RELATIONS AMONG NETWORK CODING, INDEX CODING, AND GUESSING GAMES: THE LINEAR CASE

JUAN CAMILO TORRES CHAVES

NATIONAL UNIVERSITY OF COLOMBIA
FACULTY OF SCIENCE
DEPARTMENT OF MATHEMATICS
BOGOTÁ
2012

RELATIONS AMONG NETWORK CODING, INDEX CODING, AND GUESSING GAMES: THE LINEAR CASE

JUAN CAMILO TORRES CHAVES

FINAL PAPER SUBMITTED IN
PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE
OF

MASTER OF SCIENCE IN MATHEMATICS

DIRECTOR:
HUMBERTO SARRIA ZAPATA

NATIONAL UNIVERSITY OF COLOMBIA
FACULTY OF SCIENCE
DEPARTMENT OF MATHEMATICS
BOGOTÁ
2012

Title in English: Relations among Network Coding, Index Coding, and Guessing Games: The Linear Case.

Abstract: This paper is a monograph on some relations among Network Coding, Index Coding, and Guessing Games. We have focused on the linear case. We present in detail the most important results. The purpose is that the reader appreciates the profound connexions among these seemingly different areas of Network Theory.

Key words: Network Coding, Index Coding, Guessing Games, networks, digraphs.

Título en Español: Relaciones entre Codificación de Redes, Codificación de Índices y Juegos de Adivinción: El Caso Lineal.

Resumen: Este trabajo es una monografía sobre algunas relaciones entre Codificación de Redes, Codificación de Índices y Juegos de Adivinación. Nos hemos enfocado en el caso lineal. Presentamos en detalle los resultados más importantes. El propósito es que el lector aprecie las conexiones profundas entre estas áreas aparentemente diferentes de la Teoría de Redes.

Palabras Claves: Codificación de Redes, Codificación de Índices, Juegos de Adivinación, redes, digrafos.

Contents

Introduction

We live in an interconnected world, where communication plays a central role. It is not strange, then, that mathematicians and engineers have spent so much time studying how information can be transmitted on a network. Networks are represented by digraphs, which are one of the most well studied objects in Mathematics. Thus, the study of some information/communication problems can be reduced to the study of digraphs. Among these problems, the emerging areas of Network Coding, Index Coding, and Guessing Games are of special interest - the first two for their potential applications in communication, the last one for its recreational appeal.

We have a network with a set of sources and a set of receivers where each receiver requires some of the messages generated by the sources; roughly speaking, Network Coding deals on how information should be codified using some algebraic structure, like a finite field, and how data should be operated on the edges of the network in order to satisfy the receivers' demands. On the other side, Index coding try to answer what is the minimum number of broadcast transmissions necessary to satisfy some clients' demands, where these clients already have some of the information required by the other clients.

Network Coding was introduced in [1], and Index Coding in [3]; since then, many papers has been written in these areas - some about their theoretical aspects, some about their potential applications and some about how they can be implemented. The theoretical aspects are of special interest for mathematicians, since areas like Graph Theory, Linear Algebra, Matroid Theory an Information Theory have been used to build the foundations.

Finally, we have the Guessing Games, which were introduced in [5]. The game consist of a group of n player; we assign to each of them a number from $\{1, \ldots, s\}$. Each player cannot see his or her own number but is able to see the numbers of some of the other players. The group of players wins if all of them guess right their own numbers; they all lose if any of them guesses incorrectly. The idea is to try to find a strategy that maximize the probability of winning.

Although seemingly different at first sight, these areas -Network Coding, Index Coding, and Guessing Games- are strongly connected. The present paper is a monograph in some of the relations among Network Coding, Index Coding, and Guessing Games, but we have focused on the linear case.

The reasons to write this monograph are the following ones. Usually papers on these areas are written taken the engineers as audience, making them difficult to read for mathematicians. We want to write a monograph with the usual mathematical rigor for mathematicians with no prior knowledge in the study of networks. Also, some of the results presented in this monograph come from different papers, putting them in a single place gives us a better appreciation of them and shows us how they are connected.

The prerequisites are a basic knowledge in linear algebra (operations with matrices, vector spaces, linear transformations, the Rank-Nullity Theorem) and abstract algebra (integers modulo n and finite fields), and the ones in Chapter 1.

In Chapter 1 we present the necessary prerequisites to read the next chapters. We present some basic concepts and results in finite fields, linear algebra, and graph theory. We also introduce some of the notation used throughout the work.

Chapter 2 is devoted to Network Coding. We define the concepts of network, multiple-unicast network, network code, and solution. We explain the process of reducing a general network into a multiple-unicast network presented in [4].

Chapter 3 is devoted to Index Coding. We define the concepts of index code and linear index code. We present an important result concerning linear index codes from [2]. This result plays a central role in connection with guessing games.

The last chapter, Chapter 4, is about Guessing Games. We present a detailed exposition of some of the most important concepts and results in [5] and [7]. Among these concepts, the guessing number of a digraph is of special interest; among these results, we highlight those that connect Guessing Games with Network Coding and Index Coding.

Chapter 1

Preliminaries

1.1 Alphabets and Finite Fields

In this section, we introduce some of the notation and basic terminology we will use throughout this monograph. We use the notation $\mathbb{Z}^+$ for the set of the positive integers, and $[n]$ for the set of the first n positive integers, that is, $[n] := \{1, 2, \ldots, n\}$. As it is usual, $\mathbb{F}_q$ represents the finite field with q elements, where $q = p^m$ for some prime p and positive integer m. We use the notation $\mathbb{Z}_n$ to represent the ring of integers modulo n.

An *alphabet* is a finite set with two or more elements. It is sometimes convenient to treat an alphabet $\mathcal{A}$ with s elements as it were the ring $\mathbb{Z}_s$. The following procedure justifies the preceding assertion: Let f be a bijection between $\mathcal{A}$ and $\mathbb{Z}_s$, then for $a, b \in \mathcal{A}$, we define $a \oplus b := f^{-1}(f(a) + f(b))$ and $a \otimes b := f^{-1}(f(a)f(b))$. Clearly, $\mathcal{A}$ with these operations forms a ring isomorphic to $\mathbb{Z}_s$. Along the same lines, if $s = p^m$ for some prime p and positive integer m, then $\mathcal{A}$ can be treated as $\mathbb{F}_{p^m}$.

If A and B are two non-empty finite sets, we denote the set of all functions $x : A \longrightarrow B$ as B^A. For $x \in B^A$ and $a \in A$, sometimes we use x_a to represents the value that takes a under x instead of $x(a)$; moreover, we use the notation $x = (x_a)_{a \in A}$ to represent the function $x \in B^A$. Also, if S is a non-empty subset of A, $x_S : S \longrightarrow B$ is the restriction of $x \in B^A$ to S; we also use the notation $x|_S$ for this purpose when the function already has a subscript.

Also, as usual, $\mathcal{A}^n := \{(x_1, \ldots, x_n) : x_1, \ldots, x_n \in \mathcal{A}\}$. If $x \in \mathcal{A}^n$, then x_i is the

ith-component of x; moreover, if $J = \{j_1, \ldots, j_k\} \subseteq [n]$ where $j_1 < \ldots < j_k$ and

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \in \mathcal{A}^n,$$

then $x_J \in \mathcal{A}^k$ is defined as

$$x_J := \begin{pmatrix} x_{j_1} \\ \vdots \\ x_{j_k} \end{pmatrix}.$$

Finally, we use $M_{m\times n}(\mathbb{F})$ for the set of $m \times n$ matrices with coefficients in the field $\mathbb{F}$; when $m = n$, we simply use $M_n(\mathbb{F})$. If $A \in M_{m\times n}(\mathbb{F})$, then A_{ij} is the ij-entry of A.

Lemma 1.1.1. *If $A \in M_n(\mathbb{F}_q)$, then* $\dim \operatorname{Kernel}(A) = \log_q |\operatorname{Kernel}(A)|$.

Proof. If $\dim \operatorname{Kernel}(A) = k$, then $\operatorname{Kernel}(A)$ is isomorphic to $\mathbb{F}_q^k$; therefore, $|\operatorname{Kernel}(A)| = q^k$ and $\log_q |\operatorname{Kernel}(A)| = k$. □

Definition 1.1.2. Let $\mathbb{F}$ be a finite field. For $A, B \in M_n(\mathbb{F})$, we say that $A \leq B$ if $A_{ij} = 0$ whenever $B_{ij} = 0$.

Example 1.1.3. Consider the field $\mathbb{F}_2$, and the matrices

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}, B = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \text{ and } C = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}.$$

Then, $A \leq B$ and $A \leq C$, but B and C are incomparable.

1.2 Linear Algebra

Let V be a finite vector space over the field $\mathbb{F}$. It is well known that the set V^* of all linear transformation from V to $\mathbb{F}$ is also a vector space over $\mathbb{F}$ and $\dim V = \dim V^*$.

We now introduce some basic results about bilinear forms. You may find these results in some linear algebra textbooks; we recommend [6].

Definition 1.2.1 (Bilinear Form)**.** Let V be a vector space over $\mathbb{F}$. The function $\langle\ ,\ \rangle : V \times V \longrightarrow \mathbb{F}$ is a *bilinear form* over V if

$$\langle \alpha x + \beta y, z\rangle = \alpha\langle x, z\rangle + \beta\langle y, z\rangle$$

and

$$\langle z, \alpha x + \beta y\rangle = \alpha\langle z, x\rangle + \beta\langle z, y\rangle,$$

for all $x, y, z \in V$ and $\alpha, \beta \in \mathbb{F}$.

Furthermore, if $\langle w, v\rangle = 0$ whenever $\langle v, w\rangle = 0$, then $\langle\ ,\ \rangle$ is called a reflexive bilinear form.

Definition 1.2.2 (Orthogonal Complement)**.** Let V be a vector space over $\mathbb{F}$, and let $\langle\ ,\ \rangle : V \times V \longrightarrow \mathbb{F}$ be a reflexive bilinear form over V. If W is a subspace of V, the *orthogonal complement* of W is defined as

$$W^{\perp} := \{x \in V : \langle w, x\rangle = 0, \forall w \in W\}.$$

It is easy to show that if W is a subspace of V, then $W^{\perp}$ is also a subspace of V and $W \subseteq (W^{\perp})^{\perp}$.

Theorem 1.2.3 (Riesz Representation Theorem)**.** *Let V be a finite vector space over $\mathbb{F}$, and let $\langle\ ,\ \rangle : V \times V \longrightarrow \mathbb{F}$ be a reflexive bilinear form over V such that $V^{\perp} = \{0\}$. For each $f \in V^*$ there is a unique $z \in V$ such that $f(x) = \langle x, z\rangle$, for all $x \in V$.*

Proof. For each $v \in V$, we consider the function $\varphi_v : V \longrightarrow \mathbb{F}$ where $\varphi_v(x) := \langle x, v\rangle$, for all $x \in V$. From the definition of bilinear form, it is easy to check that $\varphi_v \in V^*$, for all $v \in V$. It is also easy to check that the function

$$\begin{aligned} \psi : V &\longrightarrow V^* \\ v &\longmapsto \varphi_v \end{aligned}$$

is linear and

$$\begin{aligned} \text{Kernel}(\psi) &= \{v \in V : \varphi_v = 0\} \\ &= \{v \in V : \langle x, v\rangle = 0, \forall x \in V\} \\ &= V^{\perp} \\ &= \{0\}. \end{aligned}$$

Therefore, ψ is injective and $\dim \psi(V) = \dim V = \dim V^*$. Since $\psi(V)$ is a subspace of V^*, then $\psi(V) = V^*$, that is, ψ is surjective.

Let $f \in V^*$. Since ψ is surjective, there is $z \in V$ such that $f(x) = \langle x, z\rangle$, for all $x \in V$; since ψ is injective, z is unique. □

Proposition 1.2.4. *Let V be a finite vector space over $\mathbb{F}$, and let $\langle\ ,\ \rangle : V \times V \longrightarrow \mathbb{F}$ be a reflexive bilinear form over V. If $V^{\perp} = \{0\}$, then for every subspace W we have:*

a) $\dim W + \dim W^{\perp} = \dim V$.

b) $(W^{\perp})^{\perp} = W$.

Proof. a) We consider the function φ_v defined in the proof of the last theorem. It is readily seen that $\varphi_v|_W \in W^*$. Let

$$\begin{aligned} \rho : V &\longrightarrow W^* \\ v &\longmapsto \varphi_v|_W. \end{aligned}$$

We notice that ρ is linear and

$$\text{Kernel}\,\rho = \{v \in V : \varphi_v|_W = 0\} = \{v \in V : \langle w, v\rangle = 0, \forall w \in W\} = W^{\perp}.$$

Also, $\rho(V) = W^*$. Indeed, let $f \in W^*$; since $\langle\ ,\ \rangle|_{W\times W}$ is a bilinear form over W, we can use the last theorem to obtain a $z \in W$ such that $f(x) = \langle x, z\rangle|_{W\times W} = \langle x, z\rangle$, for all $x \in W$. It is clear, then, that $\rho(z) = f$.

