Office of Graduate Studies

Dissertation / Thesis Approval Form

This form is for use by all doctoral and master's students with a dissertation/thesis requirement. Please print clearly as the library will bind a copy of this form with each copy of the dissertation/thesis. All doctoral dissertations must conform to university format requirements, which is the responsibility of the student and supervising professor. Students should obtain a copy of the Thesis Manual located on the library website.

Dissertation/Thesis Title: Algorithms for Computing Limits on Information Flow and Storage in Networks

Author: Jayant Apte

This dissertation/thesis is hereby accepted and approved.

Signatures:

Examining Committee

Chair _____

Members _____

_____

_____

_____

_____

Academic Advisor _____

Department Head _____

Office of Graduate Studies • 3141 Chestnut St. • Randell Hall 240 • Philadelphia, PA 19104
Tel.: 215-895-0366 • **Fax**: 215-895-0495 • **Email**: graduate@drexel.edu • **Web**: www.drexel.edu/provost/graduatestudies

Algorithms for Computing Limits on Information Flow and Storage in Networks

A Thesis

Submitted to the Faculty

of

Drexel University

by

Jayant Apte

in partial fulfillment of the

requirements for the degree

of

Doctor of Philosophy

August 2016

Dedications

This thesis is dedicated to my parents, Sharad and Sunanda Apte.

Acknowledgments

I would like to thank my advisor, Dr. John MacLaren Walsh for his guidance and perseverance, and the thesis committee members, Dr. Weber, Dr. Kandasamy, Dr. Benson, Dr. Tian and, Dr. Betten, for being outstanding mentors. I am also grateful to my parents for their continued support. Finally, I am thankful to the past and present members of the Adaptive Signal Processing and Information Theory Research Group, Congduan Li, Yunshu Liu, Gwanmo Ku, Qi Chen, Solmaz Torabi, David Cinciruk, Mengke Hu and Jie Ren, for their friendship, collaborative mindset, and curiosity, that contributed to a great atmosphere for research and learning.

Table of Contents

List of Tables

List of Figures

Abstract

Algorithms for Computing Limits on Information Flow and Storage in Networks
Jayant Apte
Advisor: John MacLaren Walsh, Ph.D.

Multi-source Network Coding over Directed Acyclic Hypergraphs (HMSNC) is a general problem model which encompasses a variety of problems of practical interest, such as optimal information transfer across networks, efficient and reliable distributed storage of information, and the design of delay-mitigating codes for streaming. This thesis considers the design of computer programs for automatically proving achievability and converse theorems for rate regions of HMSNC instances. The traditional techniques for proving these theorems are based on human intuition, and are tedious at best. A computer-assisted theorem prover, albeit effective for small to moderate problem instances, is capable of rapidly producing thousands of proofs, which can then be analyzed to find patterns and make general statements about HMSNC, a problem which has proven difficult to solve in general.

The first part of this thesis considers the design of an achievability prover for the HMSNC problem. Determining achievability of a specified rate vector for a given HMSNC instance, requires one to know an algorithm for determining if there exists an almost entropic polymatroid satisfying certain constraints on its rank function. Finding such an algorithm is a fundamental open problem in information theory, also called the problem of characterizing the closure of the region of entropic vectors ($\overline{\Gamma^*_N}$, $N \in \mathbb{N}$). A special case of this very difficult problem, called the Constrained Linear Representability Problem (CLRP) is defined, in the form of two variants: existential and enumerative. Algorithms based on group theoretic techniques for combinatorial generation are then developed to solve the enumerative variant of CLRP, which allows one to compute network coding rate regions achievable with a specified class of linear codes. The same algorithms are naturally able to solve the existential variant of CLRP, by halting as soon as the first combinatorial object satisfying the specified constraints is found, which allows them to determine the achievability of a specified rate vector with a specified class of linear codes. Thus, the first part of the thesis settles

the problem of designing algorithms to compute achievable HMSNC rate regions, in the context of linear codes over a specified finite field.

