compute any outgoing or recovered message. Clearly this gives a required solution for $\mathcal{N}_2$.

Now we prove the "only if" part. Let R_{L_i} and R_{R_i} denote the messages computed at the terminal nodes t_{L_i} and t_{R_i} respectively.

The message carried by different edges are as described below.

W.l.o.g., we assume

$$\begin{aligned}
Y_{(s_j,u_i)} &= X_j \text{ for } 1 \leq i, j \leq m, j \neq i, \\
Y_{(s_{m+1},u_i)} &= X_{m+1} \text{ for } 1 \leq i \leq m, \\
Y_{(s_i,t_{R_i})} &= X_i \text{ for } 1 \leq i \leq m, \\
Y_{(s_i,w_i)} &= X_i \text{ for } 1 \leq i \leq m, \\
Y_{(u_i,v_i)} &= Y_{(v_i,t_{L_i})} = Y_{(v_i,t_{R_i})} \text{ for } 1 \leq i \leq m.
\end{aligned}$$

Further, we assume that

$$Y_{(u_i,v_i)} = f_1^i(X_i, \ldots, X_{i-1}, X_{i+1}, \ldots, X_m, X_{m+1}) \text{ for } 1 \leq i \leq m, \tag{3.9a}$$

$$Y_{(z_i,t_{L_i})} = f_2^i(X_1, \ldots, X_{i-1}, X_i, X_{i+1}, \ldots, X_m) \text{ for } 1 \leq i \leq m. \tag{3.9b}$$

The decoded messages at the terminals are given by.

$$R_{R_i} = g_1^i(Y_{(v_i,t_{R_i})}, Y_{(s_i,t_{R_i})}), \tag{3.10a}$$

$$R_{L_i} = g_2^i(Y_{(v_i,t_{L_i})}, Y_{(z_i,t_{L_i})}). \tag{3.10b}$$

Here all the symbols carried by the links $Y_{(.,.)}$ are from G^k.

We need to show that for every $i = 1, 2, \ldots, m$, communicating the sum of symbols generated at all sources to the terminals t_{L_i} and t_{R_i} is possible only if f_2^i is an $1-1$ function only of the symbol X_i and it does not depend on the other variables.

By (3.10), for every $i = 1, 2, \ldots, m$, functions f_1^i, f_2^i, g_1^i and g_2^i must satisfy the following conditions.

$$g_1^i(f_1^i, X_i) = X_1 + \cdots + X_{i-1} + X_i + X_{i+1} + \cdots + X_m + X_{m+1}, \tag{3.11a}$$

$$g_2^i(f_1^i, f_2^i) = X_1 + \cdots + X_{i-1} + X_i + X_{i+1} + \cdots + X_m + X_{m+1}. \tag{3.11b}$$

Now we prove the following claims for the functions $g_1^i, g_2^i, f_1^i, f_2^i$ for $1 \leq i \leq m$.

Claim 1. *For every $i = 1, 2, \ldots, m$, f_1^i is bijective on each variable X_j, for $1 \leq j \leq m+1, j \neq i$, when the other variables are fixed.*

Proof: Let us consider any $j \neq i$. For any fixed values of $\{X_k | k \neq j\}$, (3.11a) implies that $g_1^i(f_1^i(., X_j, .), .)$ is a bijective function of X_j. This in turn implies that f_1^i is a bijective function of X_j for fixed values of the other variables.

Claim 2. *For every $i = 1, 2, \ldots, m$, $g_1^i(., .)$ is bijective on each argument for any fixed value of the other argument.*

Proof: For any element of G^k, by claim 1, there exists a set of values for $\{X_j | j \neq i\}$ so that the first argument $f_1^i(.)$ of g_1^i takes that value. For such a set of fixed values of $\{X_j | j \neq i\}$, $g_1^i(., X_i)$ is a bijective function of X_i by (3.11a). Now, consider any $j \neq i$ and fix some values for $\{X_k | k \neq j\}$. Again by (3.11a), $g_1^i(f_1^i(., X_j, .), .)$ is a bijective function of X_j. This implies that g_1^i is a bijective function of its first argument for any fixed value of the second argument.

Claim 3. *For every $i = 1, 2, \ldots, m$, f_1^i is symmetric, i.e., interchanging the values of any two variables in its arguments does not change the value of f_1^i.*

Proof: For some fixed values of all the arguments of f_1^i, suppose the value of f_1^i is c_1. We also fix the value of X_i as c_2. Suppose $g_1^i(c_1, c_2) = c_3$. Now we interchange the values of the variables X_j and X_k where $j \neq i \neq k$. Then it follows from Claim 2 and (3.11a) that the value of f_1^i must remain the same.

Claim 4. *For every $i = 1, 2, \ldots, m$, f_2^i is a bijective function of X_i for any fixed values of $\{X_j | j = 1, 2, \ldots, m, j \neq i\}$.*

Proof: For any fixed values of $\{X_j | j = 1, 2, \ldots, m, j \neq i\}$ and X_{m+1}, $g_2^i(., f_2^i(., X_i, .))$ is a bijective function of X_i by equation (3.11b). This implies that f_2^i is a bijective function of X_i for any fixed values of the other arguments.

Claim 5. *For every $i = 1, 2, \ldots, m$, $g_2^i(., .)$ is bijective on each argument for any fixed value of the other argument.*

Proof: By equation (3.11b), $g_2^i(f_1^i(., X_{m+1}), f_2^i(.))$ and $g_2^i(f_1^i(.), f_2^i(., X_i, .))$ are both bijective functions of X_{m+1} and X_i respectively for any fixed values of the omitted variables. This implies that g_2^i is a bijective function of the first and the second argument for the other argument fixed.

Now to prove that for every $i = 1, 2, \ldots, m$, the value of $f_2^i(X_1, \ldots, X_m)$ does not depend on $\{X_j | j = 1, \ldots, m; j \neq i\}$, it is sufficient to prove that for any set of fixed values $X_1 = a_1, \ldots, X_m = a_m$, changing the value of X_j $(j \neq i)$ to any $b_j \in G^k$ does not change the value of f_2^i. Let us assign $X_{m+1} = b_j$. By (3.11b), the value of g_2^i does not change by interchanging the values of X_j and X_{m+1}. Also, the value of f_1^i does not change by this interchange by Claim 3. So, by Claim 5, the value of f_2^i also does not change by this change of value of X_j from a_j to b_j.

Now using Claim 4, it follows that f_2^i is a bijective function of only the variable X_i. This completes the proof of part (ii) of the theorem.

Proof of part *(iii)* First we prove that if the network $\mathcal{N}_1'$ has a k-length linear network code solution over $\mathbb{F}$ then the network $\mathcal{N}_2'$ has also a k-length linear network code solution over $\mathbb{F}$. Let us consider a k-length linear network code solution of $\mathcal{N}_1'$ over $\mathbb{F}$. Using such a network code, for every $i = 1, \ldots, m$, w_i can recover X_{L_i} and forward through the edge (w_i, s_i). We now extend the network code for $\mathcal{N}_1'$ to a k-length linear network code for $\mathcal{N}_2'$ by taking all the local coding coefficient matrices and decoding coefficient matrices at the terminals in the rest of the network $\mathcal{N}_2'$ to be $k \times k$ identity matrices over $\mathbb{F}$. Clearly this gives a required solution for $\mathcal{N}_2'$.

Now, we prove the "only if" part. We assume that for every $i = 1, 2, \ldots, m$, the edge (w_i, s_i) carries a linear combination

$$Y_{(w_i,s_i)} = \sum_{j=1}^{m} \beta_{i,j} X_{L_j}, \tag{3.12}$$

where $\beta_{i,j} \in \mathbb{F}^{k \times k}$.

W.l.o.g., we assume that

$$Y_{(v_j,u_j)} = Y_{(u_j,s_{m+1})} = Y_{(u_j,s_i)} \text{ for } 1 \leq j, i \leq m, j \neq i.$$

The messages carried by different edges and the corresponding local coding coefficients are as below.

$$\begin{aligned}
Y_{(t_{L_j},z_j)} &= \alpha'_{L,j}X_{L_j} \text{ for } 1 \le j \le m, && (3.13a)\\
Y_{(t_{L_j},v_j)} &= \alpha_{L,j}X_{L_j} \text{ for } 1 \le j \le m, && (3.13b)\\
Y_{(t_{R_j},v_j)} &= \alpha_{R,j}X_{R_j} \text{ for } 1 \le j \le m, && (3.13c)\\
Y_{(t_{R_j},s_j)} &= \alpha_j X_{R_j}, \text{ for } 1 \le j \le m, && (3.13d)\\
Y_{(v_j,u_j)} &= \beta_{L,j}Y_{(t_{L_j},v_j)} + \beta_{R,j}Y_{(t_{R_j},v_j)} \text{ for } 1 \le j \le m. && (3.13e)
\end{aligned}$$

For $1 \le i \le m+1$, the decoded message R_i at the terminals s_i are the following.

$$\begin{aligned}
R_i &= \sum_{\substack{j=1\\ j\neq i}}^{m} \gamma_{j,i}Y_{(u_j,s_i)} + \gamma_{i,i}Y_{(t_{R_i},s_i)} + \gamma_i Y_{(w_i,s_i)} \text{ for } 1 \le i \le m, && (3.14a)\\
R_{m+1} &= \sum_{j=1}^{m} \gamma_{j,m+1}Y_{(u_j,s_{m+1})}. && (3.14b)
\end{aligned}$$

Here all the coding coefficients and decoding coefficients are $k \times k$ matrices over $\mathbb{F}$, and the messages carried by the links $Y_{(.,.)}$ are k-length vectors over $\mathbb{F}$.

W.l.o.g., we assume that

$$\alpha'_{L,j} = \alpha_{L,j} = \alpha_{R,j} = \alpha_j = I \text{ for } 1 \le j \le m,$$

where I denotes the $k \times k$ identity matrix.

By (3.13) and (3.14), we have

$$\begin{aligned}
R_i &= \left(\sum_{\substack{j=1\\ j\neq i}}^{m} \gamma_{j,i}(\beta_{L,j}X_{L_j} + \beta_{R,j}X_{R_j}) + \gamma_{i,i}X_{R_i} \right) + \gamma_i \sum_{j=1}^{m} \beta_{i,j}X_{L_j}\\
&\qquad \text{for } 1 \le i \le m, && (3.15a)\\
\text{and } R_{m+1} &= \sum_{j=1}^{m} \gamma_{j,m+1}(\beta_{L,j}X_{L_j} + \beta_{R,j}X_{R_j}). && (3.15b)
\end{aligned}$$

By assumption, all the terminals recover the sum of symbols generated at all sources, i.e., for every $i = 1, 2, \ldots, m$,

$$R_i = R_{m+1} = \sum_{j=1}^{m}(X_{L_j} + X_{R_j}). \qquad (3.16)$$

By (3.15) and (3.16), we have

$$\gamma_{i,i} = I \text{ for } 1 \leq i \leq m, \tag{3.17a}$$

$$\gamma_{j,m+1}\beta_{L,j} = \gamma_{j,m+1}\beta_{R,j} = I \text{ for } 1 \leq j \leq m, \tag{3.17b}$$

$$\gamma_{j,i}\beta_{R,j} = I \text{ for } 1 \leq i,j \leq m, j \neq i, \tag{3.17c}$$

$$\gamma_{j,i}\beta_{L,j} + \gamma_i\beta_{i,j} = I \text{ for } 1 \leq i,j \leq m, j \neq i, \tag{3.17d}$$

$$\gamma_i\beta_{i,i} = I \text{ for } 1 \leq i \leq m. \tag{3.17e}$$

All the coding matrices in (3.17a), (3.17b), (3.17c) and (3.17e) are invertible since the right hand side of the equations are the identity matrix. Equations (3.17b), (3.17c) and (3.17d) imply

$$\gamma_i\beta_{i,j} = \mathbf{0} \;\text{ for } 1 \leq i,j \leq m, j \neq i, \tag{3.18}$$

where $\mathbf{0}$ is the $k \times k$ all-zeros matrix.

Since γ_i is an invertible matrix by (3.17e) for $1 \leq i \leq m$, we have

$$\beta_{i,j} = \mathbf{0} \;\text{ for } 1 \leq i,j \leq m, j \neq i. \tag{3.19}$$

So, for every $i = 1, 2, \ldots, m$, the edge (w_i, s_i) must carry only a scaled version of X_{L_i}, which is possible only if the reverse multiple-unicast network $\mathcal{N}_1'$ has a k-length linear network code solution over $\mathbb{F}$. Further, a k-length linear network code solution $\mathbb{F}$ of $\mathcal{N}_2'$ also gives a k-length linear code solution over $\mathbb{F}$ of $\mathcal{N}_1'$ as a part of it. This completes the proof of part (iii).

Proof of part *(iv)* First, we prove that if the reverse multiple-unicast network $\mathcal{N}_1'$ has a k-length network code solution G, then the sum-network $\mathcal{N}_2'$ has also a k-length network code solution. Consider any k-length network code solution, possibly non-linear, of $\mathcal{N}_1'$ over G. Using such a network code, for every $i = 1, \ldots, m$, w_i can recover X_{L_i} and forward it on the edge (w_i, s_i). We now extend the code for $\mathcal{N}_1'$ to a network code for $\mathcal{N}_2'$ where each intermediate/terminal node adds all the incoming messages to compute any outgoing or recovered message. Clearly this gives a required solution for $\mathcal{N}_2'$.

Now we prove the converse. Consider any k-length network code solution over G for $\mathcal{N}_2'$ where the terminal s_i computes

$$R_i = \sum_{i=1}^{m}(X_{L_i} + X_{R_i}) \text{ for } i = 1, 2, \ldots, m+1. \tag{3.20}$$

The messages carried by different edges are as below.

W.l.o.g., we assume that

$$Y_{(v_i,u_i)} = Y_{(u_i,s_j)} \text{ for } 1 \leq i \leq m,\ \ 1 \leq j \leq m+1, i \neq j. \tag{3.21a}$$

$$Y_{(t_{R_i},s_i)} = X_{R_i} \text{ for } 1 \leq i \leq m, \tag{3.21b}$$

$$Y_{(t_{L_i},v_i)} = X_{L_i} \text{ for } 1 \leq i \leq m, \tag{3.21c}$$

$$Y_{(t_{R_i},v_i)} = X_{R_i} \text{ for } 1 \leq i \leq m. \tag{3.21d}$$

We further assume that

$$Y_{(v_i,u_i)} = f_i(X_{L_i}, X_{R_i}) \text{ for } 1 \leq i \leq m, \tag{3.21e}$$

$$Y_{(w_i,s_i)} = f_i'(X_{L_1}, X_{L_2}, \ldots, X_{L_m}), \text{ for } 1 \leq i \leq m. \tag{3.21f}$$

The decoding operations are denoted as following.

$$R_i = g_i(Y_{(w_i,s_i)}, Y_{(u_1,s_i)}, \ldots, Y_{(u_{i-1},s_i)}, Y_{(u_{i+1},s_i)}, \ldots, Y_{(u_m,s_i)}, Y_{(t_{R_i},s_i)}) \text{ for } 1 \leq i \leq m, \tag{3.22a}$$

$$R_{m+1} = g_{m+1}(Y_{(u_1,s_{m+1})}, \ldots, Y_{(u_m,s_{m+1})}). \tag{3.22b}$$

Now we state some claims which can be proved using similar arguments as in the proof of part (ii) of the theorem. We omit the proof of these claims.

1. As a function of the variables X_{L_i}, X_{R_i}; $i = 1, 2, \ldots, m$, g_{m+1} is bijective in each variable for fixed values of the other variables.