Using the Rank-Nullity Theorem, we obtain

$$\begin{aligned} \dim V &= \dim \rho(V) + \dim \text{Kernel}\,\rho \\ &= \dim W^* + \dim W^{\perp} \\ &= \dim W + \dim W^{\perp}. \end{aligned}$$

b) We know that

$$\dim W + \dim W^{\perp} = \dim V.$$

Applying a) to $W^{\perp}$, we obtain

$$\dim W^{\perp} + \dim (W^{\perp})^{\perp} = \dim V.$$

Using these two equalities, we have $\dim W = \dim (W^{\perp})^{\perp}$. Since $W \subseteq (W^{\perp})^{\perp}$, then $W = (W^{\perp})^{\perp}$.

□

The following bilinear form will be useful in the proof of Theorem 3.3.5 in Chapter 3. Let $\mathbb{F}$ be field. For $x, y \in \mathbb{F}^n$, we define

$$x \bullet y := x_1y_1 + x_2y_2 + \ldots + x_ny_n.$$

It is readily seen that the mapping $(x, y) \longmapsto x \bullet y$ defines a reflexive bilinear form over $\mathbb{F}$. In this case, $(\mathbb{F}^n)^{\perp} = 0$. Indeed, $0 \in (\mathbb{F}^n)^{\perp}$, and if $x \in (\mathbb{F}^n)^{\perp}$, then $x_i = x \bullet e_i = 0$, for all $i \in [n]$, that is, $x = 0$ ($e_i \in \mathbb{F}^n$ is the vector whose ith-component is 1 and whose other components are 0). By part b) of the above proposition, $W = (W^{\perp})^{\perp}$, for all subspace W of $\mathbb{F}^n$.

1.3 Graph Theory

Definition 1.3.1 (Digraph)**.** The pair $D = (V, E)$, where V is a nonempty finite set and E is a subset of $(V \times V) - \{(v, v) : v \in V\}$, is called a *digraph*. The elements of V and E are called the *vertices* and *edges* of D, respectively.

If $e = (a, b)$ is and edge of D, we call a and b the *tail* and the *head* of e, respectively. We use the notation $\text{tail}(e) = a$ and $\text{head}(e) = b$.

Sometimes the vertices and edges of D will be denoted by $V(D)$ and $E(D)$, respectively.

Definition 1.3.2. Let $D = (V, E)$ be a digraph, and let v be a vertex of D. We define $N_D(v) := \{w \in V : (w, v) \in E\}$ and $N_D^+(v) := \{w \in V : (v, w) \in E\}$.

If we are working with a fixed digraph, we simply use $N(v)$ and $N^+(v)$, instead of $N_D(v)$ and $N_D^+(v)$.

Definition 1.3.3. Let $D = (V, E)$ be a digraph. For every edge $e = (a, b)$ of D, we define

$$\text{In}(e) := \{e' \in E : \text{head}(e') = a\}.$$

Also, if $v \in V$, we define

$$\text{In}(v) := \{e' \in E : \text{head}(e') = v\}.$$

Definition 1.3.4 (Complete Digraph)**.** The digraph $D = (V, E)$ where $V = [n]$ and $E = (V \times V) - \{(v, v) : v \in V\}$ is called the complete digraph on n vertices and is denoted by K_n.

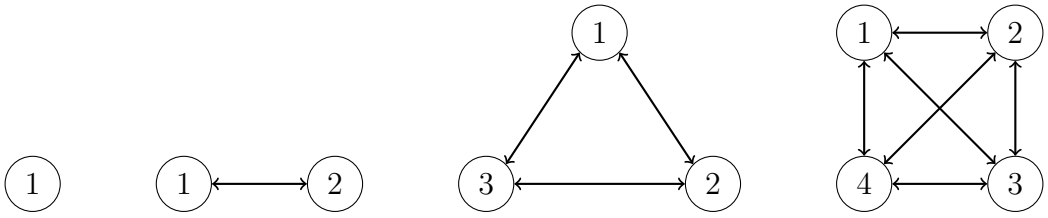

Figure 1: K_1, K_2, K_3, and K_4.

Definition 1.3.5 (Subdigraph)**.** Let D be a digraph. A subdigraph of D is a digraph H such that $V(H) \subseteq V(D)$ and $E(H) \subseteq E(D)$.

Definition 1.3.6 (Directed Cycle)**.** The digraph $C = (V, E)$ is called a directed cycle if $E = \{(v_1, v_2), (v_2, v_3), \ldots, (v_n, v_1)\}$, where $V = \{v_1, v_2, \ldots, v_n\}$. Furthermore, if C is a subdigraph of D, then C is called a directed cycle of D.

Definition 1.3.7 (Acyclic Digraph)**.** A digraph is acyclic if it has no directed cycles.

Definition 1.3.8 (Induced Subdigraph)**.** Let $D = (V, E)$ be a digraph. If $W \subseteq V$, then $D[W] := (W, E_W)$, where $E_W := \{(v, w) \in W \times W : (v, w) \in E\}$, is called the subdigraph of D induced by W.

Definition 1.3.9. For any digraph D and $S \subseteq V(D)$, we define the digraph $D - S := D[V(D) - S]$.

Definition 1.3.10 (Disjoint Union)**.** Let $D = (V, E)$ and $D' = (V', E')$ be two vertex-disjoint digraphs, that is, $V \cap V' = \emptyset$ (and therefore $E \cap E' = \emptyset$). The digraph $D \cup D' = (V \cup V', E \cup E')$ is called the *disjoint union* between D and D'.

Definition 1.3.11. Let $D = (V, E)$ be a digraph. We define $\nu(D)$ to be the maximum number m such that there are m pairwise vertex-disjoint directed cycles $C_1, C_2, \ldots, C_m$ of D, that is, the maximum number of pairwise vertex-disjoint directed cycles of D we can take. We define $\tau(D)$ to be the size of the smallest subset S of V such that $D - S$ is acyclic, that is, the minimum number of vertices we need to remove from D to make it acyclic.

Sometimes, like in the following definition, it is convenient to suppose that for a given digraph D, $V(D) = [n]$ where n is the number of vertices of D.

Definition 1.3.12 (Adjacent Matrix)**.** Let $D = ([n], E)$ be a digraph. The adjacent matrix A_D of D is defined to be the $n \times n$ matrix whose ij-entry is 1 if $(i, j) \in E$, and 0 otherwise.

Notice that since the adjacent matrix's entries are 0s or 1s, then we can see it as a matrix over any field. If we are working with a fixed field $\mathbb{F}$, then it will be understood that $A_D \in M_n(\mathbb{F})$. For the rest of this section we assume that $V(D) = [n]$, for some $n \in \mathbb{Z}^+$.

Definition 1.3.13. Let D be a digraph with n vertices and adjacent matrix A_D, and let $\mathbb{F}$ be a finite field. We say that $A \in M_n(\mathbb{F})$ fits D, if $A_{ii} = 1$ for all $1 \leq i \leq n$ and $A_{ij} = 0$ whenever $(A_D)_{ij} = 0$ for $i \neq j$.

Definition 1.3.14. Let D be a digraph with adjacent matrix A_D, and let $\mathbb{F}_q$ be a finite field. We define

$$\operatorname{minrank}(D, q) := \min_{A \text{ fits } D} \operatorname{rank} A.$$

In the linear case, the *minrank* plays a central role connecting Index Coding and Guessing Games.

Lemma 1.3.15. *Let D be a digraph with adjacent matrix A_D, and let $\mathbb{F}_q$ be a finite field. Then*

$$\operatorname{minrank}(D, q) = \min_{A \leq A_D} \operatorname{rank}(A + I) = \min_{A \leq A_D} \operatorname{rank}(A - I).$$

Proof. The first equality follows directly from definitions 1.1.2 and 1.3.13. Since $A \leq A_D$ iff $-A \leq A_D$, and $\operatorname{rank} B = \operatorname{rank}(-B)$, it follows that

$$\min_{A \leq A_D} \operatorname{rank}(A + I) = \min_{A \leq A_D} \operatorname{rank}(-A + I) = \min_{A \leq A_D} \operatorname{rank}(A - I).$$

□

Definition 1.3.16. Let $D = (V, E)$ be a digraph. We define the digraph $D^T := (V, E^T)$, which has same vertices of D and set of edges $E^T := \{(w, v) \in V \times V : (v, w) \in E\}$. That is, D^T is obtained from D by reversing the direction of each edge of D.

Lemma 1.3.17. $\operatorname{minrank}(D, q) = \operatorname{minrank}(D^T, q)$.

Proof. Clearly, A fits D iff A^T fits D^T; hence

$$\begin{aligned}
\operatorname{minrank}(D, q) &= \min_{A \text{ fits } D} \operatorname{rank} A \\
&= \min_{A^T \text{ fits } D^T} \operatorname{rank} A \\
&= \min_{A^T \text{ fits } D^T} \operatorname{rank} A^T \\
&= \min_{A \text{ fits } D^T} \operatorname{rank} A \\
&= \operatorname{minrank}(D^T, q).
\end{aligned}$$

□

Chapter 2

Network Coding

2.1 Introduction

Suppose you have a network (digraph). In this network there are some nodes (vertices) called sources and some called receivers. Each receiver demands a message generated by one of the sources. We codify the messages using the elements of an alphabet $\mathcal{A}$. Sometimes, it is convenient that the alphabet is an algebraic structure, for example a finite field. Information must travel from the sources to the receivers trough the channels (edges) of the network in order to satisfy the receivers' demands. But just only one element from $\mathcal{A}$ can travel through each channel.

Suppose that each node is just only able to replicate the data it generates or receives from the incoming channels, and send these replications through the outgoing channels. This scenario may cause some bottlenecks, and some receivers may not receive their messages. As an example, consider the network in Figure 2 called the Butterfly Network, which is the standard example to introduce Network Coding.

Here, the receivers r_1 and r_2 demand the messages generated by the sources s_1 and s_2, respectively. That is, if s_1 generates $X \in \mathcal{A}$ and s_2 generates $Y \in \mathcal{A}$, then r_1 and r_2 need to receive X and Y, respectively (for all $X, Y \in \mathcal{A}$). Notice that a bottleneck occurs on edge e_5, and therefore just only one receiver is able to obtain its message (depending whether you send X or Y through channel e_5).

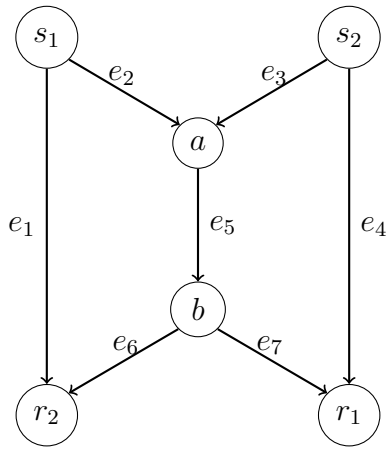

Figure 2: Butterfly Network

But Network Coding, which was introduced in [1], is a recently new paradigm that solves the above difficulty. The key of Network Coding is that you can operate data and not just only replicate it. To exemplify this, consider that $\mathcal{A}$ is $\mathbb{Z}_n$. We send X through edges e_1 and e_2, and Y through e_3 and e_4. Node a receives X and Y, and sends $X+Y$ through e_5. See Figure 3. Since receiver r_1 receives Y and $X+Y$, r_1 can recover $X = (X+Y)-Y$. Also, r_2 can recover $Y = (X+Y)-X$.

The purpose of Network Coding is to determine how the nodes of the network need to operate the data so that the receivers' demands can be satisfied.

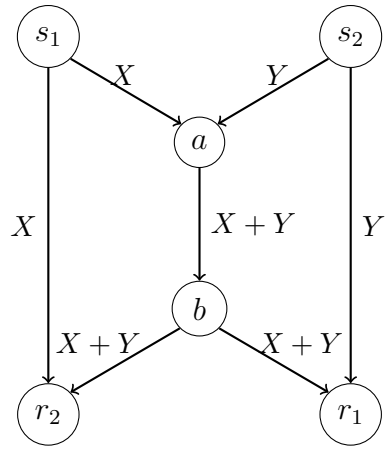

Figure 3

2.2 Networks, Network Codes and Solutions

Since we want to study Network Coding from a mathematical perspective, we have to give rigorous definitions of networks, sources, receivers, and network codes. Readers should be aware that different papers on Network Coding define these concepts in slightly different ways. The definitions presented here are the most suitable for the purpose of this monograph.