The second part of this thesis considers the problem of automating the converse proofs associated with the HMSNC problem. The tradiional approach to the converse proofs involves invocation of information inequalities one after another, to imply a specific inequality bounding the rate region of a given HMSNC instance. The only existing automation technique, is for verifying a putative inequality via linear programming, finding a collection of information inequalities such that a putative inequality bounding the HMSNC rate region can be obtained as their conic combination. However, no techniques exist to find the said putative inequalities bounding the rate region. To begin with, this thesis considers the problem of computing the tightest collection of linear inequalities bounding the rate region of a HMSNC instance, implied by a given collection of linear information inequalities which is called the problem of computing Explicit Polyhedral Outer Bounds (EPOBs) on the HMSNC rate regions. An algorithm based on a polyhedral projection technique, the Convex Hull Method (CHM), that is able to compute EPOBs exactly using rational arithmetic by working directly in the projection space is proposed. A variant of this algorithm, called symCHM, capable of reducing the complexity of computing EPOBs by exploiting problem symmetry, is also developed.

The algorithms proposed in this thesis are implemented in form of two open-source software projects, the Information Theoretic Achievability Prover (ITAP) and the Information Theoretic Converse Prover (ITCP), which are both first of their kind tools. Computational experiments with a variety of interesting HMSNC instances are performed using ITAP and ITCP to gauge the practical performance of the algorithms developed. In addition to the HMSNC problem, the generality of the techniques developed in this thesis make them applicable to computer-assisted proofs in a plethora of related problems, viz. secret sharing, co-operative guessing games played on graphs, and quantum marginal scenarios.

Chapter 1: Introduction

Ahlswede et.al., in their seminal work *Network Information Flow* [ACLY00], showed that routing is insufficient for achieving optimum rate of information transfer across networks. They introduced a generalization of routing, called *network coding*. In network coding, intermediate nodes of the network are allowed to code, achieving the optimum rate of information transfer, in case of a single source multi-cast. The cut-set bound, which is analogous to the max-flow min-cut theorem, characterizes the optimum rate of information transfer in such scenarios, while *linear* network coding is sufficient to achieve the optimum. Much progress has been reported on the single multi-cast case since, including polynomial time deterministic and randomized network code constructions (see. [JSC$^+$05] and [HMK$^+$06]).

On the other hand, the case of multiple independent sources, with sinks demanding subsets of information available at the sources, called multi-source multi-sink network coding, is rife with a variety of curious phenomena, such as:

1. There exist network instances that require non-linear network codes to achieve the vector of optimum information transfer rates for the sources [DFZ05],
2. Cut-set bounds are loose [Yeu08],
3. There exist network instances for which determining the vector of optimum information transfer rates for the sources requires *new laws of information theory*, called the non-Shannon-type information inequalities [DFZ07],
4. The problem of determining the *scalar linear capacity* is NP-hard [LL04].

The tools used today by information theorists to prove bounds on multi-source multi-sink network coding are the same as those used since the dawn of multi-terminal information theory: in order to prove converse coding theorems, information inequalities are invoked one after another using

intuition, while intuition alone is used to construct the codes proving the achievability. This thesis considers the problem of designing finite terminating algorithms for determining and proving bounds on multi-source multi-sink network coding. These algorithms, capable of rapidly proving bounds on network coding rate regions, enable one, in turn, to pursue a computational agenda for approaching problems like network coding, which are difficult to solve in general. This involves solving small instances of these problems to build large and exhaustive databases of solved instances which can then be analyzed to find patterns and structure that allow one to make much more general statements about these problems [Con15, Cona, Conb]. All of these tasks would be impossible without the use of a computer due to sheer number of man hours required.

The main contributions of this thesis can be summarized as follows:

1. Algorithms for computer-assisted achievability proofs: these are algorithms based on group theoretic techniques for combinatorial generation, for constructing polyhedral inner bounds on network coding rate regions achievable with a specified class of linear codes. **While characterizing the exact rate regions is of fundamental interest, these algorithms provide the ability to guage the possibilities with a fixed class linear of codes. Furthermore, the same algorithms can be used for determining the existance of a linear code over a given finite field, achieving a specified rate vector, providing an alternative method to the algebraic formulation of Koetter and Medard [KM03], which solves the same problem using Gröbner basis computation [CLO07].**

2. Algorithms for computer-assisted converse proofs: these are algorithms based on polyhedral computation techniques, for constructing polyhedral outer bounds on network coding rate regions implied by a specified collection of information inequalities. **The process of obtaining putative inequalities bounding a rate region departs from the traditional trial and error methodology, and is reduced to a well-defined computational problem, in the same manner as the problem of verifying a putative inequality, which can be reduced to a linear feasibility problem. The approach proposed here is more general and systematic than the previous work of Tian [Tia14], which uses a series of**

linear programs to determine the boundary of a rate region in $\mathbb{R}^2$.