2. The function g_{m+1} is bijective in each variable $Y_{(u_i,s_{m+1})}$ for fixed values of the other variables.

3. For $i = 1, 2, \ldots, m$, $f_i(X_{L_i}, X_{R_i})$ is a bijective function of each variable for a fixed value of the other variable.

4. For $i = 1, 2, \ldots, m$, $f_i(X_{L_i}, X_{R_i})$ is symmetric on its arguments.

5. For $i = 1, 2, \ldots, m$, g_i is bijective on each of its arguments for fixed values of the other arguments.

Now we prove that f_i' is a bijective function of only X_{L_i}, and it is independent of the other variables. Fix a $k \neq i$. It is sufficient to prove that for any fixed values of $X_{L_j}, j = 1, 2, \ldots, m, j \neq k$, the value of f_i' does not change if the value of X_{L_k} is changed from, say, a to b. Let us fix $X_{L_k} = a$. Let us further fix $X_{R_k} = b$ and the variables X_{R_j} for

$j \neq k$ to arbitrary values. Now, by interchanging the values of X_{L_k} and X_{R_k}, the value of g_i does not change, since the sum of the variables does not change. Further, all the arguments of g_i other than f_i' does not change since f_k is symmetric on its arguments. So by claim 5, the value of f_i' also does not change by this interchange. But this means that the value of f_i' does not change by the change of value of X_{L_k} from a to b. This completes the proof of part (iv). ■

Remark 3. *A. Though parts (i) and (iii) are stated for a finite field, the same results can be shown to hold over any finite ring R with identity, R-module with annihilator $\{0\}$, and for more general forms of linear network codes defined in [35].*

B. In parts (ii) and (iv), solvability is not restricted by linear network codes, but includes non-linear coding. The alphabet is restricted to an abelian group simply for defining the sum *of the sources.*

For the case of $k = 1$, Theorem 11 gives the following Corollary.

Corollary 12. *(i) The sum-network $\mathcal{N}_2$ has a scalar linear network code solution over $\mathbb{F}$ if and only if the multiple-unicast network $\mathcal{N}_1$ has a scalar linear network code solution over $\mathbb{F}$.*

(ii) The sum-network $\mathcal{N}_2$ has a scalar network code solution over G if and only if the multiple-unicast network $\mathcal{N}_1$ has a scalar network code solution over G.

(iii) The reverse sum-network $\mathcal{N}_2'$ has a scalar linear network code solution over $\mathbb{F}$ if and only if the reverse multiple-unicast network $\mathcal{N}_1'$ has a scalar linear network code solution over $\mathbb{F}$.

(iv) The reverse sum-network $\mathcal{N}_2'$ has a scalar network code solution over G if and only if the reverse multiple-unicast network $\mathcal{N}_1'$ has a scalar network code solution over G.

Remark 4. *The construction $\boldsymbol{C}_1$ is not the most efficient construction of a solvably equivalent sum-network in terms of the required number of extra nodes and edges. A few extra nodes and edges are added for the neat presentation and visualization. Node s_i can be combined with the node w_i for $1 \leq i \leq m$ without changing the result. Similarly, node t_{L_i} can be combined with the node z_i for $1 \leq i \leq m$ without changing the result. By doing this, a solvably equivalent sum-network can be constructed from a multiple-unicast* *network having m source-terminal pairs by adding $3m+1$ nodes and m^2+4m edges to the multiple-unicast network.*

3.3 C_2 : Construction of a Solvably Equivalent Sum-Network to a given Type I Network

First, we should note that it is possible to construct a solvably equivalent Type IA network from a Type I network by the following two-steps procedure.

1. Consider all the independent random processes generated by the sources in the original Type I network. In the new network, construct one source for each process and add an edge from each constructed source to all the sources in the given Type I network which generate that process. The original sources of the given Type I network are not considered as sources in the constructed network.

2. For every terminal in the original Type I network, construct one terminal for each process required by the original terminal and add an edge from the original terminal to these constructed terminals. The terminals of the original network are not considered as terminals of the new network.

Now, given a Type IA network, Fig. 3.7 shows a sum-network of which the given network is a part. The outer dashed box shows the Type IA network constructed from a Type I network $\mathcal{N}_3$ in the inner dashed box by the above two-steps method. The Type IA network has the sources $w_1, w_2, \ldots, w_m$ generating independent random processes, and for $i = 1, 2, \ldots, m$, each of the terminals $z_1^i, z_2^i, \ldots, z_{n_i}^i$ requires the process generated by w_i. In the constructed sum-network $\mathcal{N}_4$, there are $m+1$ sources $s_1, s_2, \ldots, s_{m+1}$, and $(m + \sum_{i=1}^{m} n_i)$ terminals $\{t_1, t_2, \ldots, t_m\} \cup \{t_j^i | 1 \leq i \leq m, 1 \leq j \leq n_i\}$.

The proof of the following theorem is similar to the proof of Theorem 11(i), (ii), and is omitted.

Theorem 13. *(i) The sum-network $\mathcal{N}_4$ has k-length linear network code solution over a finite field $\mathbb{F}$ if and only if the Type I network $\mathcal{N}_3$ has a k-length linear network code solution over $\mathbb{F}$.*

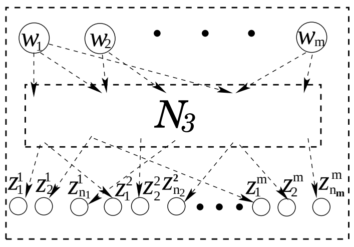
Figure 3.6: A Type IA network constructed from $\mathcal{N}_3$

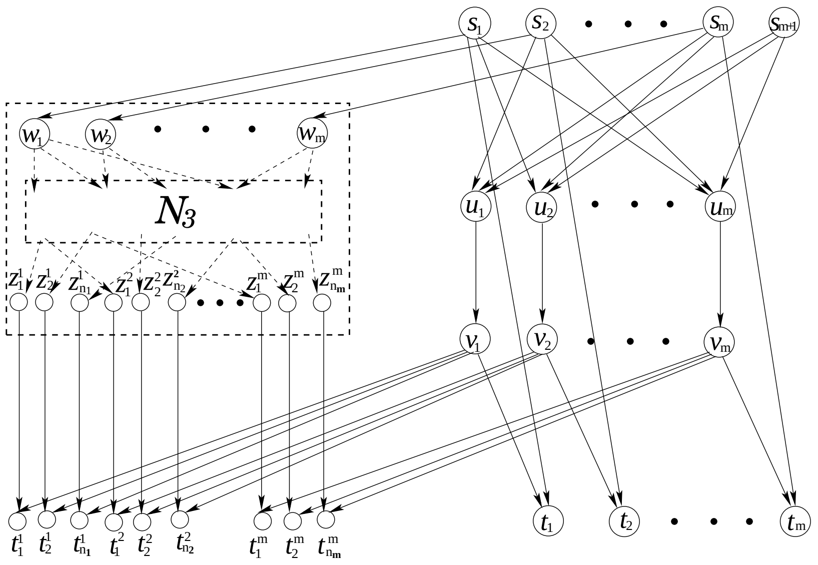
Figure 3.7: The construction (C_2) of a sum-network $\mathcal{N}_4$ from a Type I network $\mathcal{N}_3$

(ii) The sum-network $\mathcal{N}_4$ has a k-length network code solution over an abelian group G if and only if the network $\mathcal{N}_3$ has a k-length network code solution over G.

Restricting to $k = 1$, Theorem 13 gives the following corollary.

Corollary 14. *(i) The sum-network $\mathcal{N}_4$ has a scalar linear network code solution over a finite field $\mathbb{F}$ if and only if the network $\mathcal{N}_3$ has a scalar linear network code solution over $\mathbb{F}$.*

(ii) The sum-network $\mathcal{N}_4$ has a scalar network code solution over an abelian group G if and only if the network $\mathcal{N}_3$ has a scalar network code solution over G.

Now, we consider the network coding capacity of sum-networks obtained by constructions C_1 and C_2. First, we prove the following theorem for the network coding capacity of any sum-network.

Theorem 15. *The network coding capacity of a sum-network is upper bounded by the minimum of min-cut capacities of all source-terminal pairs. That is,*

$$\textit{network coding capacity} \quad \leq \quad \mathit{min}_{i,j}(\text{min-cut capacity } (s_i - t_j)).$$

Proof. For any source s_i of the sum-network, let us fix the source processes of the other sources to the all-zero (”zero” being the identity element of the alphabet group) sequence.

Then the problem of communicating the sum of symbols reduces to the multicast problem from the source s_i to all the terminals, and the network coding capacity of this problem is the minimum of the min-cut capacities from s_i to all the terminals. The overall network coding capacity of the sum-network must be less than or equal to each of these multicast capacities for different i. ■

In the construction C_1, the min-cut capacity from source s_{m+1} to every t_{R_i} for $1 \leq i \leq m$ is 1. So, the network coding capacity of the constructed sum-network by construction C_1 is upper bounded by 1 over any finite alphabet. Similarly, the network coding capacity of the constructed sum-network by construction C_2 is also upper bounded by 1 over any finite alphabet.

Though the construction C_2 (and in particular C_1) gives a solvably equivalent sum-network, the constructed sum-network may have a (k, n) fractional network code solution for $k \leq n$, even though the original network does not have a (k, n) fractional network code solution. So the network coding capacity of the constructed network may be different from the network coding capacity of the original network. For instance, consider the multiple-unicast network where m source-terminal pairs are connected through a single bottleneck link. The multiple-unicast network and the sum-network constructed from it by using construction C_1 are shown in Fig. 3.8. The multiple-unicast network has network coding capacity $1/m$, whereas the constructed sum-network has a $(1, 2)$ fractional linear network code solution. In the first time-slot, the bottle-neck link in the multiple unicast network can carry the sum $X_1 + X_2 + \ldots + X_m$ which is then forwarded to all the m left terminals of the sum network. The links (u_i, v_i) carry only X_{m+1} in the first time-slot. So, by using one time-slot, the left terminals recover the sum $X_1 + X_2 + \ldots + X_{m+1}$. In the second time-slot, the network can obviously be used to communicate the sum $X_1 + X_2 + \ldots + X_{m+1}$ to the right terminals since there is exactly one path between any source and any right terminal.

Even though Construction C_1 does not preserve the network coding capacity, the following results show the relation between the capacity of the sum-network and the capacity of the original network. Here $\mathcal{N}$ denotes a Type-I network, $C_2(\mathcal{N})$ denotes the sum-network constructed from $\mathcal{N}$ using construction C_2, and $Capacity(\mathcal{N})$ denotes the network coding capacity of the network $\mathcal{N}$.

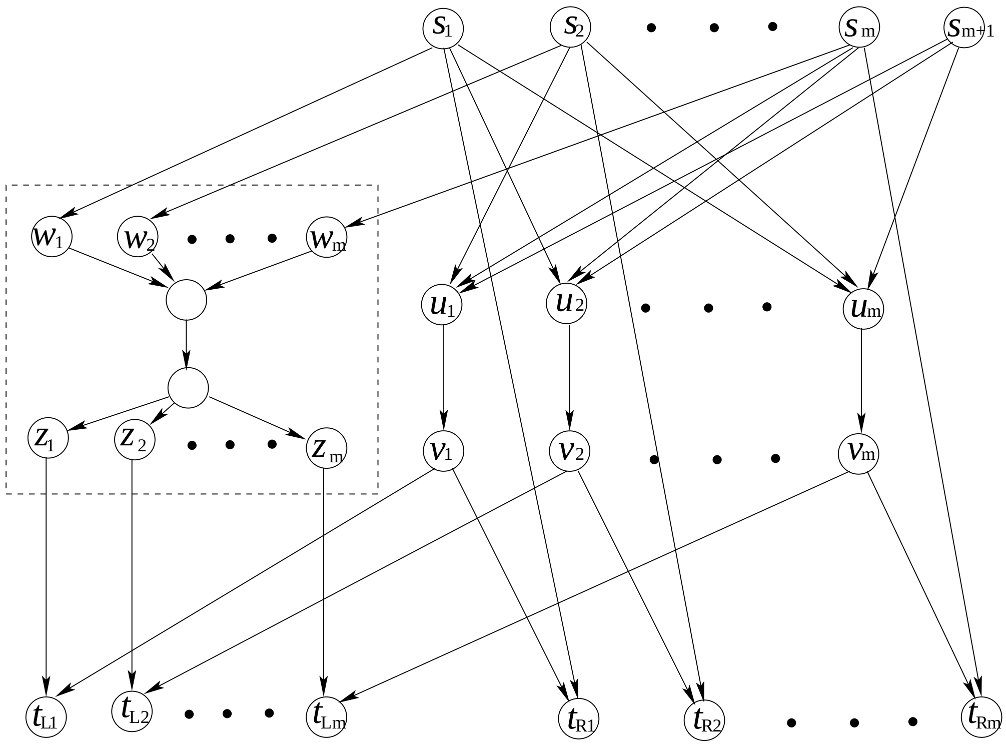

Figure 3.8: A multiple-unicast network with network coding capacity $1/m$ for which Construction C_1 gives a sum-network of network coding capacity $\geq 1/2$

Lemma 16. *For some $k \leq n$, if a network $\mathcal{N}$ has a (k, n) fractional network code (resp. fractional linear network code) solution, then the sum-network $C_2(\mathcal{N})$ also has a (k, n) fractional network code (resp. fractional linear network code) solution.*

Proof. The proof is obvious. ■

Theorem 17.

$$\min(1, Capacity(\mathcal{N})) \leq Capacity(C_2(\mathcal{N})) \leq 1$$

Proof. The first inequality follows from the preceding lemma. The second inequality follows from the fact that the network coding capacity of $C_2(\mathcal{N})$ is upper bounded by 1. ■

Corollary 18. *If the network $\mathcal{N}$ has capacity 1, then the capacity of $C_2(\mathcal{N})$ is also 1.*

3.4 C_3 : Construction of a Linear Solvably Equivalent Multiple-Unicast Network to a given Sum-Network

Consider a generic sum-network $\mathcal{N}$ shown in the dashed box in Fig. 3.9. The sum-network has m sources $w_1, w_2, \ldots, w_m$ and n terminals $z_1, z_2, \ldots, z_n$. The figure shows a multiple-unicast network $C_3(\mathcal{N})$ of which the given sum-network is a part. In this multiple-unicast network, the source-terminal pairs are
$\{(s_i, t_i)|i = 1, 2, \ldots, m\} \cup \{(s_{ij}, t_{ij})|1 \leq i \leq m, 2 \leq j \leq n\}$.

The lower half of the figure is constructed using a method used in [41]. It consists of m chains each with n copies of the network shown in Fig. 3.10 in series. This component network shown in Fig. 3.10 has the property [41] that if t_2 wants to recover the message generated by s_3, and t_1 wants to recover an independent message X_1, then s_1 and s_2 both must send X_1 on the outgoing links.

Theorem 19. *The multiple-unicast network $C_3(\mathcal{N})$ has a k-length linear network code solution over a finite field $\mathbb{F}$ if and only if the sum-network $\mathcal{N}$ is k-length linear network code solution over $\mathbb{F}$.*

Proof. First, if the sum-network $\mathcal{N}$ in the dashed box has a k-length linear network code solution over $\mathbb{F}$, then it is clear that the code can be extended to a code that solves the multiple-unicast network $C_3(\mathcal{N})$. Next, we assume that the multiple-unicast network $C_3(\mathcal{N})$ has a k-length linear network code solution over $\mathbb{F}$, and prove that the sum-network $\mathcal{N}$ has also a k-length linear network code solution over $\mathbb{F}$. It can be seen by similar arguments as used in the proof of [41, Theorem II.1] that all the terminals can recover the respective source symbols if and only if each of the intermediate nodes $r_{ji}; j = 1, 2, \ldots, n, i = 1, 2, \cdots, m$ can recover X_i. So, for any given k-length linear network code solution for $C_3(\mathcal{N})$, the intermediate nodes r_{ji} recover the respective X_i. If $m = 2$ or $n = 2$, then it means that there is a path from each source w_i to each terminal z_j in the sum-network. Then by the results in [56], the sum-network is scalar linear solvable over any field. Now let us assume $m, n \geq 3$.