Definition 2.2.1. Let D be a digraph. A vertex s of D is called a *source* if $N(s) = \emptyset$. On the other hand, a vertex r of D is called a *receiver* if $N_+(r) = \emptyset$.

Lemma 2.2.2. *An acyclic digraph always has at least one source and one receiver.*

Proof. Let D be an acyclic digraph. We will prove that D has at least one source (the proof that D has at least one receiver is similar).

Let v_0 be any vertex of D. If v_0 is a source, we are done; otherwise, there is $v_1 \in V(D)$ such that $v_1 \in N(v_0)$ and $v_1 \notin \{v_0\}$ since D is acyclic. If v_1 is a source, we are done; otherwise, there is $v_2 \in V(D)$ such that $v_2 \in N(v_1)$ and $v_2 \notin \{v_0, v_1\}$ since D is acyclic. This process ends since $V(D)$ is finite; therefore, we can always find a source. □

Definition 2.2.3 (Network)**.** Let D be an acyclic digraph, and let $\tau : R \longrightarrow S$ be a surjective function from the set of receivers R onto the set of sources S. The pair $\mathcal{N} = (D, \tau)$ is called a *network*.

A vertex of D is also called a vertex of $\mathcal{N}$; likewise, an edge of D is also called an edge of $\mathcal{N}$. Finally, the set of sources and the set of receivers of D will be denoted by $S(\mathcal{N})$ and $R(\mathcal{N})$, respectively; by lemma 2.2.2 we know that both $S(\mathcal{N})$ and $R(\mathcal{N})$ are nonempty.

According to the problem presented in the introductory section, $\tau(r) = s$ means that the receiver r required the message generated by the source s.

Before giving a rigorous definition of a network code, we notice that Network Coding is only necessary on edges where a bottleneck occurs. This motivates the following definition.

Definition 2.2.4. Let $\mathcal{N}$ be a network. The set $C(\mathcal{N})$ is defined as the set of all edges e of $\mathcal{N}$ such that $|\operatorname{In}(e)| \geq 2$.

Definition 2.2.5 (Network Code)**.** Let $\mathcal{N}$ be a network. A *network code* $\mathcal{F}$ over the alphabet $\mathcal{A}$ is a collection of functions

$$f_e : \mathcal{A}^{\operatorname{In}(e)} \longrightarrow \mathcal{A},$$

one for each edge e in $C(\mathcal{N})$, and functions

$$f_r : \mathcal{A}^{\mathrm{In}(r)} \longrightarrow \mathcal{A},$$

one for each receiver r.

We will use the notation $\mathcal{F} = (f_e, f_r)_{e \in C(\mathcal{N}), r \in R(\mathcal{N})}$ to denote a network code.

Definition 2.2.6 (Message assignment)**.** Let $\mathcal{N}$ be a network with set of sources S, and let $\mathcal{A}$ be an alphabet. A function $\omega : S \longrightarrow \mathcal{A}$ is called a *message assignment* for the network $\mathcal{N}$ over the alphabet $\mathcal{A}$.

A message assignment is a way of codifying the messages generated by the sources, more precisely, the message generated by source s is represented by the element $\omega(s) \in \mathcal{A}$. The receivers need to recover the messages they required regardless the codification used, that is, regardless the message assignment. Obviously the information that travels trough each edge depends on the message assignment and the network code used. This motivates the following definitions.

Definition 2.2.7. Let $\mathcal{N}$ be an network, and let $\mathcal{F} = (f_e, f_r)_{e \in C(\mathcal{N}), r \in R(\mathcal{N})}$ be a network code over the alphabet $\mathcal{A}$. For each edge e and each message assignment ω for the network $\mathcal{N}$ over the alphabet $\mathcal{A}$, we define $\bar{e}(\omega)$ recursively as follows.

If $|\operatorname{In}(e)| = 0$, that is, $e = (s, v)$ for some source s, then $\bar{e}(\omega) := \omega(s)$.

If $|\operatorname{In}(e)| = 1$, that is, $\operatorname{In}(e) = \{e'\}$ for some edge e', then $\bar{e}(\omega) := \bar{e'}(\omega)$.

If $|\operatorname{In}(e)| \geq 2$, that is, $e \in C(\mathcal{N})$, then $\bar{e}(\omega) := f_e((\bar{e'}(\omega))_{e' \in \operatorname{In}(e)})$.

Finally, for each receiver r and each message assignment ω for the network $\mathcal{N}$ over the alphabet $\mathcal{A}$, we define $\bar{r}(\omega) := f_r((\bar{e'}(\omega))_{e' \in \operatorname{In}(r)})$.

We notice that $\bar{e}(\omega)$ and $\bar{r}(\omega)$ are well defined since we are working with acyclic digraphs.

Definition 2.2.8 (Solution)**.** Let $\mathcal{N}$ be a network. A *network code* $\mathcal{F}$ over an alphabet $\mathcal{A}$ is a *solution* of $\mathcal{N}$ over the alphabet $\mathcal{A}$, if $\bar{r}(\omega) = \omega(\tau(r))$, for every message assignment $\omega \in \mathcal{A}^{S(\mathcal{N})}$ and every receiver r.

If a solution of $\mathcal{N}$ over the alphabet $\mathcal{A}$ exists, we say that $\mathcal{N}$ is solvable over $\mathcal{A}$.

Sometimes, it is convenient to suppose that the edges of $\mathcal{N} = (D, \tau)$ are ordered, that is, $E(D) = \{e_1, \ldots, e_m\}$. This order induces an order in each of the sets $\operatorname{In}(e)$ and $\operatorname{In}(r)$. In this way, we can work with the sets $\mathcal{A}^{|\operatorname{In}(e)|}$ and $\mathcal{A}^{|\operatorname{In}(r)|}$ instead of $\mathcal{A}^{\operatorname{In}(e)}$ and $\mathcal{A}^{\operatorname{In}(r)}$. This changes the domains of the functions f_e and f_r

of a network code. Moreover, if $S(\mathcal{N}) = \{s_1, s_2, \ldots, s_k\}$ is ordered, we can suppose that message assignments are elements of $\mathcal{A}^k$ instead of $\mathcal{A}^{S(\mathcal{N})}$.

Finally, if the alphabet is a finite field and the edges and sources are ordered, then a network code $\mathcal{F} = (f_e, f_r)_{e \in C(\mathcal{N}), r \in R(\mathcal{N})}$ is said to be *linear* if all the functions f_e and f_r are linear. If a linear solution of $\mathcal{N}$ over the finite field $\mathbb{F}$ exists, we say that $\mathcal{N}$ is linearly solvable over $\mathbb{F}$.

2.3 Multiple-Unicast Networks

Definition 2.3.1 (Multiple-Unicast Network)**.** A network $\mathcal{N} = (D, \tau)$ is called a *multiple-unicast network* if τ is bijective.

We now present a result from [4]. This result shows us how to reduce the study of general networks to the study of multiple-unicast networks.

Theorem 2.3.2. *For every network $\mathcal{N}$, there is a multiple-unicast network $\mathcal{N}'$ such that:*

(a) If $\mathcal{A}$ is an alphabet, then $\mathcal{N}$ is solvable over $\mathcal{A}$ iff $\mathcal{N}'$ is solvable over $\mathcal{A}$.

(b) If $\mathbb{F}$ is finite field, then $\mathcal{N}$ is linearly solvable over $\mathbb{F}$ iff $\mathcal{N}'$ is linearly solvable over $\mathbb{F}$.

Proof. There is nothing to prove if $\mathcal{N}$ is already a multiple-unicast network. If $\mathcal{N} = (D, \tau)$ is not a multiple-unicast network is because there are two receivers r_1 and r_2 such that $\tau(r_1) = \tau(r_2) = s_1$ for some source s_1. Consider now the network $\mathcal{N}_1 = (D_1, \tau_1)$ that is obtained from $\mathcal{N}$ by adding the gadget depicted in Figure 4.

From now on, we will use the notation in Figure 4. Notice that this new network has a new source s_0 and two new receivers r and r_0 (r_1 and r_2 are no longer receivers). The message generated by s_0 and s_1 are demanded by r_0 and r, respectively; that is, $\tau_1(r_0) = s_0$ and $\tau_1(r) = s_1$. If $\mathcal{F}$ is a solution to $\mathcal{N}$, figure 5 depicts an extension of $\mathcal{F}$ that solves $\mathcal{N}_1$. It is clear that if $\mathcal{F}$ is linear, the extension is also linear.

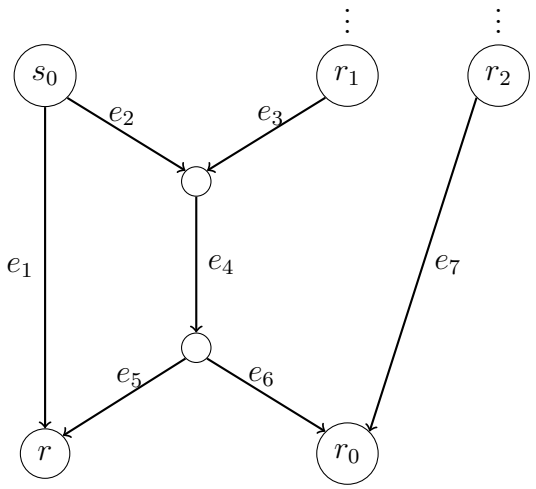

Figure 4

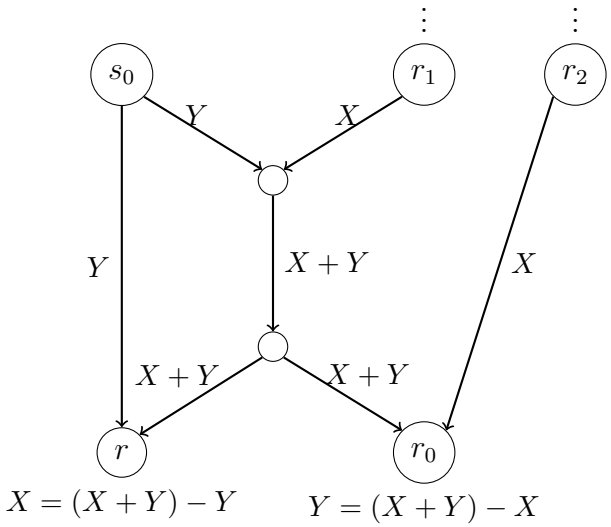

Figure 5

On the other hand, let $\mathcal{F}'$ be a solution of $\mathcal{N}_1$. We need to construct a solution $\mathcal{F} = (f_e, f_v)_{e \in C(\mathcal{N}), v \in R(\mathcal{N})}$ of $\mathcal{N}$. We take $f_e := f'_e$, for each $e \in C(\mathcal{N})$, where f'_e is the function of $\mathcal{F}'$ associated to the edge e. We also take $f_v := f'_v$, for each $v \in R(\mathcal{N}) - \{r_1, r_2\}$, where f'_v is the function of $\mathcal{F}'$ associated to the receiver v. According to this, we only need to construct the decoding functions for the receivers r_1 and r_2.

Let f, g, and h be the functions of $\mathcal{F}'$ associated to the edge e_4, the receiver r, and the receiver r_0, respectively. Let $S(\mathcal{N}_1) = \{s_0, s_1, \ldots, s_k\}$. Since $\mathcal{F}'$ is solution of $\mathcal{N}_1$, then for every message assignment $\omega = (m_0, m_1, \ldots, m_k) \in \mathcal{A}^{k+1}$ we have:

$$\bar{e_4}(\omega) = f(m_0, \bar{e_3}(\omega)), \tag{1}$$
$$m_1 = g(m_0, \bar{e_4}(\omega)), \tag{2}$$
$$m_0 = h(\bar{e_4}(\omega), \bar{e_7}(\omega)). \tag{3}$$

From (1) and (2),

$$m_1 = g(m_0, f(m_0, \bar{e_3}(\omega))). \tag{4}$$

From (2) and (3),

$$m_1 = g(h(\bar{e_4}(\omega), \bar{e_7}(\omega)), \bar{e_4}(\omega)). \tag{5}$$

Observe that $\bar{e_3}(\omega)$ and $\bar{e_7}(\omega)$ depend only on $m_1, \ldots, m_k$ and not on m_0. If d' is the function of $\mathcal{F}'$ associated to e_3 (d' is the identity in case that $e_3 \notin C(\mathcal{N}_1)$), we can see from (4) that, for a fixed $m_0 \in \mathcal{A}$, $d_1 := g(m_0, f(m_0, d'))$ works as decoding function for the sink r_1 of the network $\mathcal{N}$ (In the linear case, take $m_0 = 0$, so d_1 is linear).