3. Two open-source software projects, ITAP [JJ15] (Information Theoretic Achievability Prover) and ITCP [JJ16] (Information Theoretic Converse Prover), that implement the algorithms proposed in this thesis. **Equipped with these tools, one can approach network coding and related problems from an experimental point of view.**

Central to the algorithms developed in this thesis is the region of entropic vectors, which we will now introduce, followed by a formal introduction to the network coding problem. Consider a collection of N discrete random variableint of vs $\boldsymbol{X}_N = (X_1, \ldots, X_N)$. The associated *entropy vector* is a $2^N - 1$-dimensional vector obtained by stacking the entropies of subsets $\boldsymbol{X}_A \triangleq (X_n | n \in A)$ of $\boldsymbol{X}_N$ into a vector $\mathbf{h} \triangleq (h(\boldsymbol{X}_A) \mid A \subseteq [N])$. By convention, $h(\emptyset) = 0$. Let $\mathcal{D}_N$ be the set of all joint probability mass functions for N discrete random variables. Then the *entropy function* is the map $\mathbf{h} : \mathcal{D}_N \to \mathbb{R}^{2^N-1}$ mapping a joint probability mass function to its entropy vector. Now consider the following problem:

- (E1) Given a vector $\mathbf{h}' \in \mathbb{R}^{2^N-1}$, determine whether $\mathbf{h}' = \mathbf{h}(p)$ for some $p \in \mathcal{D}_N$.

If the answer to E1 is 'yes', then there exists a joint probability mass function p s.t. $\mathbf{h}' = \mathbf{h}(p)$, and $\mathbf{h}'$ is said to be *entropic*. We can now define the region of entropic vectors Γ_N^* as

$$\Gamma_N^* \triangleq \{\mathbf{h} \in \mathbb{R}^{2^N-1} \mid \mathbf{h} \text{ is entropic } \} \tag{1.1}$$

Based on the above definition, problem (E1) can be seen to be equivalent to testing membership in Γ_N^*. The closure of Γ_N^* is a convex cone. Characterization of Γ_N^* and $\overline{\Gamma_N^*}$ is of central importance in network information theory as they determine all fundamental information inequalities [Yeu08], and the capacity regions of all networks under network coding [YYZ12, Conb]. Fujishige [Fuj78] observed that a vector $\mathbf{h} \in \Gamma_N^*$ must satisfy the polymatroidal axioms (P1)-(P3) in the definition below.

Definition 1. *A polymatroid (E, f) consists of a set E, called the ground set, and a set function $f : 2^E \to \mathbb{R}$, called the rank function, which satisfies the following properties:*

- *[(P1)]* $f(\emptyset) = 0$ *(normalized)*
- *[(P2)]* $f(\mathscr{A}) \geq f(\mathscr{B}), \forall \mathscr{B} \subseteq \mathscr{A} \subseteq E$ *(monotone)*
- *[(P3)]* $f(\mathscr{C}) + f(\mathscr{D}) \geq f(\mathscr{C} \cup \mathscr{D}) + f(\mathscr{C} \cap \mathscr{D}),\ \forall \mathscr{C}, \mathscr{D} \subseteq E$ *(submodular)*

Note that a polymatroid (E, f) is a matroid if it is integer valued, $f : 2^E \to \mathbb{Z}_{\geq 0}$, and $f(i) \leq 1$ for each $i \in E$. (P1)-(P3) generate all Shannon type inequalities [Yeu08]. Unfortunately, Shannon type inequalities are only necessary conditions for determining whether or not a candidate vector is entropic. Inequalities not implied by (P1)-(P3) that are satisfied by all entropic vectors exist and are called non-Shannon type inequalities. The first such inequality was found by Zhang and Yeung [Z. 98]. Hundreds of linear non-Shannon type inequalities have been found since the Zhang-Yeung inequality [DFZ06, Csi13]. Furthermore, Matús [F. 07] has shown that for $N \geq 4$, an infinite number of linear information inequalities is necessary to determine $\overline{\Gamma}_N^*$.