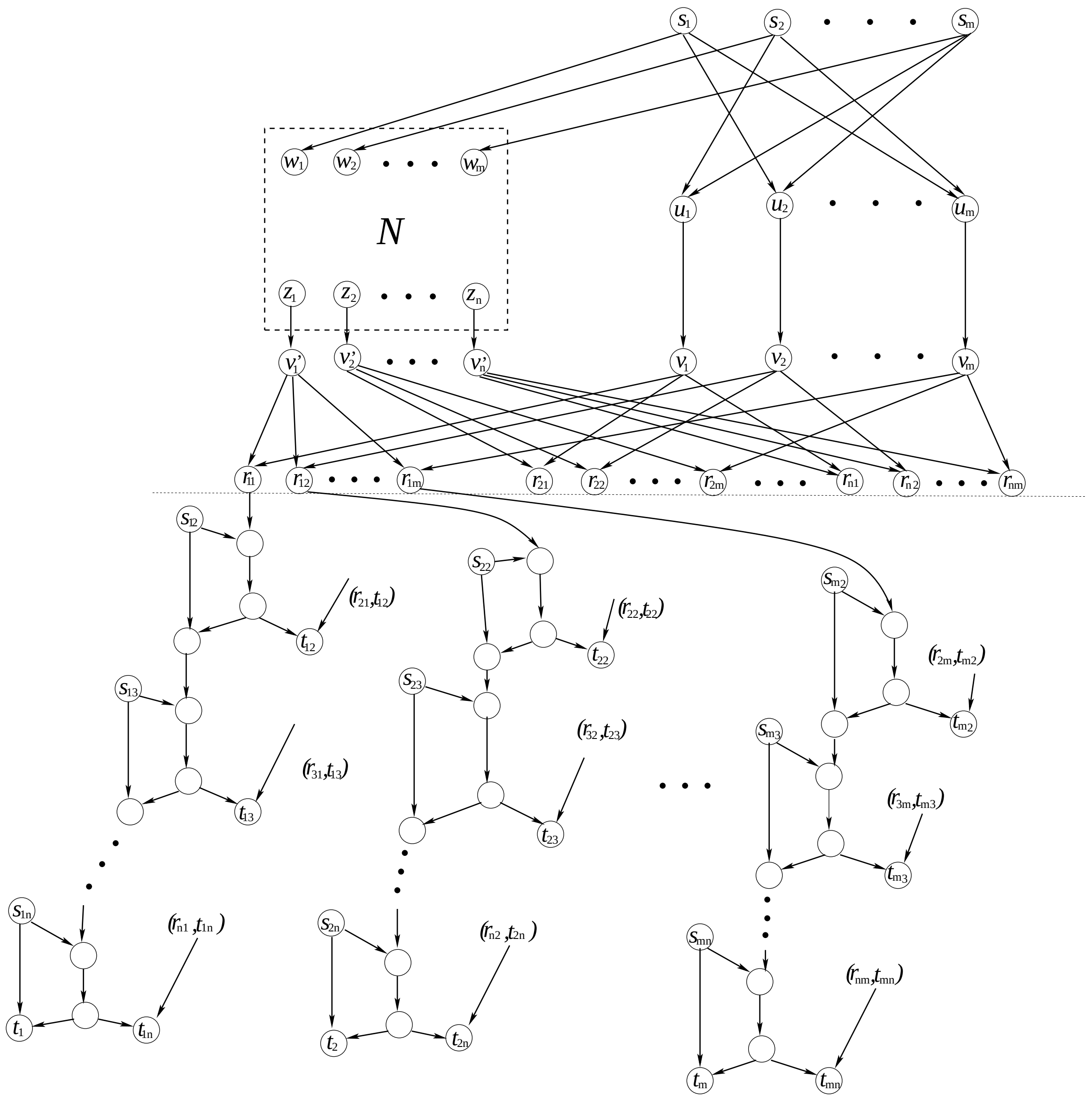

Figure 3.9: Construction (C_3) of a multiple-unicast network $C_3(\mathcal{N})$ from a sum-network $\mathcal{N}$ shown in the dashed box.

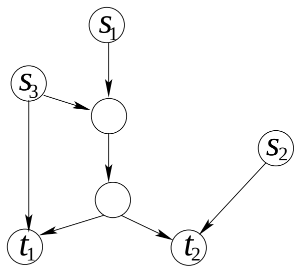

Figure 3.10: A useful component network from [41] that is used in Construction C_3.

W.l.o.g., we assume that

$$\begin{aligned}
&Y_{(s_i,u_j)} = X_i \text{ for } 1 \leq i, j \leq m, i \neq j,\\
&Y_{(s_i,w_i)} = X_i \text{ for } 1 \leq i \leq m,\\
&Y_{(u_i,v_i)} = Y_{(v_i,r_{ji})} \text{ for } 1 \leq i \leq m, 1 \leq j \leq n,\\
&Y_{(z_i,v_i')} = Y_{(v_i',r_{ij})} \text{ for } 1 \leq i \leq n, 1 \leq i \leq m.
\end{aligned}$$

Let us also assume that

$$Y_{(u_i,v_i)} = \sum_{\substack{j=1\\ j\neq i}}^{m} \beta_{ji} Y_{(s_j,u_i)} \text{ for } 1 \leq i \leq m, \tag{3.23}$$

$$Y_{(z_i,v_i')} = \sum_{j=1}^{m} \eta_{ij} X_j \text{ for } 1 \leq i \leq n, \tag{3.24}$$

where $\beta_{ij}, \eta_{ij} \in \mathbb{F}^{k\times k}$.

Let us assume that the symbols recovered at the nodes r_{ij} for forwarding on the outgoing links are

$$\begin{aligned}
R_{ij} &= \gamma_{ij}' Y_{(v_i',r_{ij})} + \gamma_{ij} Y_{(v_j,r_{ij})}\\
&= \gamma_{ij}' \sum_{l=1}^{m} \eta_{il} X_l + \gamma_{ij} \sum_{\substack{l=1\\ l\neq j}}^{m} \beta_{lj} Y_{(s_l,u_j)}\\
&= \gamma_{ij}' \sum_{l=1}^{m} \eta_{il} X_l + \gamma_{ij} \sum_{\substack{l=1\\ l\neq j}}^{m} \beta_{lj} X_l\\
&= \gamma_{ij}' \eta_{ij} X_j + \sum_{\substack{l=1\\ l\neq j}}^{m} (\gamma_{ij}' \eta_{il} + \gamma_{ij} \beta_{lj}) X_l
\end{aligned} \tag{3.25}$$

for $1 \leq i \leq n, 1 \leq j \leq m$. Since the node r_{ij} recovers X_j, we have

$$\gamma'_{ij}\eta_{ij} = I \text{ for } 1 \leq i \leq n, 1 \leq j \leq m, \tag{3.26a}$$

$$\gamma'_{ij}\eta_{il} + \gamma_{ij}\beta_{lj} = 0 \text{ for } 1 \leq i \leq n,\ 1 \leq l, j \leq m,\ l \neq j. \tag{3.26b}$$

It follows from (3.26a) that the matrices γ'_{ij}, η_{ij} are invertible for all i, j. Then it also follows from (3.26b) that the matrices γ_{ij}, β_{lj} are also invertible for i, j, l in their range with $l \neq j$.

We will now prove that $Y_{(z_i, v'_i)}$ for different i are scaled versions of each other. That is, the terminals of the sum-network recover essentially the same linear combination of the sources. For this, we need to prove that for any l, l', $\eta_{il}^{-1}\eta_{il'}$ is independent of i. Let us take a $j \neq l, l'$. This is possible since $m > 2$. Eq. (3.26b) gives

$$\begin{aligned} \eta_{il} &= -\gamma'^{-1}_{ij}\gamma_{ij}\beta_{lj}, \\ \eta_{il'} &= -\gamma'^{-1}_{ij}\gamma_{ij}\beta_{l'j}. \end{aligned}$$

These equations give

$$\eta_{il}^{-1}\eta_{il'} = \beta_{lj}^{-1}\beta_{l'j}, \tag{3.27}$$

and so this is independent of i. This proves that it is possible to communicate a fixed linear combination of the sources through the sum-network in the figure, where each linear coefficient matrix is invertible. If the sources themselves pre-multiply the source messages by the inverse of the respective linear coefficient matrix, then the terminals can recover the sum of the sources. This completes the proof. ■

3.5 Summary

In this chapter, we have shown that linear-networks and sum-networks are solvably equivalent under (k, l) fractional network coding if the component linear functions are invertible. Then we have shown that there exists a solvably equivalent (also linear solvably equivalent) sum-network for every multiple-unicast network. We have proved this result by constructing a sum-network using a generic multiple-unicast network as a part of the sum-network. We have also shown that the reverse sum-network of the

constructed sum-network is solvably equivalent (also linear solvably equivalent) to the corresponding reverse multiple-unicast network. Then we have shown that there exists a solvably equivalent (also linear solvably equivalent) sum-network for every directed acyclic Type I network. Finally, we have shown that there exists a linear solvably equivalent multiple-unicast network for every sum-network.

3.6 Open Questions

In Section 3.1, we have shown the solvable equivalence of sum-networks and linear networks under fractional linear network coding in restricted sense, namely, we have shown that solvable equivalence under fractional linear network coding holds if all the component linear functions are invertible. We pose the following question by removing this restriction.

- Is the problem of communicating the sum of symbols and the problem of communicating a linear function of symbols solvably equivalent under fractional network coding even when some of the component linear functions are not invertible?

This question is important in the sense that a positive result would mean that the results for sum-networks, in this thesis, are not only directly applicable for linear-networks whose all component linear functions are invertible but also for linear-networks whose some component linear functions are not invertible.

We have considered the solvable equivalence in Sections 3.2 and 3.3. We can generalize the considered problems in these sections as:

- Does there exist a linear solvably equivalent sum-network under (k, l) fractional linear network coding for any multiple-unicast network for every choice of k and l?

- Does there exists a solvably equivalent sum-network under (k, l) fractional network coding for any multiple-unicast network for every choice of k and l?

- Does there exist a linear solvably equivalent sum-network under (k, l) fractional linear network coding for any Type I network for every choice of k and l?

- Does there exist a solvably equivalent sum-network under (k, l) fractional network coding for any Type I network for every choice of k and l?
- Does there exist a linear solvably equivalent sum-network under (k, l) fractional linear network coding for any multiple-unicast network such that their reverse networks are also linear solvably equivalent under (k, l) fractional linear network coding for every choice of k and l?
- Does there exist a solvably equivalent sum-network under (k, l) fractional network coding for any multiple-unicast network such that their reverse networks are also solvably equivalent under (k, l) fractional network coding for every choice of k and l?

In Section 3.4, we have proved that there exists a linear solvably equivalent multiple-unicast network for any sum-network. We could not prove some counterpart results of the Section 3.2, namely,

- Does there exist a solvably equivalent multiple-unicast network for any sum-network?
- Does there exist a linear solvably equivalent multiple-unicast network for any sum-network such that their reverse networks are also linear solvably equivalent?
- Does there exist a solvably equivalent multiple-unicast network for any sum-network such that their reverse networks are also solvably equivalent?

These problems can be generalized into the following problems.

- Does there exist a linear solvably equivalent multiple-unicast network under (k, l) fractional linear network coding for any sum-network for every choice of k and l?
- Does there exist a solvably equivalent multiple-unicast network under (k, l) fractional network coding for any sum-network for every choice of k and l?
- Does there exist a linear solvably equivalent multiple-unicast network under (k, l) fractional linear network coding for any sum-network such that their reverse networks are also linear solvably equivalent under (k, l) fractional linear network coding for every choice of k and l?

- Does there exist a solvably equivalent multiple-unicast network under (k, l) fractional network coding for any sum-network such that their reverse networks are also solvably equivalent under (k, l) fractional network coding for every choice of k and l?

Chapter 4

System of Polynomial Equations, Reversibility, Insufficiency of Linear Network Codes and Unachievability of Network Coding Capacity

In this chapter, we prove some results for sum-networks, most of whose counterparts have been previously proved for multiple-unicast networks and/or Type I networks. Using solvable equivalence results in the last chapter, we prove the following results. First, we show that for any set of polynomials having integer coefficients, there exists a sum-network which a scalar linear network code solution over a finite field $\mathbb{F}$ if and only if the polynomials have a common root in $\mathbb{F}$. Next, we show that there exists a solvable sum-network whose reverse sum-network network is not solvable, i.e., there exists a sum-network which is not reversible. Next, we show that there exists a sum-network where linear network codes are insufficient to achieve a rate which is achievable by using non-linear network coding. Next, we show that there exists a sum-network whose network coding capacity is not achievable.

We also present some new results for multiple-unicast networks which are counterpart of the results for sum-networks, proved in Chapter 2. Here, we use linear solvable equivalence result of the Section 3.4.

We also prove one result where we have not used the solvable equivalence result of the last chapter, namely, reversibility of sum-networks and multiple-unicast networks under fractional linear network coding.

4.1 System of Polynomial Equations

It was shown in [27] that for every directed acyclic network, there exists a polynomial collection such that the network is scalar linear solvable over a finite field $\mathbb{F}$ if and only if the polynomials have a common root in $\mathbb{F}$. More interestingly, the converse is also true [36]. It was shown in [36] that for any collection of polynomials having integer coefficients, there exist a directed acyclic network which is scalar linear solvable over $\mathbb{F}$ if and only if the polynomials have a common root in $\mathbb{F}$. It is known that for any directed acyclic network, there exists a (scalar) linear solvably equivalent multiple-unicast network [41]. Thus, for any collection of polynomials having integer coefficients, there exists a directed acyclic multiple-unicast network which has a scalar linear network code solution over $\mathbb{F}$ if and only if the polynomials have a common root in $\mathbb{F}$ [36].

For a specific class of networks, for example multicast networks, there may not exist a network corresponding to a given set of polynomial equations. For example, for the class of multicast networks, there is no network which is scalar linear solvably equivalent to the polynomial equation $2X = 1$. This is because, the polynomial equation has a solution only over fields of characteristic not equal to 2. Whereas, if a multicast network is scalar linear solvable over any field, then it is also scalar linear solvable over large enough fields of characteristic 2.

We claim that the class of sum-networks is broad enough in the sense that for any set of integer polynomial equations, there exists a sum-network which is solvably equivalent under scalar linear network coding. This is simply because, given a system of integer polynomial equations, one can construct a multiple-unicast network, or a Type I network in general, which is solvably equivalent to the integer polynomial equations under scalar linear network coding. Then one can construct, using Construction C_1 or C_2, a sum-network which is in turn scalar linear solvably equivalent to the constructed multiple-unicast network. So, we have the following result.