Let $\gamma = (m_1, \ldots, m_k) \in \mathcal{A}^k$. For every $m \in \mathcal{A}$, we define

$$\gamma_m := (m, m_1, \ldots, m_k) \in \mathcal{A}^{k+1}$$

and $e_4^\gamma(m) := \bar{e_4}(\gamma_m)$.

If $e_4^\gamma(m) = e_4^\gamma(m')$, that is, $\bar{e_4}(\gamma_m) = \bar{e_4}(\gamma_{m'})$, then

$$h(\bar{e_4}(\gamma_m), \bar{e_7}(\gamma_m)) = h(\bar{e_4}(\gamma_{m'}), \bar{e_7}(\gamma_{m'})),$$

since $\bar{e_7}(\gamma_m) = \bar{e_7}(\gamma_{m'})$ ($\bar{e_7}$ depends only on γ). By (3), $m = m'$. So $e_4^\gamma : \mathcal{A} \longrightarrow \mathcal{A}$ is a one-to-one function and therefore bijective (since $\mathcal{A}$ is finite) for every $\gamma \in \mathcal{A}^k$.

We fix $\alpha \in \mathcal{A}$. If $\gamma = (m_1, \ldots, m_k) \in \mathcal{A}^k$, there is $m_0 \in \mathcal{A}$, such that $\bar{e_4}(\gamma_{m_0}) = e_4^\gamma(m_0) = \alpha$. From (5), it follows that $m_1 = g(h(\alpha, \bar{e_7}(\gamma_{m_0})), \alpha)$. If d'' is the function of $\mathcal{F}'$ associated to e_7 (remember that $\bar{e_7}$ does not depend on m_0), then $d_2 := g(h(\alpha, d''), \alpha)$ works as decoding function for the sink r_2 of the network $\mathcal{N}$ (In the linear case, take $\alpha = 0$, so d_2 is linear).

We continue adding gadgets until each receiver demands a different message. $\square$

Chapter 3

Index Coding

3.1 Introduction

Index Coding, which was introduced in [3], is a relatively new area that has an impact on how information is transmitted. In general terms, there is a group of n clients each one demanding specific information, which is different for each client. A broadcast source in order to satisfy the clients' demands needs to do n transmissions, one for each client. Broadcast means that each transmission reaches to every client, so each client receive unnecessary information. But if each client already knows information required by the other clients, Index Coding tries to take advantage of this in order to reduce the number of transmissions necessary to satisfy the clients' demands.

We now explain how the above problem can be reduced to the study of an appropriate digraph D. The digraph is obtained as follows. We identify the clients with the vertices of D, where there is an edge from v to w if and only if the client w already knows the information required by the client v.

As an example consider a group of three clients. Let x_i be the information required by the ith-client. Suppose that each x_i belongs to $\mathbb{Z}_n$, that is, the information required has been codified using $\mathbb{Z}_n$. Also, each client knows the information of the other two clients, that is, the first client knows x_2 and x_3, the second client knows x_1 and x_3, and the third client knows x_1 and x_2. According to these conditions, it is clear that the associated digraph is K_3.

With three transmissions each client is able to obtain his or her information (send x_i in the ith-transmission). Nevertheless, we can do better. We can use only one transmission. The broadcast source only needs to send $x_1 + x_2 + x_3$. Since the first client knows $x_1 + x_2 + x_3$, x_2, and x_3, this client is able to recover

$x_1 = (x_1 + x_2 + x_3) - x_2 - x_3$. Similarly, the second and third client can recover x_2 and x_3, respectively.

It is not difficult to see that in the generalization of the above case where there are m clients, and each client knows the information of the other clients, we still need only one transmission.

3.2 Index Codes

In this chapter, definitions are based on paper [2]. We formalize the above problem with the following definition.

Definition 3.2.1 (Index Code)**.** Let $D = (V, E)$ be a digraph, and let $\mathcal{A}$ be an alphabet. An *index code* $\mathcal{C}$ of D over $\mathcal{A}$ is formed by:

i) an encoding function $f : \mathcal{A}^V \longrightarrow \mathcal{A}^c$, for some positive integer c.

ii) a decoding function $d_v : \mathcal{A}^c \times \mathcal{A}^{N(v)} \longrightarrow \mathcal{A}$, for each vertex v of D, which satisfies $d_v(f(x), x_{N(v)}) = x_v$ for every $x \in \mathcal{A}^V$.

The parameter c is called the length of $\mathcal{C}$ and denoted length($\mathcal{C}$).

An element $x = (x_v)_{v \in V} \in \mathcal{A}^V$ can be interpreted in the following way: the element x_v is the information required by client v. The number c plays the role of the number of transmissions. Finally, the condition $d_v(f(x), x_{N(v)}) = x_v$ tell us that client v can recover x_v using the information $f(x)$ transmitted by the broadcast channel and the information $x_{N(v)}$ he or she already knows. Our purpose is to minimize the number of transmissions necessary to satisfy the clients' demands, that is, to find the minimum c such that an index code of length c exists.

Definition 3.2.2 (Optimal Index Code)**.** Let D be a digraph, and let $\mathcal{A}$ be and alphabet of size s. We define

$$l(D, s) := \min\{\text{length}(\mathcal{C}) : \mathcal{C} \text{ is an index code of } D \text{ over } \mathcal{A}\}.$$

An index code $\mathcal{C}$ of D over $\mathcal{A}$ is optimal if length($\mathcal{C}$) $= l(D, s)$.

Since length($\mathcal{C}$) ≥ 1, for any index code $\mathcal{C}$ of D over $\mathcal{A}$, we have

$$l(D, s) \geq 1.$$

3.3 Linear Index Codes

In the linear case, it is convenient to assume that $V(D) = [n]$, for some $n \in \mathbb{Z}^+$. We maintain this assumption for the rest of this section. In this case, we redefine the definition of an index code when the alphabet is a finite field $\mathbb{F}$.

An index code is composed then by an encoding function $f : \mathbb{F}^n \longrightarrow \mathbb{F}^c$, where $c \in \mathbb{Z}^+$, and a decoding function $d_i : \mathbb{F}^c \times \mathbb{F}^{|N(i)|} \longrightarrow \mathbb{F}$ for each $i \in [n]$, such that $d_i(f(x), x_{N(i)}) = x_i$, $\forall x \in \mathbb{F}^n$.

Definition 3.3.1 (Linearly-encodable index codes)**.** Let D be a digraph with n vertices, and let $\mathbb{F}$ be a finite field. An index code $\mathcal{C}$ is called *linearly-encodable* if the encoding function f is linear, that is, there is a matrix $A \in M_{\text{length}(\mathcal{C})\times n}(\mathbb{F})$ such that $f(x) = Ax$ for all $x \in \mathbb{F}^n$.

Definition 3.3.2 (Linearly-decodable index codes)**.** Let D be a digraph with n vertices, and let $\mathbb{F}$ be a finite field. An index code $\mathcal{C}$ is called *linearly-decodable* if the decoding functions f_i are linear for all $i \in [n]$. That is, for each vertex $i \in [n]$, there is $a_i \in \mathbb{F}^{\text{length}(\mathcal{C})} \times \mathbb{F}^{|N(i)|}$ such that $d_i(x) = x \bullet a_i$ for all $x \in \mathbb{F}^{\text{length}(\mathcal{C})} \times \mathbb{F}^{|N(i)|}$.

Definition 3.3.3 (Linear index codes)**.** Let D be a digraph, and let $\mathbb{F}$ be a finite field. An index code $\mathcal{C}$ is called *linear* if it is linearly-encodable and linearly-decodable.

Definition 3.3.4. Let D be a digraph, and let $\mathbb{F}_q$ be a finite field. We define

$$l_{\text{linear}}(D, q) := \min\{\text{length}(\mathcal{C}) : \mathcal{C} \text{ is a linear index code of } D \text{ over } \mathbb{F}_q\}.$$

It is clear, then, that $1 \leq l(D, q) \leq l_{\text{linear}}(D, q)$.

We end this section with a result from [2].

Theorem 3.3.5. *Let D be a digraph, and let $\mathbb{F}_q$ be a finite field. Then*

$$l_{\text{linear}}(D, q) = \text{minrank}(D, q).$$

Proof. i) In the first place, we will prove that there is a linear index code of length $\text{minrank}(D, q)$ (therefore, $l_{\text{linear}}(D, q) \leq \text{minrank}(D, q)$).

Let A be a matrix that fits D and satisfies $\text{minrank}(D, q) = \text{rank}\, A$. If $k := \text{rank}\, A$, let $a_1, \ldots a_k$ be k linearly independent rows of A.

We define the linear encoding function

$$f = \begin{pmatrix} f_1 \\ \vdots \\ f_k \end{pmatrix}$$

as follows. We set (for all $j = 1, ..., k$) $f_j(x) := a_j \bullet x$, for all $x \in \mathbb{F}_q^n$, or equivalently,

$$f(x) := \begin{pmatrix} a_1^T \\ \vdots \\ a_k^T \end{pmatrix} x.$$

We know that each row A_i (the ith-row of A) can be expressed as $A_i = \sum_{j=1}^{k} \alpha_{ij} a_j$ ($\alpha_{ij} \in \mathbb{F}_q$). Therefore,

$$A_i \bullet x = \sum_{j=1}^{k} \alpha_{ij} (a_j \bullet x) = \sum_{j=1}^{k} \alpha_{ij} f_j(x). \tag{1}$$

We now define the linear decoding function for the vertex i as

$$d_i(f_1(x), \ldots, f_k(x), x_{v_1}, \ldots, x_{v_r}) := \sum_{j=1}^{k} \alpha_{ij} f_j(x) - \sum_{j=1}^{r} A_{v_j i} x_{v_j},$$

where $N(i) = \{v_1, \ldots, v_r\}$.

Finally, we prove that indeed we have and index code. For this, we need to show that $d_i(f_1(x), \ldots, f_k(x), x_{v_1}, \ldots, x_{v_r}) = x_i$, for all $i \in [n]$ and $x \in \mathbb{F}_q^n$. Taking into consideration that $A_{ti} = 0$ ($t \neq i$), if $t \notin N(i)$ (since A fits D), we have, for $x \in \mathbb{F}_q^n$,

$$(A_i - e_i) \bullet x = \sum_{j=1}^{r} A_{v_j i} x_{v_j}. \tag{2}$$

From (1) and (2), it follows that

$$\begin{aligned} d_i(f_1(x), \ldots, f_k(x), x_{v_1}, \ldots, x_{v_r}) &= \sum_{j=1}^{k} \alpha_{ij} f_j(x) - \sum_{j=1}^{r} A_{v_j i} x_{v_j} \\ &= A_i \bullet x - (A_i - e_i) \bullet x \\ &= e_i \bullet x \\ &= x_i. \end{aligned}$$

ii) We will proceed to prove that the length of any linear index code of D over $\mathbb{F}_q$ is great or equal to $\text{minrank}(D, q)$ (therefore, $\text{minrank}(D, q) \leq l_{\text{linear}}(D, q)$).

Consider the linear index code $\mathcal{C}$ of length c with encoding function

$$f = \begin{pmatrix} f_1 \\ \vdots \\ f_c \end{pmatrix},$$

and decoding functions $d_1, \ldots, d_n$.

Since f is linear, for each $k \in [c]$, there is a vector $a_k \in \mathbb{F}_q^n$ such that $f_k(x) = a_k \bullet x$ for all $x \in \mathbb{F}_q^n$. We fix $i \in [n]$ and set $W := \text{span}(\{a_1, \ldots, a_c\} \cup \{e_j : j \in N(i)\})$.

Suppose that $e_i \notin W$. Since $(W^\perp)^\perp = W$, we have $e_i \notin (W^\perp)^\perp$. So there is $y \in W^\perp$ such that $e_i \bullet y \neq 0$ and therefore

$$y_i \neq 0. \tag{3}$$

Since $y \in W^\perp$, it follows that

$$f_k(y) = a_k \bullet y = 0, \text{ for all } k \in [c]. \tag{4}$$

Also, it follows that $y \bullet e_j = 0$, for all $j \in N(i)$, that is,

$$y_j = 0, \text{ for all } j \in N(i). \tag{5}$$

Since $\mathcal{C}$ is an index code, $d_i(f(x), x_{N(i)}) = x_i$, for all $x \in \mathbb{F}_q^n$. In particular, for y and the zero vector $\mathbf{0}$ in $\mathbb{F}_q^n$, we have $d_i(f(y), y_{N(i)}) = y_i$ and $d_i(f(\mathbf{0}), \mathbf{0}_{N(i)}) = \mathbf{0}_i = 0$. But actually, from (4) and (5),

$$f(y) = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} = f(\mathbf{0})$$

and

$$y_{N(i)} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} = \mathbf{0}_{N(i)};$$

therefore, $y_i = 0$, contradicting (3).