The characterization of the rate regions of multi-source multi-sink network coding instances depends inherently on the characterization of the region of entropic vectors. Multi-source multi-sink network coding over directed acyclic graphs (MSNC) was studied by Yan, Yeung and Zhang [YYZ12], where the authors give an implicit characterization of the rate regions in terms of Γ_N^*. The version of network coding problem considered here is *multi-source multi-sink network coding over directed acyclic hypergraphs* (HMSNC), which was studied recently by Li et. al. [Conb]. HMSNC is a general model that includes as special case the MSNC problem, the Independent Distributed Source Coding (IDSC) problem, and the Index Coding (IC) problem [Conb].

A HMSNC instance is completely described by the tuple $\mathsf{A} = (\mathscr{S}, \mathscr{G}, \mathscr{T}, \mathscr{E}, \beta)$. It consists of a directed acyclic hypergraph $(\mathscr{V}, \mathscr{E})$ where the nodes $\mathscr{V} = \mathscr{S} \cup \mathscr{T} \cup \mathscr{G}$ can be partitioned into the set of source nodes $\mathscr{S}$, the set of sink nodes $\mathscr{T}$ and the set of intermediate nodes $\mathscr{G}$. $\mathscr{E}$ is the set of directed hyperedges of the form $(v, \mathscr{A})$ where $v \in \mathscr{V}, \mathscr{A} \subseteq \mathscr{V} \setminus \{v\}$. The source nodes in $\mathscr{S}$ have no incoming edges and exactly one outgoing edge carrying the source message, the sink nodes $\mathscr{T}$ have no outgoing edges, and the intermediate nodes $\mathscr{G}$ have both incoming and outgoing edges. Any hyperedge $e \in \mathscr{E}$ then connects a source or an intermediate node to a subset of non-source nodes, i.e., $e = (i, \mathscr{F})$, where $i \in \mathscr{S} \cup \mathscr{G}$ and $\mathscr{F} \subseteq (\mathscr{G} \cup \mathscr{T}) \setminus \{i\}$. The number of source nodes will be

denoted by $|\mathscr{S}| = k$, with a source message associated with each node. For convenience, we shall label the source messages on the outgoing hyperedges of source nodes as $1, \ldots, k$. The remaining messages, carried on the rest of the hyperedges, are labeled $k+1, \ldots, |\mathscr{E}|$. Thus, we have a total of $N = |\mathscr{E}|$ messages. The demand function $\beta : \mathscr{T} \to 2^{[k]}$ associates with each sink node a subset of source messages that it desires. Each message $i \in [N]$ will be associated with a discrete random variable X_i, giving us the set $\boldsymbol{X}_N$ of message random variables.

As is typical in network coding, it is assumed that the random variables $\{X_i \mid i \in [k]\}$ associated with source messages are independent. The source independence gives rise to the first constraint on the entropy function,

$$\mathbf{h}_{[k]} = \sum_{i \in [k]} \mathbf{h}_k. \tag{1.2}$$

For notational convenience, we form the set $\mathcal{L}_1$ containing only the above constraint. For any $v \in G \cup \mathscr{T}$ define $\mathsf{In}(v)$ to be the set of its incoming messages, and for each $g \in \mathscr{G}$, define $\mathsf{Out}(g)$ to be the set of its outgoing messages. Every non-source message $i \in [N] \setminus [k]$, originates at some intermediate node $g \in \mathscr{G}$, and the associated random variable $X_i = f_i(\{X_j, j \in \mathsf{In}(g)\})$ is a function of all the input random variables of node g. This gives rise to constraints of the following type

$$\mathbf{h}_{\mathsf{In}(g) \cup \mathsf{Out}(g)} = \mathbf{h}_{\mathsf{In}(g)}, \ \forall g \in \mathscr{G}. \tag{1.3}$$

We collect the above constraints into a set $\mathcal{L}_2$. Finally, the demand function requires that each node $t \in \mathscr{T}$ be able to reconstruct those source messages with indices in $\beta(t)$, which naturally gives rise to the decoding constraints

$$\mathbf{h}_{\mathsf{In}(t) \cup \beta(t)} = \mathbf{h}_{\mathsf{In}(t)}, \ \forall t \in \mathscr{T}. \tag{1.4}$$

The decoding constraints are collected in a set $\mathcal{L}_3$. We shall denote the collected constraints on the entropy function associated with this network coding problem A as $\mathcal{I}_A = \mathcal{L}_1 \cup \mathcal{L}_2 \cup \mathcal{L}_3$.