Theorem 20. *For any set of integer polynomial equations, there exists a sum-network*

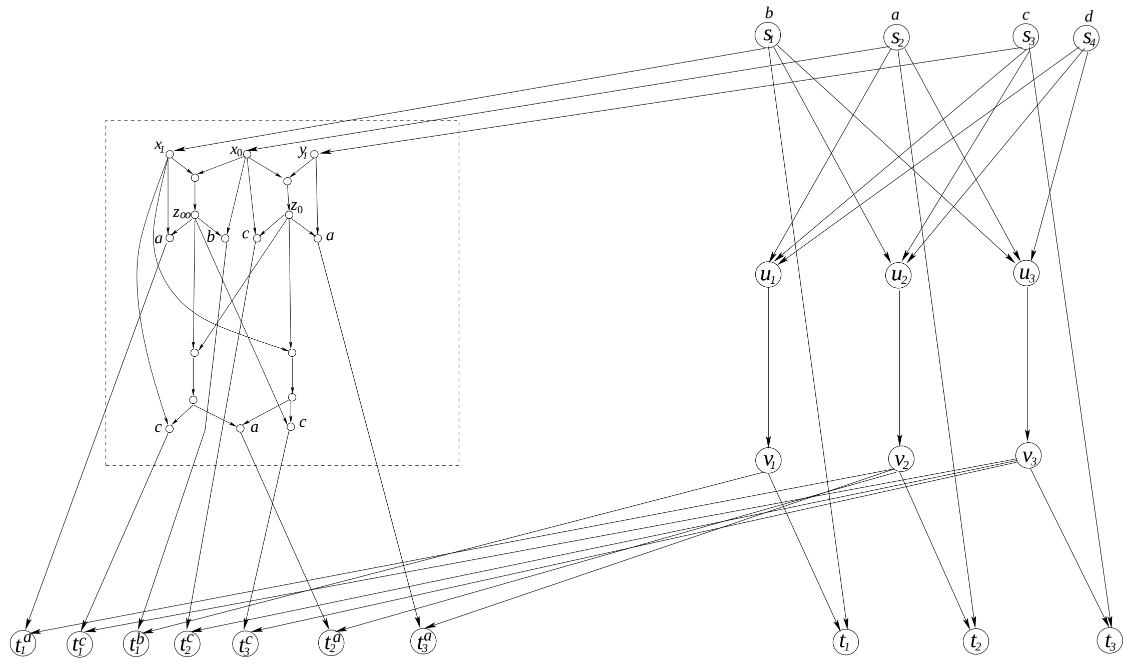

Figure 4.1: A scalar linear solvably equivalent sum-network to the polynomial $P(x) = 2$.

which has a scalar linear network code solution over a finite field $\mathbb{F}$ if and only if the set of integer polynomial equations has a solution in $\mathbb{F}$.

Example 4. *The sum-network shown in Fig. 4.1 has a scalar linear network code solution if and only if the constant polynomial $P(x) = 2$ has a root. The sum-network is constructed using Construction C_2 from a Type I network which has a scalar linear network code solution over $\mathbb{F}$ if and only the polynomial $P(x) = 2$ has a root in $\mathbb{F}$. This Type I is taken from [36] and shown in dashed box. The polynomial $P(x) = 2$ has a solution only over finite fields of characteristic* 2*. This implies that this sum-network in Fig. 4.1 has a scalar linear network code solution only over finite fields of characteristic* 2.

It is known that any set of integer polynomials has solutions only over fields having characteristics from a finite set or co-finite set of prime numbers [37, 36]. This result and the fact that every directed acyclic Type I network has a scalar linear solvably equivalent polynomial collection (which follows from [27]) imply that, a directed acyclic Type I network has a scalar linear network code solution only over fields whose characteristics belong to a finite set or co-finite set of primes. Then, Theorem 19 implies that a sum-network has a scalar linear network code solution only over fields having characteristics from finite set or co-finite set of primes. So, we have the following result.

Theorem 21. *A sum-network has a scalar linear network code solution only over fields whose characteristics belong to a finite set or co-finite set of primes.*

This result resembles with our results for sum-networks $\mathcal{S}_m$ and $\mathcal{S}_m^*$ in Chapter 2. However, a scalar linear network code solution only over fields having characteristics from finite set or co-finite set of primes does not imply a k-length linear network code solution for code length $k > 1$ only over the same fields. The restriction of fields having characteristics from finite set or co-finite set of primes under scalar linear network coding may in general be relaxed for k-length linear network code solution for code length $k > 1$. While Theorem 20 and Theorem 21 present more generalized result for the case of scalar linear network coding, the sum-networks $\mathcal{S}_m$ and $\mathcal{S}_m^*$ are special in the sense that for these networks, the restriction of finite fields having characteristics from finite set or co-finite set of primes does not depend on the code length.

In the following, we construct scalar linear solvably equivalent sum-networks of sum-networks $\mathcal{S}_m$ and $\mathcal{S}_m^*$ respectively. These constructions use the construction method given in [36] for Type I networks and Construction C_2.

Example 5. *We construct a sum-network which has scalar linear network code solution only over finite fields whose characteristics belong to the set of primes $(p_1, p_2, \ldots, p_\lambda)$ by using the construction method given in [36] and Construction C_2, described in the last chapter. The constructed sum-network is shown in Fig. 4.2. First, we construct a scalar linear solvably equivalent Type I network to the polynomial $P(x) = p_1 p_2 \cdots p_\lambda$ by using construction method given in [36]. This Type I network is shown in the dashed box. The sum-network is then obtained using Construction C_2 from this Type I network. The constructed sum-network has scalar linear network code solutions only over finite fields where the polynomial $P(x) = p_1 p_2 \cdots p_\lambda$ has solutions. Since the polynomial $P(x) = p_1 p_2 \cdots p_\lambda$ has solutions only over finite fields whose characteristics are from the set $(p_1, p_2, \ldots, p_\lambda)$, the constructed sum-network in Fig. 4.2 also has scalar linear network code solutions only over finite fields whose characteristics are from the set $(p_1, p_2, \ldots, p_\lambda)$. This sum-network is scalar linear solvably equivalent to $\mathcal{S}_m$, where $m = p_1 p_2 \cdots p_\lambda + 2$.*

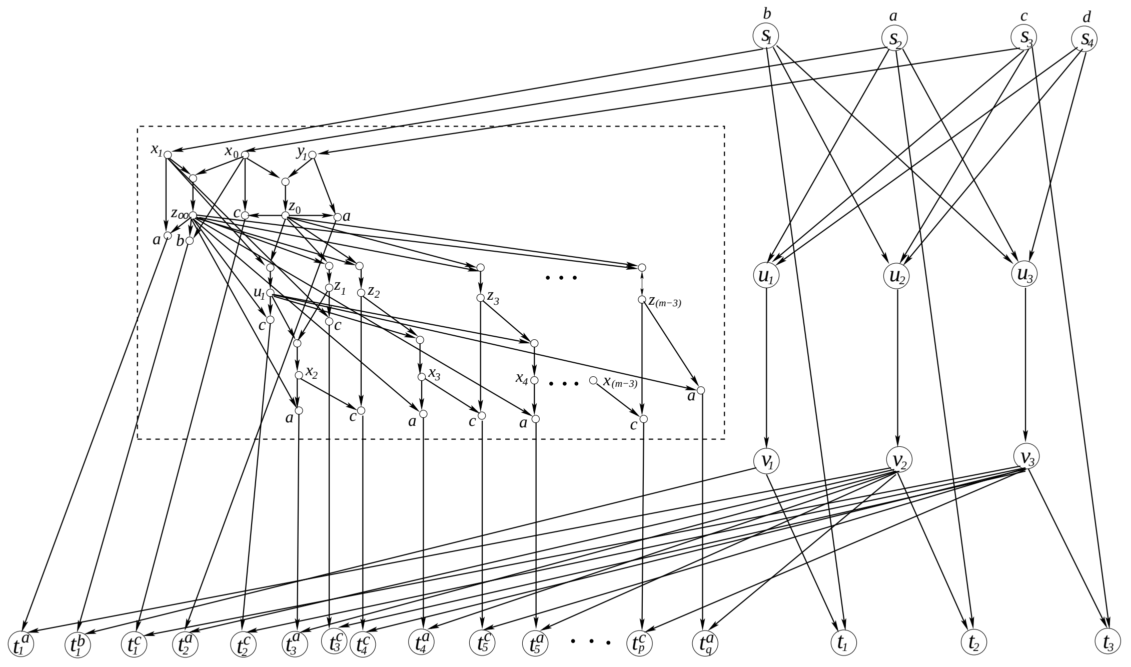

Figure 4.2: A sum-network equivalent to $P(x) = p_1p_2 \cdots p_\lambda$ under scalar linear network coding.

Example 6. *We construct a sum-network which has scalar linear network code solution only over finite fields whose characteristics do not belong to the set of primes* $(p_1, p_2, \ldots, p_\lambda)$*. We construct this sum-network by using the construction method given in [36] and Construction* C_2*. The constructed sum-network is shown in Fig. 4.3. First, we construct a scalar linear solvably equivalent Type I network to the polynomial* $P(x) = (p_1p_2 \cdots p_\lambda)x - 1$ *by using construction method given in [36]. This Type I network is shown in the dashed box. The sum-network is then obtained using Construction* C_2 *from this Type I network. The constructed sum-network has scalar linear network code solutions only over finite fields where the polynomial* $P(x) = (p_1p_2 \cdots p_\lambda)x - 1$ *has solutions. Since the polynomial* $P(x) = (p_1p_2 \cdots p_\lambda)x - 1$ *has solutions only over finite fields whose characteristics do not belong to the set* $(p_1, p_2, \ldots, p_\lambda)$*, the constructed sum-network in Fig. 4.3 also has scalar linear network code solutions only over finite fields whose characteristics do not belong to the set* $(p_1, p_2, \ldots, p_\lambda)$*. This sum-network is scalar linear solvably equivalent to* $\mathcal{S}_m^*$*, where* $m = p_1p_2 \cdots p_\lambda + 2$*.*

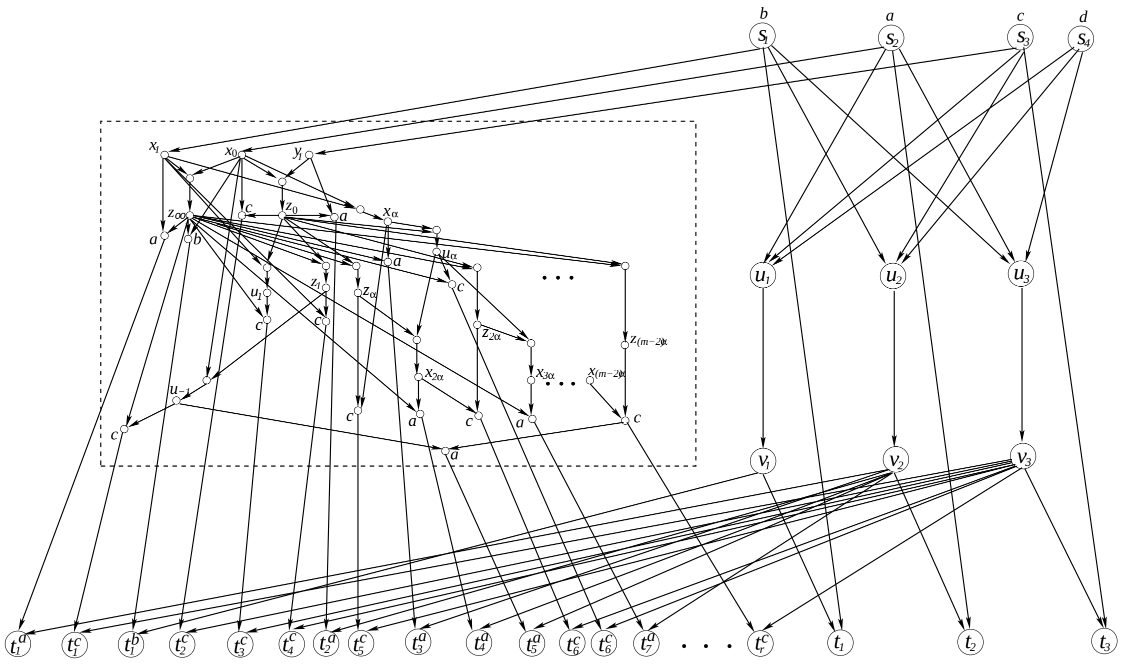

Figure 4.3: A sum-network which has scalar linear network code solutions only over finite fields where the polynomial $P(x) = (p_1 p_2 \dots p_\lambda)x - 1$ has solutions.

4.2 Reversibility of Sum-Networks

Recall that, given a network $\mathcal{N}$, its reverse network $\mathcal{N}'$ is defined to be the network with the same set of vertices, the edges reversed while keeping their capacities same, and the role of sources and terminals interchanged. For a sum-network, since $\mathcal{N}$ may have unequal number of sources and terminals, the number of sources (resp. terminals) in $\mathcal{N}$ and that in $\mathcal{N}'$ may be different. First, we prove that a sum-network has a (k, n) fraction linear network code solution over a finite alphabet if and only if its reverse network has a (k, n) fractional linear network code solution over the same finite alphabet. Then, we show that there exists a non-reversible sum-network.

4.2.1 Reversibility under Linear Network Codes

In the following, for a given network code for a network, we construct a simple corresponding network code for the reverse network and investigate the properties of this new code. We will represent a process generated at a source by an incoming edge at the source, and a process recovered at a terminal by an edge outgoing from the terminal. For any edge (or a source process or a process recovered at a terminal) e, let us denote the corresponding edge in the opposite direction in $\mathcal{N}'$ by $\tilde{e}$. If e is a source process, then in $\mathcal{N}'$, $\tilde{e}$ denotes a recovered process at that terminal, and vice versa. Consider any (k, l) fractional linear network code $\mathcal{C}$ for $\mathcal{N}$. Let the local coding coefficient for any two adjacent edges e, e' be denoted by $\alpha_{e,e'}$. Note that the local coding coefficient for any two adjacent edges is a $l \times l$ matrix; for a source node, the local coding coefficient between a source process and an outgoing edge is a $l \times k$ matrix; and for a terminal node, the local coding coefficient between an incoming edge and a recovered process is a $k \times l$ matrix. Let us consider a path $P = e_1, e_2, \ldots, e_t$ in $\mathcal{N}$, and the corresponding reverse path $\widetilde{P} = \tilde{e}_t, \ldots, \tilde{e}_2, \tilde{e}_1$ in $\mathcal{N}'$. In $\mathcal{N}$, e_1 may also denote a source process which is imagined as an incoming edge to the source, and e_t may be a recovered process at a terminal which is imagined as an outgoing edge from the terminal. The product $\alpha_{e_{t-1},e_t} \cdots \alpha_{e_2,e_3}\alpha_{e_1,e_2}$ is the gain G_P of this path. The gain G_P is a matrix of dimension $l \times l$ if both e_1 and e_t are internal edges of the network, $k \times l$ if e_1 is an internal edge and e_t is a recovered process at a terminal, and $l \times k$ if e_1 is a source process and e_t is an internal edge.

Consider the code $\mathcal{C}'$ for $\mathcal{N}'$ given by the local coding coefficients $\beta_{\tilde{e}',\tilde{e}} = \alpha^T_{e,e'}$, where '$T$' denotes transpose. We call this code as the *canonical reverse code* of $\mathcal{C}$. Under this code, the path gain of $\widetilde{P}$ is

$$G_{\widetilde{P}} = \alpha^T_{e_1,e_2}\alpha^T_{e_2,e_3} \cdots \alpha^T_{e_{t-1},e_t} = G^T_P. \tag{4.1}$$

This code can be shown to be the same as the *dual code* defined by Koetter et al. in [39] under scalar linear network coding. They used this code to show the equivalence of linear solvability and reversibility of multiple-unicast networks and multicast-networks. Though their suggested application for the reverse-multicast network was the centralized detection of an event sensed by exactly one of the many sensors deployed in an area, this is obviously a special application of the resulting sum-network. As explained below, this code is a linear network code solution for the reverse-sum network if the original code is a linear network code solution to the original sum-network.