We have proved that $e_i \in W$; therefore,

$$e_i = \sum_{k=1}^{c} \alpha_{ik} a_k + \sum_{k \in N(i)} \beta_{ik} e_k$$

for some α_{ik}'s and β_{ik}'s in $\mathbb{F}_q^n$. The vector

$$A_i := \sum_{k=1}^{c} \alpha_{ik} a_k = e_i - \sum_{k \in N(i)} \beta_{ik} e_k$$

belongs to the span of $\{a_1, \ldots, a_c\}$; hence, the matrix A with rows $A_1, A_2, \ldots, A_n$ satisfies

$$\operatorname{rank} A \leq c.$$

Also, A_i has value 0 in components outside $N(i) \cup \{i\}$ and 1 in the ith-component; therefore, A fits D. Thus,

$$\operatorname{minrank}(D, q) \leq \operatorname{rank} A.$$

Combining the last two inequalities, we obtain $\operatorname{minrank}(D, q) \leq c$.

□

Actually, part i) of the above proof shows us how to construct a linear index code of minimum length if we know a matrix A that fits D and such that $\operatorname{minrank}(D, q) = \operatorname{rank} A$. In the case of K_3,

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

According to part i), $f(x) := [1, 1, 1]^T x = x_1 + x_2 + x_3$; also,

$$\begin{aligned} d_1(f(x), x_2, x_3) &:= f(x) - x_2 - x_3, \\ d_2(f(x), x_1, x_3) &:= f(x) - x_1 - x_3, \\ d_3(f(x), x_1, x_2) &:= f(x) - x_1 - x_2. \end{aligned}$$

We also see that $1 \leq l(K_3, q) \leq l_{\text{linear}}(K_3, q) = \operatorname{minrank}(K_3, q) = 1$; hence, $l(K_3, q) = l_{\text{linear}}(K_3, q) = 1$.

Chapter 4

Guessing Games

4.1 Introduction

Consider the following puzzle from [5]: In a group of n players, each player has a number from $\{0, 1, \ldots, s-1\}$ ($s \geq 2$) on his forehead, selected uniformly at random. Each player is able to see the numbers of the others but not his or her own number. Now, each player has to guess simultaneously his or her own number. All players win if all of them guess correctly, otherwise they lose. Is there any strategy (no communication allow, that is, each player's guess is a function of the information the player already knows, in this case, the numbers of the other players) that guarantees success with a probability higher than $(1/s)^n$? Find a strategy with the highest probability of winning.

Before giving a solution, we consider now a more general type of game called *guessing game*, which was introduced in [5]: Let D be a digraph, and let $\mathcal{A}$ be an alphabet of size s. Each vertex represents a player, and we assign to each them a number, selected uniformly at random, from $\mathcal{A}$. For each $v \in V(D)$, the player v can only see those numbers assigned to the elements of $N(v)$. Again, all players win if all of them guess correctly, and we are interested in finding a strategy with the highest probability of winning.

Notice that the puzzle corresponds to the guessing game played over the complete graph K_n with an alphabet of s elements. We present now the solution from [5] of the above puzzle.

Since every player does not know his or her own number, then the probability of a player to guess correctly his or her own number is $1/s$. Let $x_i \in \{0, 1, \ldots, s-1\}$ be the number assigned to the i-th player. Let

$$y_i := -(x_1 + x_2 + \cdots + x_{i-1} + x_{i+1} + \cdots + x_{n-1} + x_n)$$

(where the sum is over $\mathbb{Z}_s$), if the i-th player guesses that $x_i = y_i$ for all $i \in [n]$, we claim that this is the best strategy players can use.

If $x_1 = y_1$, then $x_1 + x_2 + \cdots + x_n = 0$; therefore,

$$x_i = -(x_1 + x_2 + \cdots + x_{i-1} + x_{i+1} + \cdots + x_{n-1} + x_n),$$

for all $i \in [n]$. The above affirmation says that all players guess correctly with this strategy if the first player guesses correctly. Moreover, it is clear that the the first player guesses correctly with this strategy if all player guess correctly. We conclude, then, that the probability of the first player to guess correctly is equal to the probability of all players to guess correctly. Since the probability of the first player to guess correctly is $1/s$, the probability of all players to guess correctly is also $1/s$.

4.2 Strategies and the Guessing Number

In this and the next section, we present some definitions from [7] of concepts related to guessing games.

Definition 4.2.1 (Configuration)**.** Let D be a digraph, and let $\mathcal{A}$ be an alphabet. A *configuration* is a function from $V(D)$ to $\mathcal{A}$. The set of all configuration will be denoted by $\Omega(D, \mathcal{A})$.

Definition 4.2.2 (Strategy)**.** Let D be a digraph, and let $\mathcal{A}$ be an alphabet. A *strategy* $\mathcal{F}$ is a collection of functions $f_v : \mathcal{A}^{N(v)} \longrightarrow \mathcal{A}$, one for each vertex v of D. The set of strategies will be denoted by $\mathbb{S}(D, \mathcal{A})$.

Configurations and strategies depend on D and $\mathcal{A}$. Sometimes, the context makes clear the digraph and the alphabet we are working with, in that case, we use Ω and $\mathbb{S}$, instead of $\Omega(D, \mathcal{A})$ and $\mathbb{S}(D, \mathcal{A})$. Also, we use the notation $\mathcal{F} = (f_v)_{v \in V(D)}$ to express that the strategy $\mathcal{F}$ is formed by the functions f_v, $v \in V(D)$.

We use the following convention: if $v \in V(D)$ with $N(v) = \emptyset$, then f_v has to be necessarily a constant function.

Definition 4.2.3. Let $D = (V, E)$ be a digraph, and let $\mathcal{A}$ be an alphabet of size s. If $\mathcal{F} = (f_v)_{v \in V}$ is a strategy and Ω is the set of configurations, we define the *set of configurations fixed by* $\mathcal{F}$ as

$$\mathrm{Fix}(\mathcal{F}) := \{x \in \Omega : f_v(x_{N(v)}) = x_v \text{ for all } v \in V\},$$

and the *probability of success of* $\mathcal{F}$ as

$$P(\mathcal{F}) := \frac{|\,\mathrm{Fix}(\mathcal{F})|}{s^{|V|}}.$$

Definition 4.2.4 (Guessing Number)**.** Let D be a digraph, and let $\mathcal{A}$ be an alphabet of size s. If $\mathcal{F}$ is a strategy, we define $g(D, s, \mathcal{F})$ as the unique number such that

$$P(\mathcal{F}) = \frac{1}{s^{|V|-g(D,s,F)}},$$

or equivalently

$$g(D, s, \mathcal{F}) := \log_s |\operatorname{Fix}(\mathcal{F})|.$$

The number

$$g(D, s) := \max_{\mathcal{F}} g(D, s, \mathcal{F}) = \max_{\mathcal{F}} \log_s |\operatorname{Fix}(\mathcal{F})|,$$

where $\mathcal{F}$ runs over all strategies, is called the *guessing number*.

Example 4.2.5. Let $D = (V, E)$ be a digraph, and let $\mathcal{A}$ be an alphabet of size s. We define the strategy $\mathcal{F}_\sigma = (f_v)_{v\in V}$, where $\sigma \in \Omega$, as the strategy such that each f_v is constant and takes the value σ_v. It is clear that $\operatorname{Fix}(\mathcal{F}_\sigma) = \{\sigma\}$ and, therefore, $g(D, s, \mathcal{F}_\sigma) = 0$. These strategies are called constant strategies.

Definition 4.2.6 (Optimal Strategy)**.** Let D be a digraph, and let $\mathcal{A}$ be an alphabet of size s. A strategy $\mathcal{F}$ is said to be *optimal* if $g(D, s) = g(D, s, \mathcal{F})$.

4.3 Linear Strategies and the Linear Guessing Number

Definition 4.3.1 (Linear Strategy)**.** Let $D = (V, E)$ be a digraph, and let $\mathbb{F}$ be a finite field. A strategy $\mathcal{F} = (f_v)_{v\in V}$ is called a *linear strategy* if f_v is linear for all $v \in V$, that is, for each $v \in V$, there are $\alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_k} \in \mathbb{F}$, such that

$$f_v(x_{N(v)}) = \alpha_{i_1} x_{v_{i_1}} + \alpha_{i_2} x_{v_{i_2}} + \ldots + \alpha_{i_k} x_{v_{i_k}},$$

for all $x \in \Omega$, where $N(v) = \{v_{i_1}, v_{i_2}, \ldots, v_{i_k}\}$.

Definition 4.3.2 (Linear Guessing Number)**.** Let D be a digraph, and let $\mathbb{F}_q$ be a finite field. The linear guessing number is defined as

$$g_{\text{linear}}(D, q) := \max_{\mathcal{F} \text{ is linear}} g(D, q, \mathcal{F}) = \max_{\mathcal{F} \text{ is linear}} \log_q |\operatorname{Fix}(\mathcal{F})|,$$

where $\mathcal{F}$ runs over all linear strategies.

We have the obvious relation $g_{\text{linear}}(D, q) \leq g(D, q)$, when $q = p^m$ for some prime p and positive integer m.

The following theorem is from [7]; its relation with theorem 3.3.5 is evident.

Theorem 4.3.3. *Let $D = (V, E)$ be a digraph, and let $\mathbb{F}_q$ be a finite field. Then*

$$g_{\text{linear}}(D, q) = |V| - \text{minrank}(D, q).$$

Proof. Without loss of generality, suppose that $V = [n]$. In this way, the set of configuration Ω may be treated as $\mathbb{F}_q^n$. Then, a linear solution $\mathcal{F}$ is a collection of functions f_j, $j \in [n]$, such that for all $(x_1, \ldots, x_n) \in \mathbb{F}_q^n$

$$f_j(x_1, \ldots, x_n) = \sum_{i=1}^{n} \alpha_{ij} x_i$$

with $\alpha_{ij} = 0$ whenever $i \notin N(j)$ (remember that f_j depends only on the elements of $N(j)$). The matrix $A = [\alpha_{ij}]$ represents the linear solution $\mathcal{F}$ and satisfies $A \leq A_D$ (A_D is the adjacent matrix of D). Moreover, a matrix A such that $A \leq A_D$ defines a linear strategy. Therefore,

$$\begin{aligned} x \in \text{Fix}(\mathcal{F}) &\iff Ax = Ix \\ &\iff (A - I)x = O \\ &\iff x \in \text{Kernel}(A - I). \end{aligned}$$

By lemma 1.1.1, we conclude that

$$\dim \text{Kernel}(A - I) = \log_q |\text{Kernel}(A - I)| = \log_q |\text{Fix}(\mathcal{F})|;$$

hence,

$$\begin{aligned} g_{\text{linear}}(D, q) &= \max_{\mathcal{F} \text{ is linear}} \log_q |\text{Fix}(\mathcal{F})| \\ &= \max_{A \leq A_D} \dim \text{Kernel}(A - I) \\ &= \max_{A \leq A_D} \{n - \text{rank}(A - I)\} \\ &= n - \min_{A \leq A_D} \text{rank}(A - I) \\ &= |V| - \text{minrank}(D, q), \end{aligned}$$

where the last equality follows from lemma 1.3.15. □

Corollary 4.3.4. *Let $D = (V, E)$ be a digraph, and let $\mathbb{F}_q$ be a finite field. Then*

$$l_{\text{linear}}(D, q) + g_{\text{linear}}(D, q) = |V|.$$

4.4 Basic Results

In this section, we present some basic results about guessing games. These results are from [7].