Definition 2. *A network code for a HMSNC instance A is a collection of N discrete random variables $\boldsymbol{X}_N$ with joint entropies satisfying the constraints $\mathcal{I}_A$.*

Equivalently, a network code is an entropic polymatroid satisfying the constraints $\mathcal{I}_A$ on its rank function. From the standpoint of the rate region of a network coding problem, all that matters, given the knowledge that the joint entropies satisfy the constraints of the network, is the singleton entropies. Thus, we will say that a network code $\boldsymbol{X}_N$ *achieves* a rate vector $\mathbf{r} = (r_1, \ldots, r_N), r_i \in \mathbb{Z}_{\geq 0}$ if $\mathbf{h}_i = r_i,\ \forall i \in [N]$.

Let $\mathbb{R}^M$ be the space with subset entropies, source rate variables $\boldsymbol{\omega} = (\omega_i \mid i \in [k])$ and edge capacity variables $\mathbf{r} = (R_j \mid j \in [N] \setminus [k])$ as co-ordinates, i.e. $M = 2^N - 1 + N$. We use the term *rate vector* to refer to any vector $(\widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}}) \in \mathbb{R}^N$, obtained by concatenating a specified vector of source rates $\widetilde{\boldsymbol{\omega}}$ and a specified vector of edge capacities $\widetilde{\mathbf{r}}$. The achievable rate region of a HMSNC instance is the set of all rate vectors $(\widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}}) \in \mathbb{R}^N$, for which there exists a block code with a diminishing probability of error. The result of Yan et. al. [YYZ12], can be easily generalized to provide the (closure of) the set of all rate vectors achievable for a given HMSNC instance A [Conb], thereby yielding the rate region of the network as

$$\mathcal{R}^* = \mathsf{proj}_{\boldsymbol{\omega},\mathbf{r}}(\overline{\mathsf{con}(\Gamma_N^* \cap \mathscr{L})} \cap \mathscr{L}') \tag{1.5}$$

where $\mathsf{con}(\mathscr{B})$ is the conic hull of set of vectors $\mathscr{B}$, $\mathsf{proj}_{\boldsymbol{\omega},\mathbf{r}}(\mathscr{B})$ is the projection onto coordinates $\boldsymbol{\omega}, \mathbf{r}$ and where

$$\mathscr{L} \triangleq \{(\widetilde{\mathbf{h}}, \widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}}) \in \mathbb{R}^M \mid \widetilde{\mathbf{h}} \text{ satisfies } \mathcal{L}_1 \cup \mathcal{L}_2\} \tag{1.6}$$

$$\mathscr{L}' \triangleq \left\{ (\widetilde{\mathbf{h}}, \widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}}) \in \mathbb{R}^M \;\middle|\; \begin{array}{l} (\widetilde{\mathbf{h}} \text{ satisfies } \mathcal{L}_3) \\ \wedge(\widetilde{\mathbf{r}}_i \geq \widetilde{\mathbf{h}}_i, \forall i \in [N] \setminus [k]) \\ \wedge(\widetilde{\boldsymbol{\omega}}_i \leq \widetilde{\mathbf{h}}_i, \forall i \in [k]) \end{array} \right\}$$

Under this formulation, the rate region for the network is a convex cone, described by a series of inequalities linking the variables associated with the rates of the sources $\boldsymbol{\omega} = (\omega_i, i \in [k])$, with the variables associated with the capacities of the links $\mathbf{r} = (R_i, i \in [N] \setminus [k])$. As Γ_N^* is not yet fully characterized for $N \geq 4$, most of the HMSNC rate region characterizations known so far have been found using the method of *sandwich bounds* [DFZ07, Conb], which is based on substituting

polyhedral inner and outer bounds (see fig. 1.1) in place of Γ_N^* in (1.5). Polyhedral inner bounds on Γ_N^* can be obtained from representable matroids and polymatroids (see Ch. 2 for a detailed discussion), while polyhedral outer bounds can be obtained from information inequalities (see Ch. 3 for a detailed discussion). This yields inner and outer bounds $\mathcal{R}_{\mathsf{in}}$ and $\mathcal{R}_{\mathsf{out}}$ on $\mathcal{R}^*$,

$$\mathcal{R}_{\mathsf{x}} = \mathsf{proj}_{\boldsymbol{\omega},\mathbf{r}}(\Gamma_{\mathsf{x}} \cap \mathscr{L} \cap \mathscr{L}'), \mathsf{x} \in \{\mathsf{in},\mathsf{out}\} \tag{1.7}$$

Once we compute $\mathcal{R}_{\mathsf{in}}$ and $\mathcal{R}_{\mathsf{out}}$, if $\mathcal{R}_{\mathsf{in}} = \mathcal{R}_{\mathsf{out}}$, we know that $\mathcal{R}^* = \mathcal{R}_{\mathsf{in}} = \mathcal{R}_{\mathsf{out}}$, which is the essence of the sandwich method.