Consider two cuts χ_1 and χ_2 in $\mathcal{N}$. The first cut may include edges to the sources which correspond to the source processes, and the second cut may include edges out of the terminals corresponding to the recovered processes. Let the edges in χ_1 and χ_2 be $e_{11}, e_{12}, \ldots, e_{1m_1}$ and $e_{21}, e_{22}, \ldots, e_{2m_2}$ respectively. The transfer matrix $T_{\chi_1\chi_2}$ between the two cuts relates the messages carried by the two cuts as

$$\begin{bmatrix} Y_{e_{21}} \\ Y_{e_{22}} \\ \vdots \\ Y_{e_{2m_2}} \end{bmatrix} = T_{\chi_1\chi_2} \begin{bmatrix} Y_{e_{11}} \\ Y_{e_{12}} \\ \vdots \\ Y_{e_{1m_1}} \end{bmatrix}. \tag{4.2}$$

Note that for each i, j, $Y_{e_{ij}}$ is itself a column vector of dimension l or k depending on whether it is an internal edge or an edge for a source process or a process recovered at a terminal. The transfer matrix has appropriate dimension depending on the dimensions of the column vectors on the left hand side and right hand side. It is convenient to view $T_{\chi_1\chi_2}$ as a $m_2 \times m_1$ matrix of blocks of appropriate sizes. The (i, j)-th block element of the matrix has dimension $dim(Y_{e_{2i}}) \times dim(Y_{e_{1j}})$. The (i, j)-th block is the sum of gains of all the paths in the network from e_{1j} to e_{2i}. By (4.1), each path gain is transposed in the reverse network under this code. So, it is clear that the transfer matrix from the cut χ_2 to the cut χ_1 for the code $\mathcal{C}'$ for $\mathcal{N}'$ is given by

$$\widetilde{T}_{\chi_2\chi_1} = T^T_{\chi_1\chi_2}. \tag{4.3}$$

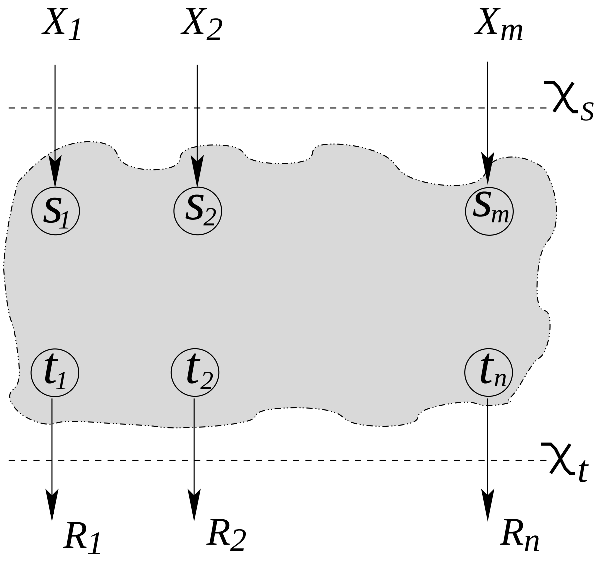

Figure 4.4: The source-cut and the terminal-cut of a generic sum-network.

Consider a generic sum-network $\mathcal{N}$ depicted in Fig. 4.4. Consider the cuts χ_s and χ_t shown in the figure. We call these cuts, the *source-cut* and the *terminal-cut* of the sum-network respectively. The transfer matrix from χ_s to χ_t is an $n \times m$ block matrix $T_{\chi_s\chi_t}$ with each block of size $k \times k$ over the alphabet field. It relates the vectors $\mathbf{X} = (X_1^T, X_2^T, \dots, X_m^T)^T$ and $\mathbf{R} = (R_1^T, R_2^T, \dots, R_n^T)^T$ as $\mathbf{R} = T_{\chi_s\chi_t}\mathbf{X}$. Here X_i, R_j are all column vectors of length k. The (i, j)-th element ('block' for fractional linear network coding) of the transfer matrix is the sum of the path gains of all paths from X_j to R_i. A (k, l) network code provides a rate k/l solution for the sum network if and only if this transfer matrix is the all-identity matrix, i.e., if

$$T_{\chi_s\chi_t} = \begin{bmatrix} I_k & I_k & \cdots & I_k \\ I_k & I_k & \cdots & I_k \\ \vdots & \vdots & \ddots & \vdots \\ I_k & I_k & \cdots & I_k \end{bmatrix}_{nk \times mk} . \tag{4.4}$$

Clearly transposition of this transfer matrix preserves the same structure. For a multiple-unicast network on the other hand, a (k, l) fractional linear network code solution exists if and only if the transfer matrix between the source cut and the terminal cut is

$$T_{\chi_s\chi_t} = \begin{bmatrix} I_k & \mathbf{0}_k & \cdots & \mathbf{0}_k \\ \mathbf{0}_k & I_k & \cdots & \mathbf{0}_k \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_k & \mathbf{0}_k & \cdots & I_k \end{bmatrix}_{mk \times mk} , \tag{4.5}$$

where $\mathbf{0}_k$ denotes the $k \times k$ all-zero matrix. This matrix is symmetric, and so is invariant under transposition.

Lemma 22. *A sum-network $\mathcal{N}$ has a (k, l) fractional linear network code solution if and only if the reverse network $\mathcal{N}'$ also has a (k, l) fractional linear network code solution.*

Proof. Consider a given (k, l) fractional linear network code solution for $\mathcal{N}$. By (4.3) and (4.4), the canonical reverse code for the reverse network $\mathcal{N}'$ also gives a (k, l) fractional linear network code solution for $\mathcal{N}'$. ■

For the case $k = l$, Lemma 22 gives the following Corollary.

Corollary 23. *A sum-network $\mathcal{N}$ has a k-length linear network code solution if and only if the reverse network $\mathcal{N}'$ has a k-length linear network code solution.*

Theorem 24. *A sum-network and its reverse network have the same linear network coding capacity.*

As an alternative to the technique used in this section, one may view (k, l) fractional coding as scalar coding over the "fattened" network obtained by replacing each internal edge by l parallel edges, each link incoming to a source corresponding to a source process by k links signifying k symbols, and each link outgoing from a terminal corresponding to a recovered process by k links signifying k recovered symbols. Then again one may use the same transfer function approach to conclude the same results.

Lemma 22 shows that a sum-network and its reverse network are solvably equivalent under fractional linear network coding. The same result also follows for multiple-unicast network from (4.3) and (4.5), and this was proved in [39, 40].

4.2.2 Nonreversible Sum-Network

If non-linear coding is allowed, then it was shown in [41] that there exists a solvable multiple-unicast network (shown in the dashed box in Fig. 4.5) whose reverse multiple-unicast network is not solvable over any finite alphabet. The network in Fig. 4.5 is obtained by using Construction $\mathbf{C}_1$ on this network. By the properties of Construction $\mathbf{C}_1$, Theorem 11 (ii) and (iv) to be precise, it follows that the network in Fig. 4.5 allows a nonlinear coding solution whereas its reverse sum-network does not have a solution. So, we have,

Theorem 25. *There exists a solvable sum-network whose reverse network is not solvable over any finite alphabet.*

4.3 Insufficiency of Linear Network Codes

Linear codes are proven to be sufficient to achieve the capacity for multicast networks [26, 27, 28]. However, It has been shown that scalar linear network coding is insufficient for Type I networks in the sense that there exists a Type I network which does not admit a scalar linear network code over any finite alphabet while it has a k-length network code

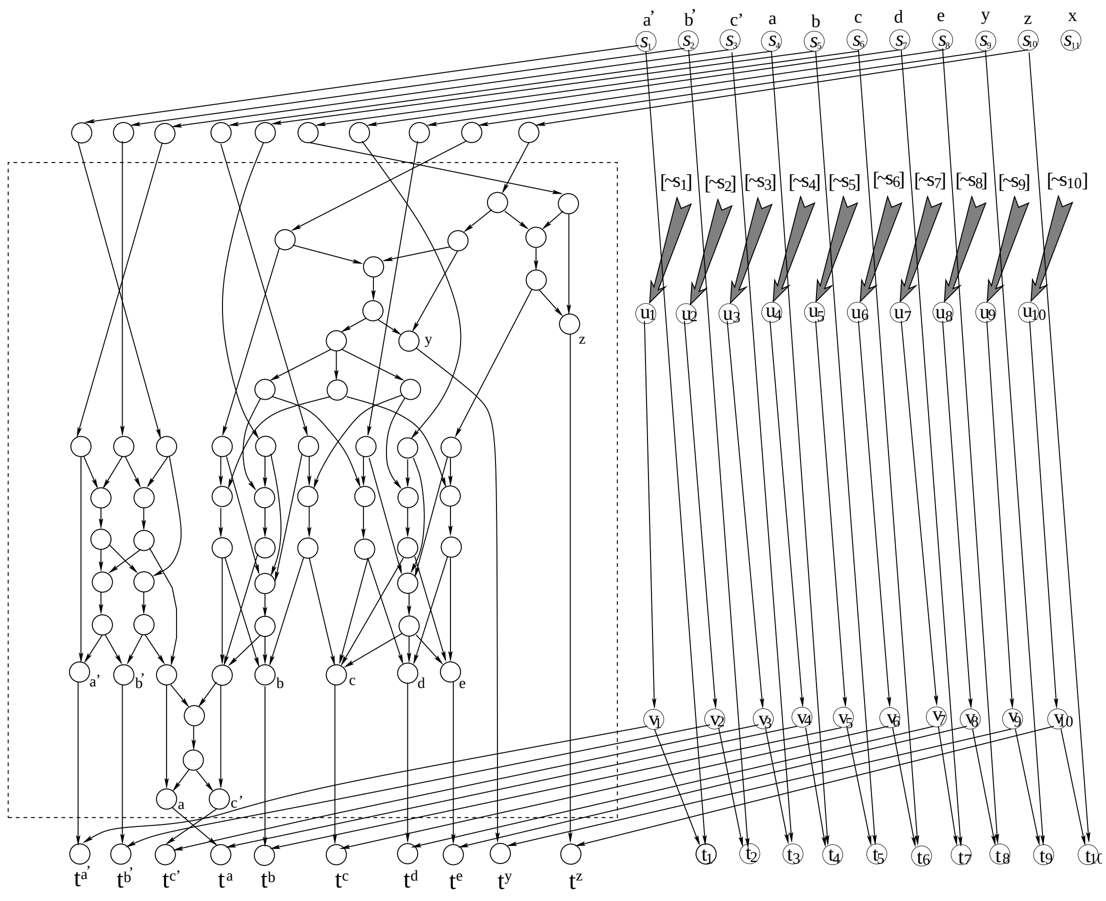
a' b' c' a b c d e y z x
s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11
[~s1] [~s2] [~s3] [~s4] [~s5] [~s6] [~s7] [~s8] [~s9] [~s10]
u1 u2 u3 u4 u5 u6 u7 u8 u9 u10
v1 v2 v3 v4 v5 v6 v7 v8 v9 v10
t1 t2 t3 t4 t5 t6 t7 t8 t9 t10
y z a' b' b c d e a c'
ta' tb' tc' ta tb tc td te ty tz

Figure 4.5: A nonreversible sum-network. The fat arrows incoming to the nodes u_i denote connections from all the sources except s_i. The non-reversible multiple-unicast network in the dashed box is taken from [41]. The sum-network is constructed from this multiple-unicast network using Construction C_1.

solution for code length $k = 2$ [32]. It has also been shown in [34] that even fractional linear network coding is not sufficient for Type I networks in the sense that a rate may be achievable by nonlinear network coding even though the same rate may not be achievable by linear network coding. In this section, first, we show that scalar linear network coding is insufficient for sum-networks. Then, we show that fractional linear network code is also insufficient for sum-networks.

4.3.1 Insufficiency of Scalar Linear Network Codes

We construct a sum-network which does not admit a scalar linear network solution over any finite alphabet while it has a k-length linear network code for code length $k = 2$.

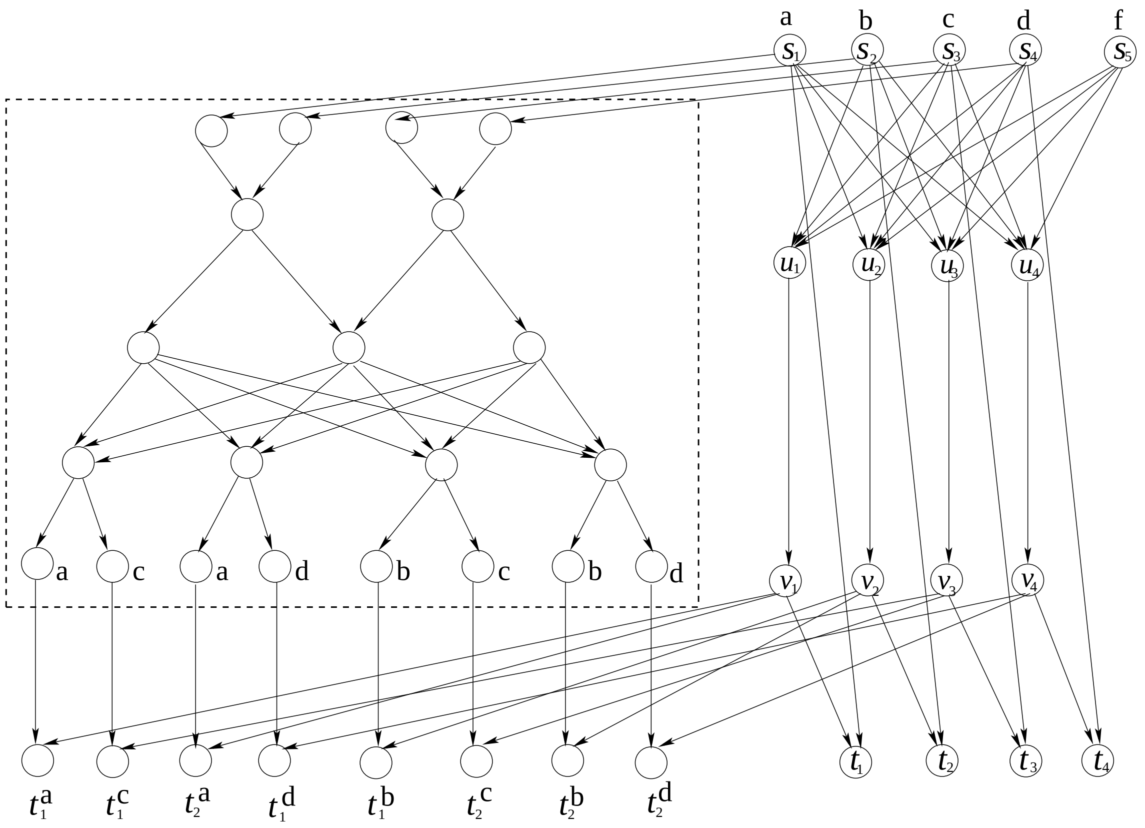

Figure 4.6: A sum-network which has a k-length linear network code solution for code length $k = 2$ while it does not have a scalar linear network code solution over any finite alphabet. This sum-network is constructed using a Type I network given in [32] and using Construction C_2.

We use the Type I network given in [32] and Construction C_2 to construct such a sum-network. The constructed sum-network is shown in Fig. 4.6. The Type I network is shown in the dashed box. This Type I network does not admit a scalar linear network code solution over any alphabet but has a k-length linear network code solution for code length 2 over every finite alphabet. By Corollary 14, we have the following result.