Proposition 4.4.1. *For any digraph $D = (V, E)$,*

$$0 \leq g(D, s) \leq |V| - 1.$$

Proof. If $\mathcal{C}$ is a constant strategy (example 4.2.5), then

$$0 = g(D, s, \mathcal{C}) \leq g(D, s).$$

Let $\mathcal{F} = (f_v)_{v \in V}$ be an optimal strategy, and let $w \in V$ be a fixed vertex. If $x \in \mathrm{Fix}(\mathcal{F})$, we define $T_x := \{x' \in \Omega : x'_v = x_v \text{ if } v \neq w, \text{ and } x'_w \neq x_w\}$; it is clear that $|T_x| = s - 1$. If $x' \in \mathrm{Fix}(\mathcal{F}) \cap T_x$, then $f_w(x'_{N(w)}) = x'_w$ (since $x' \in \mathrm{Fix}(\mathcal{F})$) and $x_{N(w)} = x'_{N(w)}$ (since $x' \in T_x$); but since $x \in \mathrm{Fix}(\mathcal{F})$, we conclude that $x_w = f_w(x_{N(w)}) = f_w(x'_{N(w)}) = x'_w$, a contradiction. So, $T_x \subseteq \Omega - \mathrm{Fix}(\mathcal{F})$ for all $x \in \mathrm{Fix}(\mathcal{F})$, and therefore

$$\bigcup_{x \in \mathrm{Fix}(\mathcal{F})} T_x \subseteq \Omega - \mathrm{Fix}(\mathcal{F}).$$

Now, for $x, y \in \mathrm{Fix}(\mathcal{F})$, we prove that $T_x \cap T_y = \emptyset$ whenever $x \neq y$. Let $z \in T_x \cap T_y$. We have

$$z_v = x_v = y_v \text{ if } v \neq w. \tag{1}$$

Hence, $x_{N(w)} = y_{N(w)}$. Since $x, y \in \mathrm{Fix}(\mathcal{F})$, then

$$x_w = f_w(x_{N(w)}) = f_w(y_{N(w)}) = y_w. \tag{2}$$

From (1) and (2), it follows that $x = y$.

Hence,

$$\begin{aligned}
(s-1)|\,\mathrm{Fix}(\mathcal{F})| &\leq |\Omega - \mathrm{Fix}(\mathcal{F})| \\
s|\,\mathrm{Fix}(\mathcal{F})| &\leq |\Omega - \mathrm{Fix}(\mathcal{F})| + |\,\mathrm{Fix}(\mathcal{F})| \\
s|\,\mathrm{Fix}(\mathcal{F})| &\leq |\Omega| \\
s|\,\mathrm{Fix}(\mathcal{F})| &\leq s^{|V|} \\
|\,\mathrm{Fix}(\mathcal{F})| &\leq s^{|V|-1}.
\end{aligned}$$

We conclude that $g(D, s) = \log_s |\,\mathrm{Fix}(\mathcal{F})| \leq |V| - 1$. □

Corollary 4.4.2. *Let $D = (V, E)$ be a digraph. If $q = p^m$ for some prime p and positive integer m, then*

$$0 \leq g_{\mathrm{linear}}(D, q) \leq g(D, q) \leq |V| - 1.$$

Proof. If $\mathcal{C}$ is a constant strategy (and therefore linear), then

$$0 = g(D, q, \mathcal{C}) \leq g_{\text{linear}}(D, q).$$

The rest is obvious. □

We want to find a relation between the guessing numbers (over an alphabet of s elements) of H and D if H is a subdigraph of D. Without loss of generality, suppose that the alphabet is $\mathbb{Z}_s$. For that purpose, we want to extend a strategy $\mathcal{F} = (f_v)_{v \in V(H)} \in \mathbb{S}(H, \mathbb{Z}_s)$ to a strategy $\bar{\mathcal{F}} \in \mathbb{S}(D, \mathbb{Z}_s)$. Let $x \in \Omega(D, \mathbb{Z}_s)$. We define $\bar{\mathcal{F}} = (\bar{f}_v)_{v \in V(D)}$ by $\bar{f}_v(x_{N_D(v)}) := f_v(x_{N_H(v)})$ if $v \in V(H)$, and $\bar{f}_v(x_{N_D(v)}) := 0$ if $v \notin V(H)$.

Also, if $x \in \Omega(H, \mathbb{Z}_s)$, we can extend it to $\bar{x} \in \Omega(D, \mathbb{Z}_s)$, where $\bar{x}_v := x_v$ if $v \in V(H)$, and $\bar{x}_v := 0$ if $v \notin V(H)$.

Lemma 4.4.3. *If H is a subdigraph of D, then $g(H, s) \leq g(D, s)$.*

Proof. Let $\mathcal{F} \in \mathbb{S}(H, \mathbb{Z}_s)$ be an optimal strategy. If $x \in \text{Fix}(\mathcal{F})$, then $\bar{x} \in \text{Fix}(\bar{\mathcal{F}})$. Let $x, y \in \text{Fix}(\mathcal{F})$, since $x \neq y$ implies that $\bar{x} \neq \bar{y}$, we have

$$|\,\text{Fix}(\mathcal{F})| \leq |\,\text{Fix}(\bar{\mathcal{F}})|.$$

Therefore,

$$g(H, s) = g(H, s, \mathcal{F}) = \log_s |\,\text{Fix}(\mathcal{F})| \leq \log_s |\,\text{Fix}(\bar{\mathcal{F}})| = g(D, s, \bar{\mathcal{F}}) \leq g(D, s).$$

□

Lemma 4.4.4. *If C is a directed cycle, then $g(C, s) \geq 1$.*

Proof. Take the strategy $\mathcal{F} = (id)_{v \in V(C)}$. It is clear that

$$\text{Fix}(\mathcal{F}) = \{x \in \Omega : \exists c \in \mathcal{A} \text{ such that } x_v = c, \forall v \in V(C)\};$$

therefore, $|\,\text{Fix}(\mathcal{F})| = s$. So, $1 = \log_s |\,\text{Fix}(\mathcal{F})| = g(C, s, \mathcal{F}) \leq g(C, s)$. □

We prove later that $g(C, s) = 1$ if C is a directed cycle.

Lemma 4.4.5. *Let D be a digraph, then $g(D, s) = 0$ iff D is acyclic.*

Proof. $\Rightarrow$) If D is cyclic, there is a subdigraph C such that C is a directed cycle. Using the last two lemmas, $1 \leq g(C, s) \leq g(D, s)$ and therefore $g(D, s) \neq 0$.

$\Leftarrow$) If D is acyclic, without loss of generality, we can suppose that $V(D) = \{1, 2, \ldots, n\}$; and if $(i, j) \in E(D)$, then $i < j$.

Let $\mathcal{F} = (f_i)_{i\in[n]}$ be a strategy and suppose that $x = (x_i)_{i\in[n]} \in \operatorname{Fix}(\mathcal{F})$. Since D is acyclic, $f_1 \equiv c$ is constant ($N(1) = \emptyset$) and therefore x_1 is uniquely determined ($x_1 = c$). Now, $N(2) = \emptyset$ (f_2 is constant) or $N(2) = \{1\}$ ($x_2 = f_2(x_1)$); in both cases, x_2 is uniquely determined since x_1 is uniquely determined. In general, we can determine x_k uniquely after determining $x_1, \ldots, x_{k-1}$. So, if $\operatorname{Fix}(\mathcal{F}) \neq \emptyset$, then $|\operatorname{Fix}(\mathcal{F})| = 1$. Thus, $g(D, s) \leq 0$. By proposition 4.4.1, we have $g(D, s) = 0$. □

If $\mathcal{F} = (f_v)_{v\in V(D)}, \mathcal{F}' = (f'_v)_{v\in V(D)} \in \mathbb{S}(D, \mathbb{Z}_s)$, we define

$$\mathcal{F} + \mathcal{F}' := (f_v + f'_v)_{v\in V(D)} \in \mathbb{S}(D, \mathbb{Z}_s)$$

by $(f_v + f_{v'})(x_{N(v)}) := f_v(x_{N(v)}) + f_{v'}(x_{N(v)})$, where the last sum is over $\mathbb{Z}_s$.

Lemma 4.4.6. *For two vertex-disjoint digraphs H_1 and H_2, we have*

$$g(H_1 \cup H_2, s) = g(H_1, s) + g(H_2, s).$$

Proof. Without loss of generality, we suppose that the alphabet is $\mathbb{Z}_s$. Since H_1 and H_2 are subdigraphs of the disjoint union $D = H_1 \cup H_2$, we can use the notation introduced before lemma 4.4.3.

If $\mathcal{F}_1 \in \mathbb{S}(H_1, \mathbb{Z}_s)$ and $\mathcal{F}_2 \in \mathbb{S}(H_2, \mathbb{Z}_s)$, then $\mathcal{F} := \bar{\mathcal{F}}_1 + \bar{\mathcal{F}}_2 \in \mathbb{S}(H_1 \cup H_2, \mathbb{Z}_s)$. On the other hand, if $\mathcal{F} \in \mathbb{S}(H_1 \cup H_2, \mathbb{Z}_s)$, then there are unique $\mathcal{F}_1 \in \mathbb{S}(H_1, \mathbb{Z}_s)$ and $\mathcal{F}_2 \in \mathbb{S}(H_2, \mathbb{Z}_s)$ such that $\mathcal{F} = \bar{\mathcal{F}}_1 + \bar{\mathcal{F}}_2$. The same argument can be used for configurations.

Let $\mathcal{F} = \bar{\mathcal{F}}_1 + \bar{\mathcal{F}}_2$ and $x = \bar{x}_1 + \bar{x}_2$. It is clear that $x \in \operatorname{Fix}(\mathcal{F})$ iff $x_i \in \operatorname{Fix}(\mathcal{F}_i)$ for $i = 1, 2$, and therefore

$$|\operatorname{Fix}(\mathcal{F})| = |\operatorname{Fix}(\mathcal{F}_1)||\operatorname{Fix}(\mathcal{F}_2)|.$$

So,

$$\begin{aligned} g(H_1 \cup H_2, s, \mathcal{F}) &= \log_s |\operatorname{Fix}(\mathcal{F})| \\ &= \log_s |\operatorname{Fix}(\mathcal{F}_1)| + \log_s |\operatorname{Fix}(\mathcal{F}_2)| \\ &= g(H_1, s, \mathcal{F}_1) + g(H_2, s, \mathcal{F}_2). \end{aligned}$$

If $\mathcal{F} = \bar{\mathcal{F}}_1 + \bar{\mathcal{F}}_2 \in \mathbb{S}(H_1 \cup H_2, \mathbb{Z}_s)$ is an optimal strategy, then

$$\begin{aligned} g(H_1 \cup H_2, s) &= g(H_1 \cup H_2, s, \mathcal{F}) \\ &= g(H_1, s, \mathcal{F}_1) + g(H_2, s, \mathcal{F}_2) \\ &\leq g(H_1, s) + g(H_2, s). \end{aligned}$$

On the other hand, if $\mathcal{F}_1 \in \mathbb{S}(H_1, \mathbb{Z}_s)$ and $\mathcal{F}_2 \in \mathbb{S}(H_2, \mathbb{Z}_s)$ are optimal strategies, then for $\mathcal{F} = \bar{\mathcal{F}}_1 + \bar{\mathcal{F}}_2$, we have

$$\begin{aligned} g(H_1, s) + g(H_2, s) &= g(H_1, s, \mathcal{F}_1) + g(H_2, s, \mathcal{F}_2) \\ &= g(H_1 \cup H_2, s, \mathcal{F}) \\ &\leq g(H_1 \cup H_2, s). \end{aligned}$$

In conclusion, $g(H_1 \cup H_2, s) = g(H_1, s) + g(H_2, s)$. □

Lemma 4.4.7. *Let H_1 and H_2 be two vertex-disjoint digraphs. If $q = p^m$ for some prime p and positive integer m, then*

$$g_{\text{linear}}(H_1 \cup H_2, q) = g_{\text{linear}}(H_1, q) + g_{\text{linear}}(H_2, q).$$

Proof. Similar to the proof of the above lemma. □

Lemma 4.4.8. *Let D be a digraph. If H is an induced subdigraph of D, then $g(D, s) \leq g(H, s) + |V(D) - V(H)|$.*

Proof. i) If $|V(D) - V(H)| = 0$, then $D = H$ and $g(D, s) = g(H, s)$.

ii) If $|V(D) - V(H)| = 1$, let $\mathcal{F} = (f_v)_{v \in V(D)} \in \mathbb{S}(D, \mathcal{A})$, where $\mathcal{A}$ is an alphabet with s elements, and let w be the only vertex that belongs to $V(D) - V(H)$.

For $a \in \mathcal{A}$, we define the strategy $\mathcal{F}^a = (f_v^a)_{v \in V(H)} \in \mathbb{S}(H, \mathcal{A})$ as follows: for all $v \in V(H)$ such that $w \notin N_D(v)$, $f_v^a := f_v$; otherwise, for every configuration $x \in \Omega(H, \mathcal{A})$, $f_v^a(x_{N_H(v)}) := f_v(\hat{x}_{N_D(v)})$, where $\hat{x} \in \Omega(D, \mathcal{A})$ is the configuration defined by $\hat{x}_w := a$ and $\hat{x}_{V(H)} := x$.