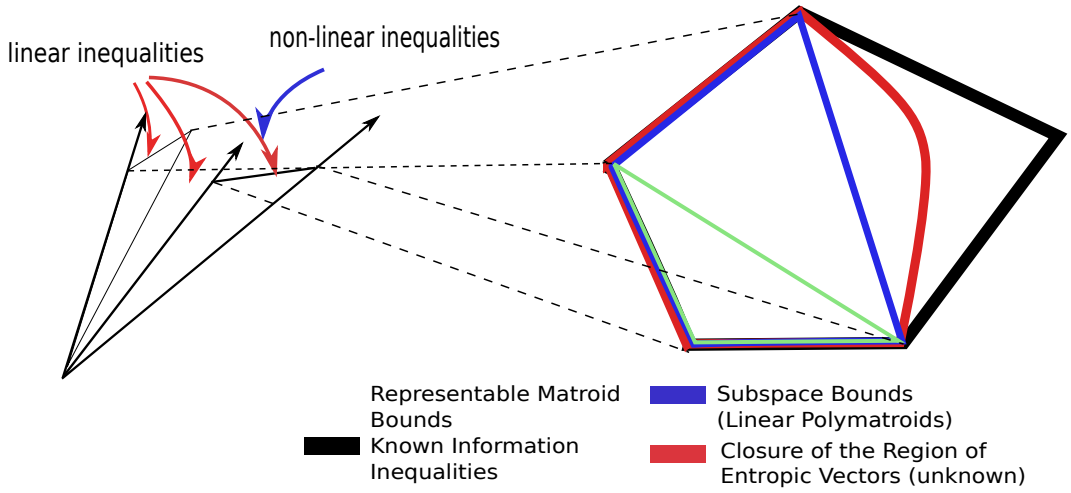

Figure 1.1: Illustration of the inner and outer bounds on $\overline{\Gamma_N^*}$. Under the sandwich method, bounds on $\mathcal{R}^*$ can be obtained as projections of bounds on $\overline{\Gamma_N^*}$.

Equation (1.7) is an indirect way of stating bounds on network coding rate regions. A more direct and practically useful way of specifying such bounds is by a series of inequalities linking the rates of the sources $\boldsymbol{\omega} = (\omega_i, i \in [k])$, with the capacities of the links $\mathbf{r} = (R_i, i \in [N] \setminus [k])$. The main problem considered in this thesis is that of designing computer programs that can compute such series of inequalities, given a HMSNC instance A and a description of an inner or outer bound.

The rest of this thesis is organized as follows: The algorithms for computation of inner bounds on network coding rate regions are discussed in chapter 2, while the algorithms for computation of outer bounds on network coding rate regions are discussed in chapter 3. Chapter 2 defines the constrained

linear representability problem (CLRP) for polymatroids that determines whether there exists a polymatroid that is linear over a specified field while satisfying a collection of constraints on the rank function. Using a computer to test whether a certain rate vector is achievable with vector linear network codes for a HMSNC instance and whether there exists a multi-linear secret sharing scheme achieving a specified information ratio for a given secret sharing instance are shown to be special cases of CLRP. Methods for solving CLRP built from group theoretic techniques for combinatorial generation are developed and described. These techniques form the core of the Information Theoretic Achievability Prover, and several computational experiments with interesting instances of network coding and secret sharing demonstrating the utility of the method are provided.