Theorem 26. *There exists a sum-network which admits a k-length linear network code solution for code length $k = 2$ over a finite alphabet while it does not have a scalar linear network code solution over any finite alphabet.*

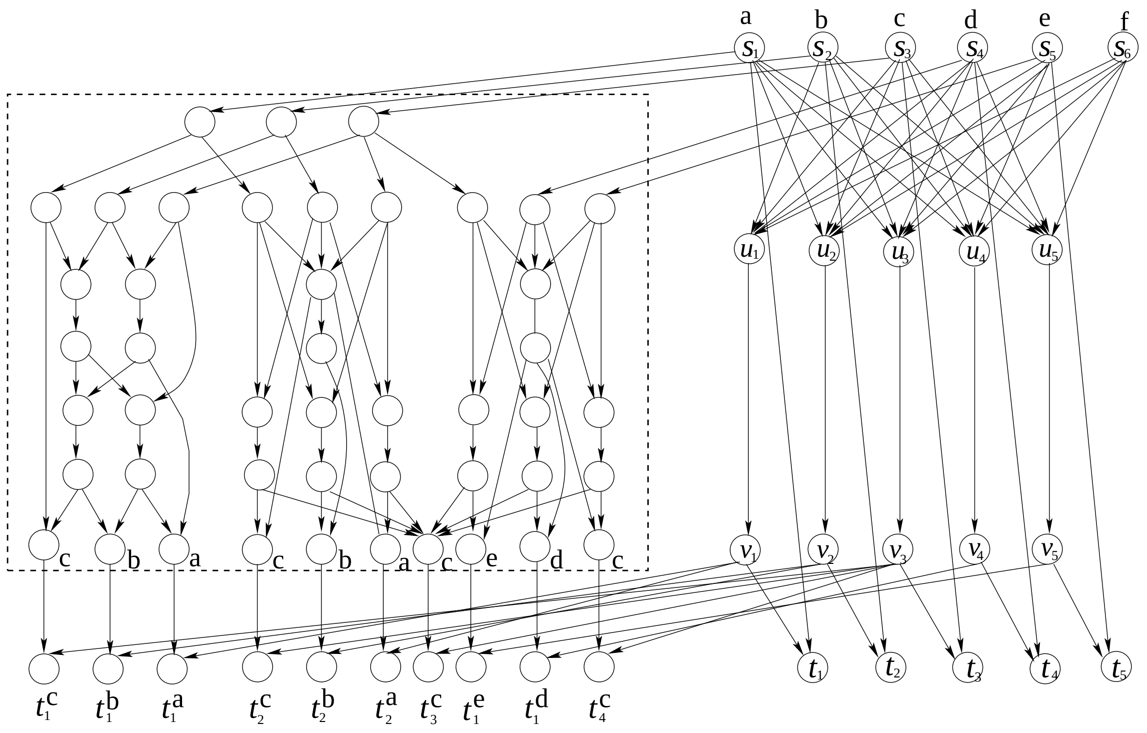

Figure 4.7: A sum-network for which linear network coding is insufficient.

4.3.2 Insufficiency of Fractional Linear Network Codes

Now we construct a sum-network where linear network code itself is insufficient. Specifically, we construct a sum-network which does not have a rate-1 linear network solution for any code length which it has a rate-1 non-linear network code solution. This sum-network is shown in Fig. 4.7. We construct this sum-network using a Type I network presented in [34] and Construction C_2. The Type I network is shown in the dashed box in Fig. 4.7. It was proved in [34] that for this Type I network, there is a rate-1 non-linear network code solution over the ternary alphabet even though there is no rate-1 linear network code solution over any finite field for any code length. By Theorem 13, this network has a rate-1 nonlinear network code solution over F_3, but it does not have any rate-1 linear network code solution over any finite field for any code length. So over F_3, rate-1 is achievable using non-linear network coding but not using linear network coding.

This gives us the following result.

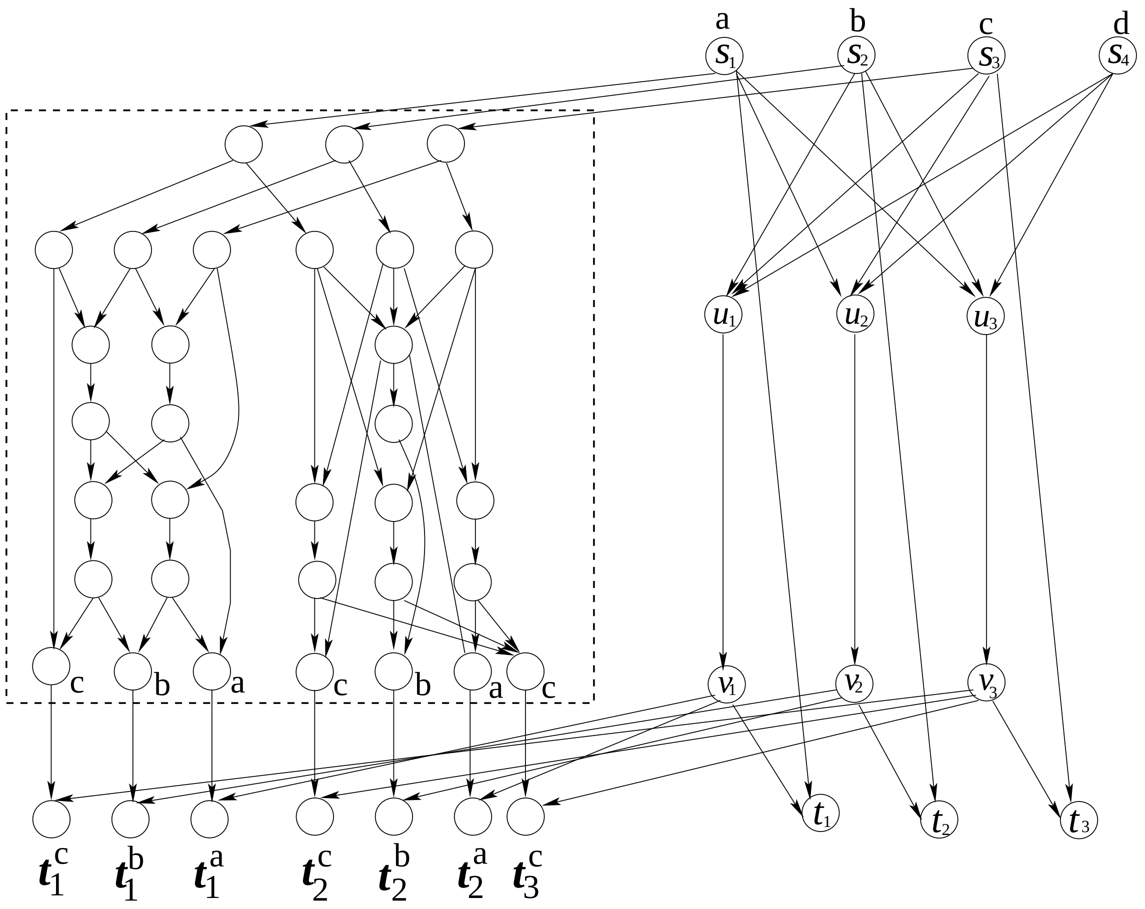

Figure 4.8: A sum-network whose network coding capacity is unachievable

Theorem 27. *There exists a solvable sum-network with the sum defined over a finite alphabet which is not linear solvable over any finite field.*

It was also shown in [34] that the network in the dashed box in Fig. 4.7 is not solvable using linear network coding even over R-module, or by using more general forms of the linear network codes defined in [35]. The same results hold for the sum-network in Fig. 4.7.

4.4 Unachievability of Coding Capacity

It is known that the network coding capacity of a Type I network is independent of the alphabet [24]. It was shown in [38] that there exists a directed acyclic network whose network coding capacity is not achievable over any finite alphabet. In this section, we show that there also exists a sum-network whose network coding capacity is not achievable over any finite module over a commutative ring with identity. Now consider the sum-network shown in Fig. 4.8. The network in the dashed box is taken from [38]. It was shown in [38] that the network coding capacity of this network in the dashed box is 1 and is not achievable. The sum-network in Fig. 4.8 is constructed from this network using the Construction C_2. By Corollary 18, the coding capacity of the sum-network over any field is 1. On the other hand, by Theorem 13 Part (ii), rate-1 is not achievable for the sum-network over any field. This gives the following theorem.

Theorem 28. *There exists a sum-network whose network coding capacity is not achievable.*

Even though the network coding capacity of a sum-network can be approached arbitrarily close by using sufficiently long message dimensions and network usase, Theorem 28 shows that for some sum-networks, the network coding capacity might not be exactly achievable by any network code.

4.5 New Results for Multiple-Unicast Networks

In Chapter 2, we have presented sum-networks $\mathcal{S}_m$ and $\mathcal{S}_m^*$. As we have mentioned earlier, these sum-networks are special in the sense that for these sum-networks, a rate-1 linear network code solution depends only on the characteristics of the field. By using Theorem 19, we have the following results for multiple-unicast networks.

Theorem 29. *For any finite, possibly empty, set $\mathcal{P} = \{p_1, p_2, \ldots, p_\lambda\}$ of positive prime numbers, there exists a multiple-unicast network so that for any $k \geq 1$, it has a k-length linear network code solution over a finite field if and only if the characteristic of the finite field belongs to $\mathcal{P}$.*

Theorem 30. *For any finite set $\mathcal{P} = \{p_1, p_2, \ldots, p_\lambda\}$ of positive prime numbers, there exists a multiple-unicast network so that for any $k \geq 1$, it has a k-length linear network code solution over a finite field if and only if the characteristic of the finite field does not belong to $\mathcal{P}$.*

To the best of our knowledge, these are new results for multiple-unicast networks. Every acyclic directed Type I network has a scalar linear solution only over fields whose characteristics belong to finite or co-finite set of primes. However, this result may not hold for k-length linear network code for code length $k > 1$. Theorems 29 and 30 show the existence of multiple-unicast networks whose rate-1 linear network code solutions depend only on the characteristics of fields and not on the code length.

4.6 Summary

In this chapter, first, we have shown that there exists a scalar linear solvable equivalent sum-networks for any set of integer polynomial equations. Then, we have shown that a sum-network is reversible under fractional linear network coding. Next, we have shown the existence of a non-reversible sum-network. Then, we have shown the insufficiency of scalar linear network coding as well as linear network coding in general for sum-networks. We have also shown that there exists a sum-network whose network coding capacity is unachievable. Finally, we have given some new results for multiple-unicast networks.

Chapter 5

Bounds on Network Coding Capacity

In this chapter, we consider the problem of determining the network coding capacity of sum-networks. It has been shown that network coding is crucial for multicast networks to achieve the upper bound on network coding capacity [16]. The maximum throughput in a network using only routing is termed as *routing capacity* in the literature. There exist multicast networks whose routing capacity is strictly less than the network coding capacity. However, the network coding capacity of Type I netwoks, in general, is not known [19, 31]. For sum-networks, the network coding capacity is upper bounded by the minimum of min-cut capacities of all source-terminal pairs (Theorem 15). We call this upper bound as *min-cut bound.*

First, we show that for a sum-network having either one source or one terminal, min-cut bound is achieved by linear network coding. Then, we give various lower bounds on the network coding capacity of sum-networks having more than one source and more than one terminal. In a special class of sum-network having either 3 sources or 3 terminal, we show that our given lower bound is tight. We present these bounds in different sections dealing with various numbers of sources and terminals. We also show that there exists sum-networks having more than two sources and more than two terminals where the min-cut bound is not achievable over any finite alphabet. In this chapter, m and n will denote the number of sources and the number of terminals respectively.

5.1 The case of $\min\{m, n\} = 1$

In Chapters 3 and 4, we have considered the reverse sum-network of a sum-network. We have shown that if a sum-network has a (k, n) fractional linear network code solution, then from such a network code, one can also construct a (k, n) fractional linear network code solution of the reverse sum-network. This means that the linear network coding capacity of the reverse sum-network is the same as the linear network coding capacity of the original sum-network (Lemma 16).

The min-cut bound on the network coding capacity may not be achievable in general. A special case arises when there is only one source or one terminal in the sum-network. If the sum-network has only one source, then the network is a multicast network. Note that a multicast network is a special case of sum-network with only one source, and its reverse network is a sum-network with only one terminal. The network coding capacity of a multicast network is known to be *equal* to the minimum of the min-cuts of the source-terminal pairs, and thus the network coding capacity achieves the min-cut bound [16]. Moreover, this network coding capacity is achieved by scalar linear network codes if alphabet is a finite field [26].

Now, over a finite field, for the case of $n = 1$, let us consider the reverse sum-network network of a sum-network obtained by reversing the direction of the edges and interchanging the role of the sources and the terminals. The reverse network is a multicast network and thus has linear network coding capacity equal to the minimum of the min-cut capacities of the source-terminal pairs. So the linear network coding capacity of the original one-terminal sum-network is the minimum of the min-cut capacities of the source-terminal pairs. Since the network coding capacity is also upper bounded by the min-cut bound, the coding capacity of a one-terminal sum-network is the minimum of the min-cuts of the source-terminal pairs. So we have

Theorem 31. *The network coding capacity (and the linear network coding capacity) of a one-source or one-terminal sum-network is the minimum of the min-cuts of all source-terminal pairs.*

There exists a polynomial time algorithm to design a scalar linear network code to achieve the linear network coding capacity for multicast networks [28]. By using this polynomial time algorithm, we can design a network coding capacity achieving scalar linear network code for the reverse sum-network (which is a multicast network) of a sum-network having only one terminal. Now, by the canonical reverse code construction presented in Section 4.2.1, a network coding capacity achieving scalar linear network can be designed for the sum-network.

We mention that the result in Theorem 31 has also been proved in Theorem 5 in [39] using the concept of dual code. However, our treatment is more elementary.

5.2 The case of $\min\{m, n\} = 2$

It was proved in [56] that for a network with $\min\{m, n\} = 2$ where every source-terminal pair is connected, it is possible to communicate the sum of symbols generated at all sources to every terminal terminal by using a scalar linear network code over every finite field. Which means that for $\min\{m, n\} = 2$, network coding capacity ≥ 1 if the min-cut bound is at least 1. So, the min-cut bound is tight in this case if the min-cut bound is 1. However, if the min-cut bound is greater than 1, then it is not known if this min-cut bound is achievable. However, we can always achieve the half of the min-cut bound by time-sharing. For example, for $m = 2$, each source can communicate its symbols to all the terminals in one time slot at the rate of min-cut bound and then after two time-slots, the terminals can add the symbols received from the two sources. Similarly, for $n = 2$, the sum of the symbols generated at all the sources can be communicated to each terminal in one time slot at the rate of min-cut bound. So, we have

Theorem 32. *For $\min\{m, n\} = 2$, the network coding capacity of a sum-network is bounded as*

$$\begin{aligned} \textit{network coding capacity} \;\geq\; & \max\{\min\{1, min_{i,j}(\text{min-cut } (s_i - t_j))\}, \\ & \qquad 0.5 \times min_{i,j}(\text{min-cut } (s_i - t_j))\}\,. \end{aligned}$$

If the min-cut bound is 1, the lower bound in Theorem 32 is tight and follows from [56]. Our contribution in Theorem 32 is essentially for the case of min-cut bound greater than 1.