We define $\operatorname{Fix}_a(\mathcal{F}) := \{x \in \operatorname{Fix}(\mathcal{F}) : x_w = a\}$, so $\operatorname{Fix}(\mathcal{F}) = \bigcup_{a \in \mathcal{A}} \operatorname{Fix}_a(\mathcal{F})$, where the union it is actually a disjoint union. Clearly, $x, y \in \operatorname{Fix}_a(\mathcal{F})$ with $x \neq y$ implies $x_H, y_H \in \operatorname{Fix}(\mathcal{F}^a)$ with $x_H \neq y_H$. We conclude that

$$|\operatorname{Fix}(F)| = \sum_{a \in \mathcal{A}} |\operatorname{Fix}_a(\mathcal{F})| \leq \sum_{a \in \mathcal{A}} |\operatorname{Fix}(\mathcal{F}^a)|.$$

If $\mathcal{F} \in \mathbb{S}(D, \mathcal{A})$ and $\mathcal{F}' \in \mathbb{S}(H, \mathcal{A})$ are optimal strategies, then

$$\begin{aligned} g(D, s) &= g(D, s, \mathcal{F}) \\ &= \log_s |\operatorname{Fix}(\mathcal{F})| \\ &\leq \log_s(\sum_{a \in \mathcal{A}} |\operatorname{Fix}(\mathcal{F}^a)|) \\ &\leq \log_s(s|\operatorname{Fix}(\mathcal{F}')|) \\ &= 1 + \log_s |\operatorname{Fix}(\mathcal{F}')| \\ &= 1 + g(H, s). \end{aligned}$$

iii) For the general case, $V(D) - V(H) = \{v_1, \ldots, v_n\}$, we define $V_i := V(H) \cup \{v_1, \ldots, v_i\}$. Let H_i be the subdigraph of D induced by V_i. Applying ii) consecutively, we have

$$\begin{aligned} g(D,s) &\leq g(H_{n-1},s) + 1 \leq g(H_{n-2},s) + 2 \leq \ldots \\ &\leq g(H_1) + n - 1 \leq g(H) + |V(D) - V(H)|. \end{aligned}$$

□

Proposition 4.4.9. *For any digraph D, $\nu(D) \leq g(D,s) \leq \tau(D)$.*

Proof. If $\nu(D) = n$, then there are n disjoint cycles of D, $C_1, \ldots, C_n$. We have

$$\nu(D) \leq g(C_1,s) + \ldots + g(C_n,s) = g(C_1 \cup \cdots \cup C_n, s) \leq g(D,s).$$

If $\tau(D) = m$, then there is a subset S of $V(D)$, of size m, such that $D - S = D[V(D) - S]$ is acyclic. So,

$$g(D) \leq g(D-S,s) + |V(D) - V(D-S)| = 0 + m = m.$$

□

Corollary 4.4.10. *Let D be a digraph. If $q = p^m$ for some prime p and positive integer m, then $\nu(D) \leq g_{\text{linear}}(D,q) \leq g(D,q) \leq \tau(D)$.*

Corollary 4.4.11. *If C is a directed cycle, then $g(C,s) = 1$.*

Proof.

$$1 = \nu(C) \leq g(C,s) \leq \tau(C) = 1.$$

□

4.5 The Relation between Network Coding and Guessing Games

In [5], guessing games are introduced and also a connection between guessing games and network coding is presented. We end this monograph presenting this connection. General speaking, the idea is to associate a digraph $D_{\mathcal{N}}$ to each multiple-unicast network $\mathcal{N}$ such that the problem of finding optimal strategies of $D_{\mathcal{N}}$ over the alphabet $\mathcal{A}$ is equivalent to the problem of finding solutions of $\mathcal{N}$ over $\mathcal{A}$. This connection, together with the ones in Theorem 2.3.2, Theorem 3.3.5, and Theorem 4.3.3, give us a good appreciation on how Network Coding, Index Coding, and Guessing Games are connected.

The results presented in this section are from [5]. We follow the general idea in [5], but we introduce some additional notation we think makes the construction of $D_{\mathcal{N}}$ and the proofs easier to understand.

Let $s \in S(\mathcal{N})$ and $r \in R(\mathcal{N})$, we use the notation $s \to r$ if there are edges $e_1, \ldots, e_m \in E(\mathcal{N}) - C(\mathcal{N})$ such that $s = \text{tail}(e_1)$, $\text{head}(e_1) = \text{tail}(e_2)$,..., $\text{head}(e_{m-1}) = \text{tail}(e_m)$, $\text{head}(e_m) = r$.

Let $s \in S(\mathcal{N})$ and $e \in C(\mathcal{N})$, we use the notation $s \to e$ if there are edges $e_1, \ldots, e_m \in E(\mathcal{N}) - C(\mathcal{N})$ such that $s = \text{tail}(e_1)$, $\text{head}(e_1) = \text{tail}(e_2)$,..., $\text{head}(e_{m-1}) = \text{tail}(e_m)$, $\text{head}(e_m) = \text{tail}(e)$.

Let $e, e' \in C(\mathcal{N})$, $e \neq e'$, we use the notation $e \to e'$ if $\text{head}(e) = \text{tail}(e')$ or there are edges $e_1, \ldots, e_m \in E(\mathcal{N}) - C(\mathcal{N})$ such that $\text{head}(e) = \text{tail}(e_1)$, $\text{head}(e_1) = \text{tail}(e_2)$,..., $\text{head}(e_{m-1}) = \text{tail}(e_m)$, $\text{head}(e_m) = \text{tail}(e')$.

Finally, let $e \in C(\mathcal{N})$ and $r \in R(\mathcal{N})$, we use the notation $e \to r$ if $\text{head}(e) = r$ or there are edges $e_1, \ldots, e_m \in E(\mathcal{N}) - C(\mathcal{N})$ such that $\text{head}(e) = \text{tail}(e_1)$, $\text{head}(e_1) = \text{tail}(e_2)$,..., $\text{head}(e_{m-1}) = \text{tail}(e_m)$, $\text{head}(e_m) = r$.

Roughly speaking, if $a \to b$, the information in a is the same as the information in b, since the information travels from a to b trough edges of $E(\mathcal{N}) - C(\mathcal{N})$, that is, edges where codification is not required.

Notice that this notation can be used in any network, not necessarily a multiple-unicast network. We can put the additional condition in definition 2.2.3, that we never have $\tau(r) \to r$ for some receiver r. The reason is that if $\tau(r) \to r$, there are $e_1, \ldots, e_m \in E(\mathcal{N}) - C(\mathcal{N})$ such that $\tau(r) = \text{tail}(e_1), \text{head}(e_1) = \text{tail}(e_2), \ldots, \text{head}(e_{m-1}) = \text{tail}(e_m), \text{head}(e_m) = r$. We can send the message generated by $\tau(r)$ through these edges, and the receiver r will always be able to receive the message it needs.

From now on, we suppose that every network $\mathcal{N}$ satisfies the above condition. This condition is required in order to guarantee that $D_{\mathcal{N}}$ has no loops (edges whose tail and head are the same).

We are now in position to construct $D_{\mathcal{N}}$. Let $\mathcal{N}$ be a multiple-unicast network, where $S(\mathcal{N}) = \{s_1, \ldots, s_n\}$, $R(\mathcal{N}) = \{r_1, \ldots, r_n\}$, and $\tau(r_i) = s_i$, for all $i \in [n]$. The digraph $D_{\mathcal{N}}$ is defined by $V(D_{\mathcal{N}}) := C(\mathcal{N}) \cup R(\mathcal{N})$, and the edges of $D_{\mathcal{N}}$ are constructed as follows:

$\star$ If $s_i \to r_j$, $s_i \in S(\mathcal{N})$, $r_j \in R(\mathcal{N})$, then $(r_i, r_j) \in E(D_{\mathcal{N}})$.

⋆ If $s_i \to e$, $s_i \in S(\mathcal{N})$, $e \in C(\mathcal{N})$, then $(r_i, e) \in E(D_{\mathcal{N}})$.

⋆ If $e \to e'$, $e, e' \in C(\mathcal{N})$, $e \neq e'$, then $(e, e') \in E(D_{\mathcal{N}})$.

⋆ If $e \to r_i$, $e \in C(\mathcal{N})$, $r_i \in R(\mathcal{N})$, then $(e, r_i) \in E(D_{\mathcal{N}})$.

As an example, consider the butterfly network (Figure 6). Here, $\tau(r_1) = s_1$ and $\tau(r_2) = s_2$.

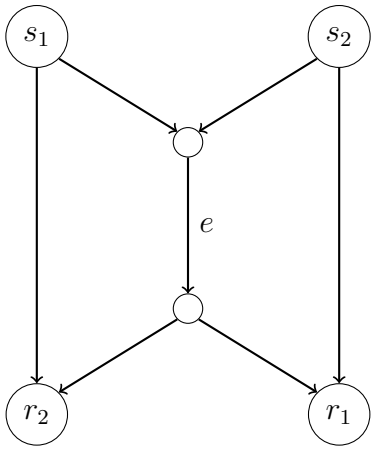

Figure 6

We see that $s_1 \to r_2, s_1 \to e, s_2 \to e, s_2 \to r_1, e \to r_2, e \to r_1$. Then, $V(D_{\mathcal{N}}) = \{e, r_1, r_2\}$ and $E(D_{\mathcal{N}}) = \{(r_1, r_2), (r_1, e), (r_2, e), (r_2, r_1), (e, r_2), (e, r_1)\}$. Figure 7 depicts $D_{\mathcal{N}}$, which may be treated as K_3.

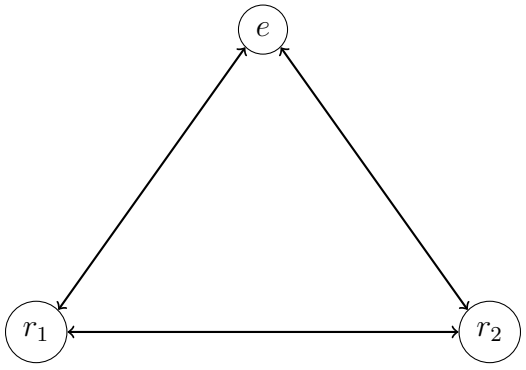

Figure 7

Lemma 4.5.1. *Let $\mathcal{N} = (D, \tau)$ be a multiple-unicast network. Then*

$$g(D_{\mathcal{N}}[C(\mathcal{N})], s) = 0.$$

Proof. We know that (e, e') is and edge of $D_{\mathcal{N}}[C(\mathcal{N})]$ iff $e \to e'$.

Suppose that $D_{\mathcal{N}}[C(\mathcal{N})]$ is not acyclic, that is, there are $e_1, \ldots, e_k \in C(\mathcal{N})$ such that $(e_1, e_2), (e_2, e_3), \ldots, (e_k, e_1)$ are edges of $D_{\mathcal{N}}[C(\mathcal{N})]$. Therefore, $e_1 \to e_2, e_2 \to e_3, \ldots, e_k \to e_1$. This implies that D is not acyclic. This is a contradiction since D is acyclic by the same definition of a network; therefore, $D_{\mathcal{N}}[C(\mathcal{N})]$ is acyclic. Using lemma 4.4.5, we have $g(D_{\mathcal{N}}[C(\mathcal{N})], s) = 0$. □

Lemma 4.5.2. *Let $\mathcal{N}$ be a multiple-unicast network with n receivers. Then,*

$$g(D_{\mathcal{N}}, s) \leq n.$$

Proof. Using lemma 4.4.8 and the above lemma,

$$g(D_{\mathcal{N}}, s) \leq [g(D_{\mathcal{N}}[C(\mathcal{N})], s) + |V(D_{\mathcal{N}}) - C(\mathcal{N})| = 0 + n = n.$$

□

The idea is to associate strategies in $D_{\mathcal{N}}$ to network codes in $\mathcal{N}$ such that optimal strategies correspond to solutions. We notice that in both cases (strategies in $D_{\mathcal{N}}$ and network codes in $\mathcal{N}$) we assign a function f_v to each element in $C(\mathcal{N}) \cup R(\mathcal{N})$. The difference is the domain where this function is defined. In the first case, the domain is $\mathcal{A}^{N(v)}$. In the latter case, the domain is $\mathcal{A}^{\text{In}(v)}$. With this in mind, the first thing we need to do is to find a bijection from $N(v)$ onto $\text{In}(v)$.

Let $\mathcal{N} = (D, \tau)$ be a multiple-unicast network. Since we are working with two digraphs at the same time, $D_{\mathcal{N}}$ and D, it is important to distinguish which digraph we are talking about when we use notation like $N(v)$ or $\text{In}(v)$, where $v \in C(\mathcal{N}) \cup R(\mathcal{N})$. When we use the notation $N(v)$, we are referring to $D_{\mathcal{N}}$. On the other hand, when we use the notation $\text{In}(v)$, we are referring to D.