Chapter 3 considers the problem of automating rate region converse proofs in multi-source network coding, and an algorithm for explicitly computing polyhedral outer bounds on the rate regions of multi-source network coding instances using known bounds on the entropy function region is proposed. This algorithm is based on the Convex Hull Method, which is an algorithm to compute projections of polyhedra. An interpretation of the network symmetry group as a group of symmetries of a polyhedron is discussed, which in turn enables the use of well-known techniques for exploiting symmetry in polyhedral computation to reduce the complexity of calculating the rate region bounds. A variant of the convex hull method that is capable of exploiting the knowledge of the network symmetry group is discussed. The techniques described in this work are implemented in a software package, called the Information Theoretic Converse Prover. The computational performance of convex hull method for explicitly computing rate region outer bounds with varying knowledge of symmetry is discussed, followed by several examples of computation with the Information Theoretic Converse Prover.

While the characterization of the HMSNC rate regions is the main motivating problem of this thesis, the algorithms proposed in this work could be discussed in the context of several other mathematical problems whose solution depends on the characterization of the region of entropic vectors, such as,

1. Secret Sharing and Secure Network Coding [Pad13, CY02a],

2. Guessing games played on graphs [Rii06],

3. Subgroup size inequalities [CY02b],

4. Conditions for Quantum Contexuality [FC13].

ITAP and ITCP support various computations in the context of above problems, some of which will be considered as examples in chapters 2 and 3.

Chapter 2: Algorithms for constructing inner bounds

Using a computer to perform an arbitrary achievability proof requires one to know an algorithm to determine if there exists an almost entropic polymatroid satisfying certain constraints on its rank function. Finding such an algorithm is a fundamental open problem in information theory, also known as the problem of characterization of the closure of the region of entropic vectors ($\overline{\Gamma_N^*}$). We consider a special case of this very difficult problem, which we call the Constrained Linear Representability Problem (CLRP) for polymatroids. We show that the ability to solve CLRP can automate the achievability proofs that one encounters while determining the performance of linear codes in multi-source multi-sink network coding over directed acyclic hypergraphs and in secret sharing and while proving new linear rank inequalities.

This chapter develops and describes a method for solving constrained linear representability problems built from group theoretic techniques for combinatorial generation. More precisely, in section 2.1, after reviewing the notion of polymatroid representability, we define two variants of CLRP, one existential and one enumerative. Then, Section 2.2 shows in detail how the problems of calculating achievability proofs for fundamental limits for network coding rate regions and secret sharing can be viewed as instances of these CLRPs. Section 2.3 then defines key concepts and terminology which enable techniques developed for combinatorial generation to be applied to CLRP, explaining along the way key decisions enabling CLRP to be solved in an efficient manner that can exploit problem symmetry and handle isomorphism. Building upon these ideas, section 2.4 presents the developed algorithm for solving CLRP, which is also implemented as a GAP package the Information Theoretic Achievability Prover– ITAP, the first of its kind, accompanying the the thesis. Finally, Section 2.5 describes several quantities playing a key role in the complexity of the developed method, then provides a series of examples of achievability problems solved with ITAP.

2.1 Constrained Linear Representability Problems for polymatroids

The *representable* polymatroids are a sub-class of entropic polymatroids that arise from subspace arrangements.

Definition 3. *A subspace arrangement in a k-dimensional vector space $V = \mathbb{K}^k$ over some field $\mathbb{K}$ is a multiset of N subspaces $\mathscr{S} = \{V_1, V_2, \ldots, V_N\}$ with $V_n \subseteq V\ \forall n \in [N]$. All vector spaces considered in this work are assumed to be finite dimensional. The rank function of the subspace arrangement is $\mathsf{rk} : 2^{[N]} \to \mathbb{Z}$ defined as*

$$\mathsf{rk}(\mathscr{I}) \triangleq \mathsf{dim} \sum_{i \in \mathscr{I}} V_i, \quad \forall \mathscr{I} \subseteq [N] \tag{2.1}$$

Note that in (2.1) the sum is understood to be the direct sum of subspaces of a vector space. As $([N], \mathsf{rk})$ satisfies (P1)-(P3), it is a valid polymatroid. However, the converse does not necessarily hold true. This motivates the notion of representable polymatroids.