5.3 The case of $m = n = 3$

The case of $m = n = 3$ is intriguing. On one hand, these are the smallest values of m, n for which there is a sum-network (called $\mathcal{S}_3$ in Chapter 2 and shown again in Fig. 5.1) where every source-terminal pair is connected, but still does not have a rate-1 linear network code solution for any code length (Chapter 2) or a non-linear rate-1 network code solution [57]. So, these are the smallest parameters for which the min-cut bound is known to be not achievable. (Though it is still not clear at this point if the min-cut bound may still be achievable in the limit as the suppremum of achievable rates.) On the other hand, from elaborate investigation of possible networks with these parameters, there seems to be very limited types of networks. The S_3 and its extensions (essentially the sum-network $\mathcal{S}_3'$ in Chapter 2 and shown again in Fig. 5.2) seem to be the only "non-solvable" sum-networks for $m = n = 3$. The sum-network $\mathcal{X}_3$ shown in Fig. 5.3 has been presented in Chapter 2 and has been shown to be solvable by scalar linear code over all fields except the binary field $\mathbb{F}_2$.

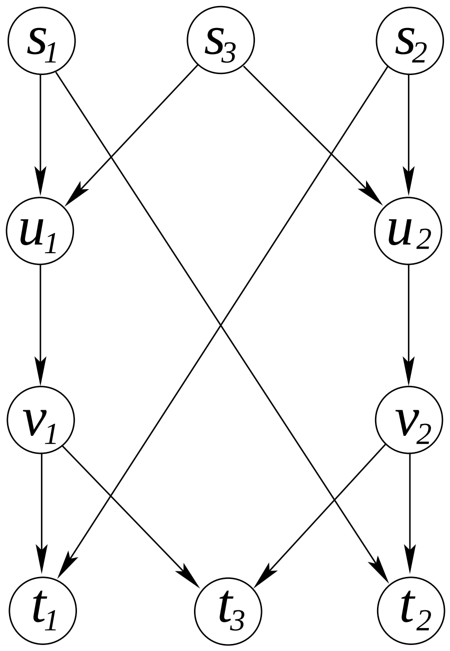

Figure 5.1: The network $\mathcal{S}_3$

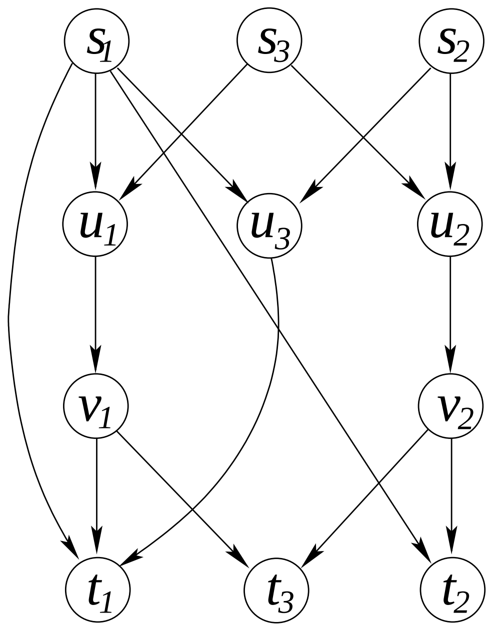

Figure 5.2: The network $\mathcal{S}_3'$

First, we give a generic lower bound on the network coding capacity of any sum-network with $\min\{m, n\} = 3$ and with min-cut bound ≥ 1.

Theorem 33. *The linear coding capacity of any sum-network with* $\min\{m, n\} = 3$ *with min-cut bound* ≥ 1 *is at least* $2/3$.

Proof. W.l.o.g., let us assume that the number of terminals is 3 (otherwise consider the reverse network). Let us consider two symbols at each source: X_{i1}, X_{i2} at s_i for $i = 1, 2, \ldots, m$. Let the two sums be denoted as $Sum_1 = \sum_{i=1}^{m} X_{i1}$ and $Sum_2 = \sum_{i=1}^{m} X_{i2}$. If we take two terminals at a time, the resulting network has a network coding capacity ≥ 1 as discussed in Section 5.2 using scalar linear network coding as proposed in [56]. Now, the two sums Sum_1 and Sum_2 can be communicated to all the terminals in three time slots. In the first time slot, Sum_1 is communicated to t_1 and t_2. In the second time slot, Sum_2 is communicated to t_2 and t_3. In the third time slot, $Sum_1 + Sum_2 = \sum_{i=1}^{m}(X_{i1} + X_{i2})$ is communicated to t_1 and t_3. Having received Sum_1 (respectively Sum_2) and $Sum_1 + Sum_2$, the terminal t_1 (respectively t_3) can recover Sum_2 (respectively Sum_1) as well. So all the terminals recover the two sums in three time slots, thus achieving a rate $2/3$ using linear network coding. ■

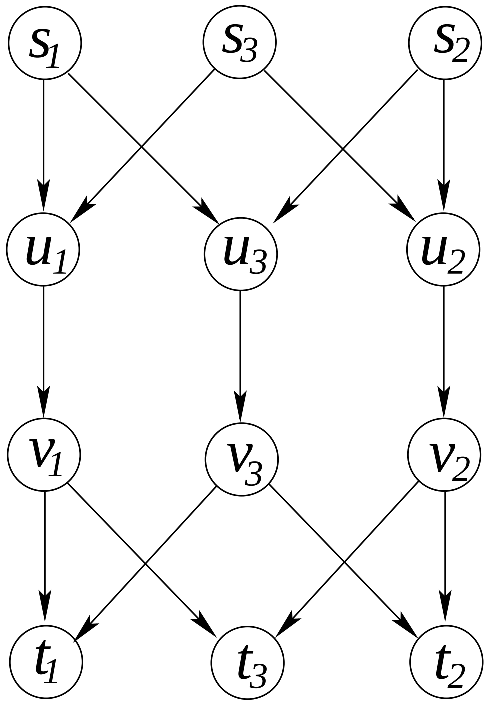

Figure 5.3: The network $\mathcal{X}_3$

It was proved in [57] that if a sum-network with $m = n = 3$ has two edge-disjoint paths between any source-terminal pairs, then the network is linearly solvable, that is, rate-1 is achievable by scalar linear coding. This gives the following bound.

Proposition 34. *The linear network coding capacity of any sum-network with $m = n = 3$ with min-cut ≥ 2 is at least 1.*

Now we show that the sum-network $\mathcal{S}_3$ and its extension $\mathcal{S}_3'$ both have network coding capacity exactly 2/3 whereas their min-cut bound is 1. So, there is a gap between the network coding capacity capacity and the min-cut upper bound.

Theorem 35. *The network coding capacity and linear network coding capacity of $\mathcal{S}_3$ and $\mathcal{S}_3'$ is $2/3$.*

Proof. Clearly, the sum-network $\mathcal{S}_3'$ is obtained from $\mathcal{S}_3$ by adding one direct edge from s_1 to t_1, and subdividing the edge (s_2, t_1) and adding one edge into it from s_1. So, the network coding capacity and the linear network coding capacity of the sum-network $\mathcal{S}_3'$ is at least that of $\mathcal{S}_3$. By the previous theorem, the rate 2/3 is achievable by linear network coding in $\mathcal{S}_3'$. Now we will show that the network capacity of $\mathcal{S}_3'$ is bounded from above by 2/3. This will prove that both the sum-networks have the same network coding capacity and linear network coding capacity and that these are both 2/3.

Consider any (k, l) fractional network code solution of $\mathcal{S}_3'$. Let $X_1, X_2, X_3 \in \mathbb{F}^k$ be the message blocks generated at the three sources, where $\mathbb{F}$ is a finite field. Let the edges (u_1, v_1) and (u_2, v_2) carry the functions $\phi(X_1, X_3)$and $\psi(X_2, X_3)$ respectively. For any fixed values of X_1 and X_2, the set of messages received by the terminal t_1 should be a one-one function of X_3 since the terminal can recover the sum $X_1 + X_2 + X_3$ which is a one-one function of X_3. Since the messages on (s_1, t_1) and (u_3, t_1) are fixed by the values of X_1 and X_2, the message on (v_1, t_1) and thus $\phi(X_1, X_3)$ must be a one-one function of X_3 for a fixed value of X_1.

Now clearly, for t_2 to be able to recover the sum, the function ψ should be such that one can recover X_2+X_3 from $\psi(X_2, X_3)$. Since t_3 recovers the sum $X_1+X_2+X_3$, and it can recover $X_2 + X_3$ from the message on (v_2, t_3), it can also recover X_1 by subtracting. Now, t_3 receives $\phi(X_1, X_3)$ on (v_1, t_3) (W.l.o.g.) and this is a one-one function of X_3 for any given X_1. So, having recovered X_1, t_3 can recover X_3 from $\phi(X_1, X_3)$. Then by using $\psi(X_2, X_3)$ received on (v_2, t_3) and the value of X_3, t_3 can also recover X_2. So, t_3 can recover all the original messages X_1, X_2, X_3. Now (X_1, X_2, X_3) takes a total of $|F|^{3k}$ possible values as a triple. On the other hand $\{(u_1, v_1), (u_2, v_2)\}$ is a cut between the sources and t_3, and this cut can carry at most $|F|^{2l}$ possible different message-pairs. So, we have $|F|^{2l} \geq |F|^{3k} \Rightarrow k/l \leq 2/3$. ■

The following observations lead us to believe that the sum-network $\mathcal{S}_3'$ is essentially the only maximal extension of $\mathcal{S}_3$ which has the same network coding capacity.

1. Further subdividing (u_3, t_1) and adding an edge from it to t_2 makes the sum-network $\mathcal{X}_3$ a subgraph of the resulting network, and thus the network coding capacity of the sum-network increases to 1.

2. Also subdividing (s_1, t_2) and adding an edge from it into t_1 does not change its network coding capacity since there is already an edge (s_1, t_1) and the new edge can not carry any extra information to t_1.

3. Also subdividing (s_1, t_2) and adding an edge to it from s_2 gives a strictly richer network than $\mathcal{X}_3$, and thus the network coding capacity of the network increases

to 1.

4. Instead of the edge (s_1, t_1), if an edge (s_2, t_2) is added, then the resulting network (shown in Fig. 5.4) is strictly richer than $\mathcal{X}_3$ because the edges (s_1, t_2) and (s_2, t_2) can jointly carry more information than an edge from u_3 to t_2. So the resulting network has network coding capacity 1 even though it does not have a binary scalar solution (like $\mathcal{X}_3$).

These observations also lead us to believe that

Conjecture 36. *The network coding capacity of a sum-network with* $m = n = 3$ *is either* $0, 2/3$ *or at least* 1.

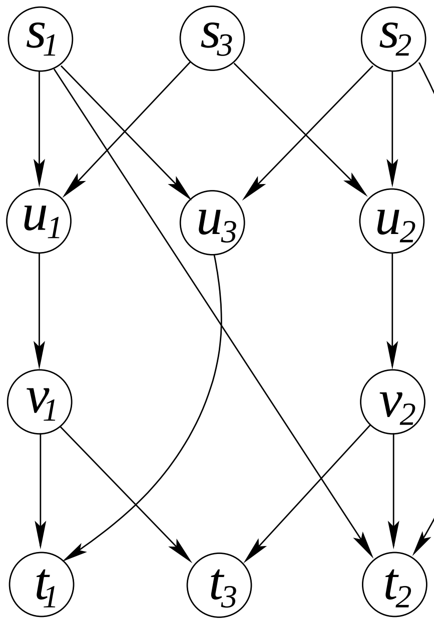

Figure 5.4: The network $\mathcal{X}_3'$

5.4 The case of $m, n > 3$

This is the most ill-understood class of sum-networks. We only have what we suspect to be a very loose lower bound on the capacity of this class of networks. This lower bound is obtained by similar coding by time-sharing scheme as in the proof of Theorem 33.

Theorem 37. *The linear coding capacity of a sum-network with* $\min\{m,n\} \geq 2$ *and min-cut* ≥ 1 *is at least* $2/\min\{m,n\}$*.*

Proof. The case of $\min\{m,n\} = 2$ follows from Theorem 32. For $\min\{m,n\} > 2$, without loss of generality, let us assume that $n \leq m$. For even n, we can group the terminals into $n/2$ pairs and in each time slot communicate the sum of the source symbols to one pair of terminals. So, in $n/2$ time slots we can communicate one sum of the source symbols to all the terminals thus achieving a rate $2/n$. For odd n, we can group the terminals into $(n-3)/2$ pairs and one triple. We can communicate one sum to each pair in one time slot. So, we can communicate two sums to all the pairs in $(n-3)$ slots. Then using the same scheme as in the proof of Theorem 33, we can communicate two sum to the group of three terminals in three time slots. So, in overall n time slots, we can communicate two sums to all the terminals in the network. This gives us a rate $2/n = 2/\min\{m,n\}$. ■

If the min-cut bound is 1 and $\min\{m,n\} = 2$, the lower bound in Theorem 37 is tight and follows from [56]. Our contribution in Theorem 37 is for the case of $\min\{m,n\} > 2$.

Remark 5. *For the case* $m = 3, n \geq 3$ *or* $m \geq 3, n = 3$*, the lower bound by Theorem 37 is* $2/3$*. We claim that this bound is tight. We prove this claim by using the construction* $\mathcal{S}_3 \bowtie K_{m-3,n-3}$*. Now, for the sum-network* $\mathcal{S}_3 \bowtie K_{m-3,n-3}$*, by taking* $m = 3, n \geq 3$ *or* $m \geq 3, n = 3$*, our claim follows from Theorem 35.*

Remark 6. *The results by Lemma 33, Theorem 35, and Theorem 37 also follow for alphabets having more general algebraic structures such as a module over a ring.*

We believe that the bound in Theorem 37 is very loose and there is scope for improvement. Even though we failed to come up with any achievable scheme of higher rate, we also failed to construct a network satisfying the bound with equality.

We suspect a decreasing degree of tightness in these bounds as the numbers of sources and terminals increase. We summarize the bounds in Table 5.1 to bring out this observation. We have given lower bounds for the min-cut bound > 1 by using time shared multicasting and the result that a sum-network is reversible under fractional linear network coding. The parenthetic comments in the table entries indicate the tightness of

	Solvability		Network coding capacity	
	min-cut bound $= 1$	min-cut bound > 1	min-cut bound $= 1$	min-cut bound > 1
$\min\{m,n\} = 1$	Solvable	Solvable	$= 1$	$=$ min-cut bound
$\min\{m,n\} = 2$	Solvable	Solvable	$= 1$	$\geq$ min-cut bound/2 (loose/tight?)
$m = n = 3$	Network dependent	Solvable	$\geq 2/3$ (tight)	$\geq \max\{1, \text{min-cut bound}/3\}$ (loose!)
$\min\{m,n\} = 3$	Network dependent	?	$\geq 2/3$ (tight)	$\geq$ min-cut/3 (loose!)
$\min\{m,n\} > 3$	Network dependent	?	$\geq 2/\min\{m,n\}$ (loose!)	$\geq$ min-cut bound$/\min\{m,n\}$ (loose!)

Table 5.1: Solvability and bounds on the capacity

the bound as known or conjectured (indicated with an exclamation (!) mark.) The interrogation (?) mark as an entry indicates that nothing is known about the case.

5.5 Summary

In this chapter, we have made an attempt to explore the network coding capacity of the sum-networks. We have shown that the network coding capacity of a sum network with either one source or one terminal is equal to min-cut bound and is achievable using linear network coding. Then we have presented lower bounds for sum-networks depending on number of sources, number of terminals, and min-cut bound. We have shown that for the case $m = 3, n \geq 3$ or $m \geq 3, n = 3$, the lower bound is tight.

5.6 Open Questions

For the class of multiple-unicast network, for any rational number, one can easily construct a multiple-unicast network which has network coding capacity equal to that rational number. One can also easily construct a multiple-unicast network which has linear network coding capacity equal to that rational number. It is not clear if this is possible for the class of sum-networks. Based on this observation, we pose the following questions. In the following, $\mathbb{Q}_+$ represents the set of positive rational numbers.