Let $e \in C(\mathcal{N})$. We define a bijection $g_e : N(e) \longrightarrow \text{In}(e)$ in the following way:

- ⋆ If $e' \in N(e), e' \in C(\mathcal{N})$, then $\text{head}(e') = \text{tail}(e)$ or there are $e_1, \ldots, e_m \in E(\mathcal{N}) \cup C(\mathcal{N})$ such that $\text{head}(e') = \text{tail}(e_1)$,...,$\text{head}(e_m) = \text{tail}(e)$. In the first case, $g_e(e') := e'$. In the latter case, $g_e(e') := e_m$.
- ⋆ If $r \in N(e), r \in R(\mathcal{N})$, then $\tau(r) \to e$, that is, there are $e_1, \ldots, e_m \in E(\mathcal{N}) - C(\mathcal{N})$ such that $\tau(r) = \text{tail}(e_1), \ldots, \text{head}(e_m) = \text{tail}(e)$. In this case, $g_e(r) := e_m$.

Let $r \in R(\mathcal{N})$. We define a bijection $g_r : N(r) \longrightarrow \mathrm{In}(r)$ in the following way:

- ⋆ If $e \in N(r), e \in C(\mathcal{N})$, then $\mathrm{head}(e) = r$ or there are $e_1, \ldots, e_m \in E(\mathcal{N}) - C(\mathcal{N})$ such that $\mathrm{head}(e) = \mathrm{tail}(e_1), \ldots, \mathrm{head}(e_m) = r$. In the first case, $g_r(e) := e$. In the latter case, $g_r(e) := e_m$.
- ⋆ If $r' \in N(r), r' \in R(\mathcal{N})$, then $\tau(r') \to r$, that is, there are $e_1, \ldots, e_m \in E(\mathcal{N}) - C(\mathcal{N})$ such that $\tau(r') = \mathrm{tail}(e_1) \ldots, \mathrm{head}(e_m) = r$. In this case, $g_r(r') := e_m$.

Using the above functions, we can now find a bijection $h_v : \mathcal{A}^{\mathrm{In}(v)} \longrightarrow \mathcal{A}^{N(v)}$, for each $v \in C(\mathcal{N}) \cup R(\mathcal{N})$. The function h_v is defined, for each $x \in \mathcal{A}^{\mathrm{In}(v)}$, by

$$\begin{aligned} h_v(x) : N(v) &\longrightarrow \mathcal{A} \\ w &\longmapsto x(g_v(w)). \end{aligned}$$

According to this, there is a correspondence between the strategy $(f_v)_{v \in C(\mathcal{N}) \cup R(\mathcal{N})}$ and the network code $(f_v \circ h_v)_{v \in C(\mathcal{N}) \cup R(\mathcal{N})}$. Equivalently, there is a correspondence between the network code $(f_v)_{v \in C(\mathcal{N}) \cup R(\mathcal{N})}$ and the strategy $(f_v \circ h_v^{-1})_{v \in C(\mathcal{N}) \cup R(\mathcal{N})}$.

Theorem 4.5.3. *Let $\mathcal{N} = (D, \tau)$ be a multiple-unicast network with n receivers, and let $\mathcal{A}$ be an alphabet with s elements.*

The strategy $(f_v)_{v \in C(\mathcal{N}) \cup R(\mathcal{N})}$ of $D_{\mathcal{N}}$ over $\mathcal{A}$ is optimal iff the network code $(f_v \circ h_v)_{v \in C(\mathcal{N}) \cup R(\mathcal{N})}$ of $\mathcal{N}$ over $\mathcal{A}$ is a solution. Equivalently, the network code $(f_v)_{v \in C(\mathcal{N}) \cup R(\mathcal{N})}$ of $\mathcal{N}$ over $\mathcal{A}$ is a solution iff the strategy $(f_v \circ h_v^{-1})_{v \in C(\mathcal{N}) \cup R(\mathcal{N})}$ of $D_{\mathcal{N}}$ over $\mathcal{A}$ is optimal.

Moreover, $\mathcal{N}$ is solvable over $\mathcal{A}$ iff $g(D_{\mathcal{N}}, s) = n$.

Proof. Let $\mathcal{F} = (f_v)_{v \in C(\mathcal{N}) \cup R(\mathcal{N})}$ be a network code of $\mathcal{N}$ over $\mathcal{A}$, and consider its associated strategy $\mathcal{H} = (f_v \circ h_v^{-1})_{v \in C(\mathcal{N}) \cup R(\mathcal{N})}$ of $D_{\mathcal{N}}$ over $\mathcal{A}$.

Let $\omega \in \mathcal{A}^{S(\mathcal{N})}$ be a message assignment. The configuration $\tilde{\omega} \in \mathcal{A}^{C(\mathcal{N}) \cup R(\mathcal{N})}$ is defined by $\tilde{\omega}_r := \omega_{\tau(r)}$ if $r \in R(\mathcal{N})$, and $\tilde{\omega}_e := \bar{e}(\omega)$ if $e \in C(\mathcal{N})$.

We have the following equivalences:

$$\begin{aligned} \mathcal{F} \text{ is a solution of } \mathcal{N} &\Longleftrightarrow \tilde{\omega} \in \mathrm{Fix}(\mathcal{H}), \forall \omega \in \mathcal{A}^{S(\mathcal{N})} \\ &\Longleftrightarrow s^n \leq |\mathrm{Fix}(\mathcal{H})| \\ &\Longleftrightarrow n \leq g(D_{\mathcal{N}}, s, \mathcal{H}) \leq g(D_{\mathcal{N}}, s) \\ &\Longleftrightarrow n = g(D_{\mathcal{N}}, s, \mathcal{H}) = g(D_{\mathcal{N}}, s) \text{ (lemma 4.5.2)} \\ &\Longleftrightarrow \mathcal{H} \text{ is a optimal strategy of } D_{\mathcal{N}} \text{ over } \mathcal{A} \\ &\qquad \text{and } g(D_{\mathcal{N}}, s) = n. \end{aligned}$$

From the preceding implications, the only non-trivial is

$$s^n \leq |\operatorname{Fix}(\mathcal{H})| \Longrightarrow \tilde{\omega} \in \operatorname{Fix}(\mathcal{H}), \forall \omega \in \mathcal{A}^{S(\mathcal{N})}.$$

We now prove it. Let $\omega \in \mathcal{A}^{S(\mathcal{N})}$. We define the strategy $\mathcal{H}^\omega := (f_e^\omega)_{e \in C(\mathcal{N})}$ of $D_{\mathcal{N}}[C(\mathcal{N})]$ over $\mathcal{A}$ as follows: If $x \in \mathcal{A}^{C(\mathcal{N})}$, then $f_e^\omega(x) := (f_e \circ h_e^{-1})(\hat{x})$ where $\hat{x}_{C(\mathcal{N})} := x$ and $\hat{x}_r := \omega_{\tau(r)}$, for all $r \in R(\mathcal{N})$.

It is clear that the configuration $z = (z_e)_{e \in C(\mathcal{N})} \in \mathcal{A}^{C(\mathcal{N})}$, where $z_e := \bar{e}(\omega)$, $e \in C(\mathcal{N})$, belongs to $\operatorname{Fix}(\mathcal{H}^\omega)$. Therefore, $1 \leq |\operatorname{Fix}(\mathcal{H}^\omega)|$. On the other hand, $|\operatorname{Fix}(\mathcal{H}^\omega)| \leq 1$ since $g(D_{\mathcal{N}}[C(\mathcal{N})], s) = 0$ (lemma 4.5.1). Thus, $|\operatorname{Fix}(\mathcal{H}^\omega)| = \{z\}$.

If there exists $y \in \operatorname{Fix}(\mathcal{H})$ such that $y_r = \omega_{\tau(r)}$, for all $r \in R(N)$, then it is clear that $y_{C(\mathcal{N})} \in \operatorname{Fix}(\mathcal{H}^\omega)$; hence, $y_{C(\mathcal{N})} = z$, that is, $y_e = \bar{e}(\omega)$, for all $e \in C(\mathcal{N})$. In conclusion, $y = \tilde{\omega}$.

Since $|S(\mathcal{N})| = |R(\mathcal{N})| = n$ ($\mathcal{N}$ is a multiple-unicast network), there is a bijection between $\mathcal{A}^{S(\mathcal{N})}$ and $\mathcal{A}^{R(\mathcal{N})}$. This implies that for each $y \in \operatorname{Fix}(\mathcal{H})$, there is a unique $\omega \in \mathcal{A}^{S(\mathcal{N})}$ such that $y_r = \omega_{\tau(r)}$, for all $r \in R(\mathcal{N})$, and therefore $y = \tilde{\omega}$. We have proved that $\operatorname{Fix}(\mathcal{H}) \subseteq \{\tilde{\omega} : \omega \in \mathcal{A}^{S(\mathcal{N})}\}$.

It is clear that each $\omega \in \mathcal{A}^{S(\mathcal{N})}$ determines a unique $\tilde{\omega}$. Hence,

$$s^n \leq |\operatorname{Fix}(\mathcal{H})| \leq |\{\tilde{\omega} : \omega \in \mathcal{A}^{S(\mathcal{N})}\}| = |\mathcal{A}^{S(\mathcal{N})}| = s^n.$$

Therefore, $\operatorname{Fix}(\mathcal{H}) = \{\tilde{\omega} : \omega \in \mathcal{A}^{S(\mathcal{N})}\}$. This completes the proof. □

Corollary 4.5.4. *Let $\mathcal{N}$ be a multiple-unicast network with n receivers, and let $\mathbb{F}_q$ be a finite field. The following statements are equivalent:*

i) *$\mathcal{N}$ is linearly solvable over $\mathbb{F}_q$.*

ii) $g_{\text{linear}}(D_{\mathcal{N}}, q) = n$.

iii) $l_{\text{linear}}(D_{\mathcal{N}}, q) = |C(\mathcal{N})|$.

iv) $\operatorname{minrank}(D_{\mathcal{N}}, q) = |C(\mathcal{N})|$.

Proof. We noted earlier that there is a correspondence between network codes and strategies. If $\mathcal{H}$ is the strategy associated to the network code $\mathcal{F}$, it is clear that $\mathcal{F}$ is linear iff $\mathcal{H}$ is linear. From this observation and the last theorem, equivalence i)⇔ii) follows. The other equivalences are consequences of Theorem 3.3.5 and Theorem 4.3.3. □

Conclusions

In this monograph, we have explored some interesting relations among Network Coding, Index Coding, and Guessing games, but we have focused on the linear case. This is just an invitation to explore the relations in the general case. Even in the linear case, there are more things to say, for example, what happens if the alphabet is a finite ring instead of a finite field?

Apart from the adjacent matrix, other matrices are associated to digraphs and graphs; since linear algebra was used extensively in this monograph, this raises the question of how much information can we get from these matrices that helps us to calculate parameters like the guessing number $g(D, s)$, and the minimum length of an index code $l(D, s)$.

Further studies may include the analysis of particular classes of digraphs and graphs, and the presentation of more examples with explicit calculations of $g(D, s)$ and $l(D, s)$. This may include examples of digraphs such that $g(D, s)$ and $l(D, s)$ change when s changes. The design of algorithms and the use of Information Theory to calculate these parameters are also of special interest.

Also, there are many other well-studied digraph parameters, especially for graphs, that may tell us something about the topics treated here.

Bibliography

[1] R. Ahlswede, N. Cai, S.-Y. R. Li and R. W. Yeung, *Network information flow*. IEEE Trans. Inform. Theory, vol. 46, pp. 1204-1216, July 2000.

[2] Z. Bar-Yossef, Y. Birk, T.S. Jayram and T. Kol. *Index coding with side information*. Proc. of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), pp. 197-206.

[3] Y. Birk and T. Kol. *Coding-On-Demand by an Informed Source (ISCOD) for Efficient Broadcast of Different Supplemental Data to Caching Clients*. IEEE Transactions on Information Theory, vol. 52, no. 6, pp. 2825-2830, 2006, early version appeared in INFOCOM ′98.

[4] R. Dougherty and K. Zeger. *Nonreversibility and Equivalent Constructions of Multiple-Unicast Networks*. IEEE Transactions on Information Theory vol. 52, no. 11, pp. 5067-5077, November 2006.

[5] S. Riis. *Utilising public information in Network Coding* in General Theory of Information Transfer and Combinatorics, pp. 866-897, 2006.

[6] Steven Roman. *Advanced Linear Algebra*. Second Edition. Springer. 2005.

[7] T. Wu, P. Cameron, S. Riis. *On the guessing number of shift graphs*. J. Discrete Algorithms 7, pp. 220-226, 2009.