- Does there exist a sum-network which has a chosen network coding capacity C for every choice of $C \in \mathbb{Q}_+$?

- Does there exist a sum-network which has a chosen linear network network coding capacity C for every choice of $C \in \mathbb{Q}_+$?

The above questions can be generalized for real numbers. In the following, $\mathbb{R}_+$ represents the set of positive real numbers.

- Does there exist a sum-network which has a chosen network coding capacity C for every choice of $C \in \mathbb{R}_+$?
- Does there exist a sum-network which has a chosen linear network coding capacity C for every choice of $C \in \mathbb{R}_+$?

It would be interesting to prove or disprove the tightness of lower bounds given in this chapter, namely,

- Is the lower bound given in Theorem 32 tight?
- Is the lower bound given in Theorem 37 tight?

Chapter 6

Summary and Directions for Future Work

In this thesis, we have considered a communication problem over a directed acyclic network where every terminal wants to recover the sum of symbols generated at all the sources. We have assumed that each source generates one i.i.d. random process with uniform distribution over a finite alphabet, having abelian group structure, and different source processes are independent. We have also assumed that all the links in the network are unit capacity links which are error-free and delay free. We have referred such a directed acyclic network as a *sum-network*.

First, we have considered rate-1 linear network code solutions over finite fields. We have constructed two special classes of directed acyclic networks. In the first constructed class, for any finite set of primes, the problem of communicating the sum of symbols generated at all sources to all the terminals has rate-1 linear network code solutions over a finite field if and only if the characteristic of the finite field belong to the given set. In the second constructed class, for any finite set of primes, the problem of communicating the sum of symbols generated at all sources to all the terminals has rate-1 linear network code solutions over a finite field if and only if the characteristic of the finite field does not belong to the given set. In these two classes, rate-1 linear network code solutions depends only on the characteristics of the finite fields. Then we have shown that there exists a directed acyclic network where a scalar linear network code solution for communicating the sum of symbols not only depend on the characteristic of the finite field but also on the size of the finite field. Then we have shown that source-terminal pairs connectivity is insufficient for the solvability over any finite alphabet for the problem of communicating the sum of symbols over a directed acyclic network with more than two sources and more than two terminals.

Then we have constructed some equivalent networks. We have shown that the problem of communicating the sum of the symbols over a acyclic directed network and the problem of communicating a linear function of the symbols over the same acyclic directed network are solvably equivalent under (k, l) fractional network coding if the component linear functions are invertible. Then we have constructed a solvably equivalent (also linear solvably equivalent) sum-network to any given multiple-unicast network. We have also shown that the reverse sum-network of the constructed sum-network is solvably equivalent (also linear solvably equivalent) to the corresponding reverse multiple-unicast network. Next we have constructed a solvably equivalent (also linear solvably equivalent) sum-network to any given directed acyclic Type I network. Finally, we have constructed a linear solvably equivalent multiple-unicast network to any given sum-network.

Using equivalent network constructions, we have shown that for any set of polynomials having integer coefficients, there exists a sum-network which is scalar linear solvable over a finite field $\mathbb{F}$ if and only if the polynomials have a common root in $\mathbb{F}$. We have also shown that there exists a solvable sum-network whose reverse network is not solvable. However, we have shown that a sum-network and its reverse sum-network are solvably equivalent under fractional linear network coding. Then we have shown that linear network codes are insufficient to achieve a rate which is achievable by using non-linear network codes by constructing an example sum-network. We have also shown unachievability of the network coding capacity for sum-networks.

Finally, we have investigated on the network coding capacity of sum-networks. We have presented some bounds in terms of min-cut bound, and the numbers of sources and terminals.

6.1 Directions for Future Work

We have mentioned specific open questions at the end of Chapters 2, 3 and 5 based on the work in those chapters. Here, we mention some broad directions for future work.

1. In this thesis, we have assumed that different source processes are independent. It would be intersting to consider the problem of communicating the sum of symbols when different source processes are correlated.

2. In this thesis, we have considered sum-networks which are directed acyclic networks. It would be interesting to consider sum-networks which are directed networks with cycle. It would also be intersting to consider sum-networks which are undirected networks.

3. It would be interesting to consider the problem of communicating the modulo-sum of symbols. Consider a sum-network having m sources and n terminals. In this problem setup, for $1 \leq i \leq m$, source s_i generates a symbol X_i from the alphabet $\{$0,1,. . . ,p-1$\}$, where $p \in \mathbb{Z}$; and each terminal requires $(X_1+X_2+\ldots+X_m) \bmod p$.

4. In this thesis, all the link are assumed to be wireline and error-free links. It would be intersting to consider erroneous wireless links.

Bibliography

[1] C. E. Shannon. A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423, 623–656, 1948.

[2] E. G. Jaynes. Information theory and statistical mechanics. *Phys. Rev.*, 106(4):620–630, 1957.

[3] A. Katz. *Principles of Statistical Mechanics: The Information Theory Approach.* Freeman, San Francisco, 1967.

[4] P. Algoet and T. M. Cover. Asymptotic optimality and asymptotic equipartition property of log-optimal investment. *Ann. Prob.*, 16(2):876–898, 1988.

[5] T. M. Cover. Universal portfolios. *Math. Finance*, 1–29, 1991.

[6] C. Adami Information Theory in Molecular Biology. *Physics of Life Reviews*, 1:3–22, 2004.

[7] J. Avery. *Information Theory and Evolution.* World Scientific Publishing, 2003.

[8] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon limit error-correcting coding and decoding: Turbo codes. In *IEEE International Conference on Communications*, pages 1064–1070, Geneva, Switzerland, 1993.

[9] D. J. C. MacKay and R. M. Neal. Near Shannon limit performance of low density parity check codes. *Electron. Lett.*, 32(18):1645–1646, 1996.

[10] T. Cover and J. Thomas. *Elements of Information Theory.* John Wiley & Sons, Inc., New York, 1991.

[11] S. Verdú. Fifty Years of Shannon Theory. *IEEE Trans. Inform. Theory*, 44(6):2057–2078, 1998.

[12] A. Ephremides and B. Hajek. Information theory and communication networks: an unconsummated union. *IEEE Trans. Inform. Theory*, 44(6):2416–2434, 1998.

[13] L. R. Ford and D. R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8:399–404, 1956.

[14] P. Elias, A. Feinstein, and C. Shannon. A note on the maximum flow through a network. *IRE Trans. Inform. Theory*, 2(4):117–119, 1956.

[15] K. Jain, M. Mahdian, and M. R. Salavatipour. Steiner trees. In *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms (SODA)*, 266–274, 2003

[16] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung. Network information flow. *IEEE Trans. Inform. Theory*, 46(4):1204–1216, 2000.

[17] N. Cai, M. Chiang, M. Effros, R. Koetter, M. Medard, B. Prabhakar, R. Srikant, D. Towsley, and R. W. Yeung, Introduction to the Special Issue on Networking and Information Theory. *IEEE Trans. Inform. Theory*, 52(6):2285–2288, 2006.

[18] T. Ho. *Networking from a network coding perspective.* Ph.D. Thesis, Massachusetts Institute of Technology, 2004.

[19] A. R. Lehman. *Network Coding.* Ph.D. thesis, Massachusetts Institute of Technology, 2005.

[20] A. Ramamoorthy. *Generalised ACE codes and information theoretic results in network.* Ph.D. Thesis, University of California, Los Angeles, 2005.

[21] S. Jaggi. *Design and Analysis of Network Codes.* Ph.D. Thesis, California Institute of Technology, Pasadena, California, 2006.

[22] Y. Wu. *Network Coding for Multicasting.* Ph.D. Thesis, Princeton University, 2006.

[23] J. L. Cannons. *Topics in Network Communications.* Ph.D. Thesis, University of California, San Diego, 2008.

[24] J. Cannons, R. Dougherty, C. Freiling, and K. Zeger. Network routing capacity. *IEEE Trans. Inform. Theory*, 52(3):777–788, 2006.

[25] N. Karamchandani, R. Appuswamy, M. Franceschetti and K. Zeger. Network computing capacity for the reverse butterfly network. In *Proceedings of IEEE International Symposium on Information Theory*, Seoul, Korea, 2009.

[26] S.-Y. R. Li, R. W. Yeung, and N. Cai. Linear network coding. *IEEE Trans. Inform. Theory*, 49(2):371–381, 2003.

[27] R. Koetter and M. Médard. An algebraic approach to network coding. *IEEE/ACM Trans. Networking*, 11(5):782–795, 2003.

[28] 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*, 51(6):1973–1982, 2005.

[29] T. Ho, R. Koetter, M. Médard, M. Effros, J. Shi, and D. Karger. A random linear network coding approach to multicast. *IEEE Trans. Inform. Theory*, 52(10):4413–4430, 2006.

[30] A. R. Lehman and E. Lehman. Complexity classification of network information flow problems. In *Proceedings of 41st Annual Allerton Conference on Communication, Control, and Computing,* Monticello, IL, October 2003.

[31] M. Langberg and A. Sprintson. On the hardness of approximating the network coding capacity. In *Proceedings of IEEE International Symposium on Information Theory*, Toronto, Canada, 2008.

[32] M. Médard, M. Effros, T. Ho, and D. Karger. On coding for nonmulticast networks. In *Proceedings of 41st Annual Allerton Conference on Communication, Control, and Computing,* Monticello, IL, October 2003.

[33] S. Riis. Linear versus nonlinear boolean functions in network flow. In *Proceedings of 38th Annual Conference on Information Sciences and Systems,* Princeton, NJ, March 2004.

[34] R. Dougherty, C. Freiling, and K. Zeger. Insufficiency of linear coding in network information flow. *IEEE Trans. Inform. Theory*, 51(8):2745–2759, 2005.

[35] S. Jaggi, M. Effros, T. Ho, and M. Médard. On linear network coding. In *Proceedings of 42nd Annual Allerton Conference on Communication, Control, and Computing,* Monticello, IL, September 2004.

[36] R. Dougherty, C. Freiling, and K. Zeger. Linear network codes and systems of polynomial equations. *IEEE Trans. Inform. Theory*, 54(5):2303–2316, 2008.

[37] R. Baines and P. Vámos. An algorithm to compute the set of characteristics of a system of polynomial equations over the integers. *J. Symb. Comput.*, 35:269–279, 2003.

[38] R. Dougherty, C. Freiling, and K. Zeger. Unachievability of network coding capacity. *IEEE Trans. Inform. Theory*, 52(6):2365–2372, 2006.

[39] R. Koetter, M. Effros, T. Ho, and M. Médard. Network codes as codes on graphs. In *Proceedings of 38th Annual Conference on Information Sciences and Systems*, Princeton, NJ, March 2004.

[40] S. Riis. Reversible and irreversible information networks. In *IEEE Trans. Inform. Theory*, 53(11):4339–4349, 2007.

[41] R. Dougherty and K. Zeger. Nonreversibility and equivalent constructions of multiple-unicast networks. *IEEE Trans. Inform. Theory*, 52(11):5067–5077, 2006.

[42] T. Ho and D. S. Lun *Network Coding: An Introduction.* Cambridge University Press 2008.

[43] R. W. Yeung. *Information Theory and Network Coding.* Springer 2008.

[44] C. Fragouli and E. Soljanin *Network coding fundamentals.* Foundations and Trends® in Networking, 2(1):1–133, January 2007.

[45] C. Fragouli and E. Soljanin *Network coding applications.* Foundations and Trends® in Networking, 2(2):135–269, January 2007.

[46] J. Korner and K. Marton. How to encode the modulo-two sum of binary sources. *IEEE Trans. Inform. Theory*, 25(2):219–221, 1979.

[47] T. S. Han and K. Kobayashi. A dichotomy of functions $f(x, y)$ of correlated sources (x, y). *IEEE Trans. Inform. Theory*, 33(1):69–86, 1987.

[48] A. Orlitsky and J. R. Roche. Coding for computing. *IEEE Trans. Inform. Theory*, 47(3):903–917, 2001.

[49] H. Feng, M. Effros, and S. A. Savari. Functional source coding for networks with receiver side information. In *Proceedings of 42nd Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, September 2004.

[50] D. Krithivasan and S. S. Pradhan. An achievable rate region for distributed source coding with reconstruction of an arbitrary function of the sources. In *Proceedings of IEEE International Symposium on Information Theory*, pages 56–60, Toronto, Canada, 2008.

[51] J. N. Tsistsiklis. Decentralized detection by a large number of sensors. *Mathematics of Control, Signals and Systems*, 1(2):167–182, 1988.

[52] R. G. Gallager. Finding parity in a simple broadcast network. *IEEE Trans. Inform. Theory*, 34(2):176–180, 1988.

[53] A. Giridhar and P. R. Kumar. Computing and communicating functions over sensor networks. *IEEE J. Select. Areas Commun.*, 23(4):755–764, 2005.

[54] Y. Kanoria and D. Manjunath. On distributed computation in noisy random planar networks. In *Proceedings of IEEE International Symposium on Information Theory*, Nice, France, 2008.

[55] S. Boyd, A. Ghosh, B. Prabhaar, and D. Shah. Gossip algorithms: design, analysis and applications. In *Proceedings of IEEE INFOCOM*, pages 1653–1664, 2005.

[56] A. Ramamoorthy. Communicating the sum of sources over a network. In *Proceedings of IEEE International Symposium on Information Theory*, pages 1646–1650, Toronto, Canada, July 06-11, 2008.

[57] M. Langberg and A. Ramamoorthy. Communicating the sum of sources in a 3-sources/3-terminals network. In *Proceedings of IEEE International Symposium on Information Theory*, Seoul, Korea, 2009.

[58] N. Karamchandani, R. Appuswamy, M. Franceschetti and K. Zeger. Network coding for computing. In *Proceedings of 46th Annual Allerton Conference on Communication, Control, and Ccomputing,* Monticello, IL, September 2008.

[59] A. Ramamoorthy and M. Langberg. 2008. personal communication.

[60] B. K. Rai, B. K. Dey, and A. Karandikar. Some results on communicating the sum of sources over a network. In *Proceedings of NetCod 2009*, EPFL, Lausanne, Switzerland, 2009.

[61] B. K. Rai and B. K. Dey. Feasible alphabets for communicating the sum of sources over a network. In *Proceedings of IEEE International Symposium on Information Theory*, Seoul, Korea, 2009.

[62] B. K. Rai and B. K. Dey. Sum-networks: system of polynomial equations, reversibility, insufficiency of linear network coding, unachievability of coding capacity. *Submitted to IEEE Trans. Info. Theory*, August 2009.

[63] B. K. Rai, B. K. Dey, and S. Shenvi. Some bounds on the capacity of communicating the sum of sources. In *Proceedings of IEEE International Theory Workshop*, Cairo, Egypt, January 2010.

Publications from the Thesis

1. Brijesh Kumar Rai, Bikash Kumar Dey, and Sagar Shenvi. Some bounds on the capacity of communicating the sum of sources. In *Proceedings of IEEE International Theory Workshop*, Cairo, Egypt, January 2010.

2. Brijesh Kumar Rai and Bikash Kumar Dey. Sum-networks: system of polynomial equations, reversibility, insufficiency of linear network coding, unachievability of coding capacity. *Submitted to IEEE Trans. Info. Theory*, August 2009.

3. Brijesh Kumar Rai and Bikash Kumar Dey. Feasible alphabets for communicating the sum of sources over a network. In *Proceedings of IEEE International Symposium on Information Theory*, Seoul, Korea, 2009.

4. Brijesh Kumar Rai, Bikash Kumar Dey, and Abhay Karandikar. Some results on communicating the sum of sources over a network. In *Proceedings of NetCod 2009*, EPFL, Lausanne, Switzerland, 2009.