Definition 4. *A polymatroid $P = (E, f)$ is said to be representable if there exists a subspace arrangement $\mathscr{S} = \{V_1, \ldots, V_{|E|}\}$ over some field $\mathbb{K}$ and a bijection $m : E \to [|E|]$ s.t. $f(\mathscr{S}) = \mathsf{rk}(m(\mathscr{S}))$, $\forall \mathscr{S} \subseteq E$.*

A theorem of Rado [Rad57] which states that a polymatroid is representable over an infinite field then it is also representable over some finite field, allows us to restrict attention to fields $\mathbb{K}$ with only a finite number of elements when considering representability of polymatroids. In the above definition, if $\mathbb{K}$ is a finite field with q elements (i.e. q is a prime power), then we say that the polymatroid is $\mathbb{F}_q$-*representable*. Note that it is possible for $P = (E, f)$ to not be $\mathbb{F}_q$-representable while a scaled version of P i.e. $P^c = (E, cf), c > 0$ is. We emphasize this distinction, so as to avoid confusion later. One fact that makes representable polymatroids interesting is stated below.

Lemma 1. *([HRAS00], Thm. 2) Every representable polymatroid is proportional to an entropic polymatroid.*

To see why the above statement is true, and to fix some notation, consider a representable polymatroid $P = ([N], f)$ and let $\{V_1, \ldots, V_N\}$ be the associated subspace arrangement over finite

field $\mathbb{F}_q$ with q elements. Let the bijection mentioned in def. 4 be the identity map from $[N]$ to itself. We assume that each subspace $V_i, i \in [N]$ is presented as a set $\mathsf{rep}(i)$ of $f(i)$ vectors in $\mathbb{F}_q^{f([N])}$ forming a basis of the subspace V_i. Let $\mathbb{M}(\mathscr{I})$ be the matrix $[\mathsf{rep}(i)|i \in \mathscr{I}]$ for each $\mathscr{I} \subseteq [N]$ (i.e. collect together the columns in $\mathsf{rep}(i)$ for each $i \in \mathscr{I}$). Consider a random row vector $\mathbf{u} \sim \mathcal{U}(\mathbb{F}_q^{f([N])})$ uniformly distributed over $\mathbb{F}_q^{f([N])}$, and create a collection of random variables $\boldsymbol{X}_N = \{X_1, \ldots, X_N\}$ such that $X_i = \mathbf{u}\mathbb{M}(i)$ is a random variable taking values in $\mathbb{F}_q^{f(i)}$. The entropy function maps $\boldsymbol{X}_N$ to a vector $\mathbf{h}$ such that for each $\mathscr{A} \subseteq [N]$, the entropy $\mathbf{h}_{\mathscr{A}} = f(\mathscr{A}) \log_2 q = \mathsf{rk}(\mathbb{M}(\mathscr{A})) \log_2 q$. The polymatroid $P^{\log_2 q} = ([N], (\log_2 q)f)$ is then entropic by construction, thus completing the proof.

We now state the representability problem for integer polymatroids.

- [E2] Given an integer polymatroid $P = (E, f)$, determine if there exists a representation of P^α over a finite field $\mathbb{F}_q$ for some $\alpha \in \mathbb{N}$.

Lemma 1, along with the fact that $\overline{\Gamma_N^*}$ is a convex cone allow us to construct an inner bound $\Gamma_N^{\mathsf{space}}$ on $\overline{\Gamma_N^*}$, that we call the *subspace inner bound*, which is the conic hull of all representable polymatroids. Problem (E2) is equivalent to testing membership of an integer polymatroid within an inner bound to $\Gamma_N^{\mathsf{space}}$ formed by taking the conic hull of all polymatroids representable over a particular finite field. $\Gamma_N^{\mathsf{space}}$ is polyhedral for $N \leq 5$ [DFZ09], while for $N \geq 6$, it remains unknown whether this set is polyhedral. The linear inequalities that are true for all points in $\Gamma_N^{\mathsf{space}}$ are called *rank inequalities*. For $N \leq 3$, the minimal (conic independent) set of rank inequalities is same as the minimal set of Shannon-type inequalities, while for $N = 4$, the minimal set of rank inequalities is the minimal set of Shannon-type inequalities and all 6 permutations of Ingleton's inequality [Ing71, HRAS00]. For $N = 5$, there are 24 new linear rank inequalities that, together with the Shannon-type and Ingleton inequalities, form the minimal set of inequalities [DFZ09]. One way to approach [E2] computationally, is to solve a restriction of [E2] for specific α and size of finite field q, iteratively for increasing values of α and q until we find a representation. This leads us to the following problem:

- [E2$_q$] Given a polymatroid $P = (E, f)$, determine if there exists a representation of P over the finite field $\mathbb{F}_q$ with q elements.

A special case of [E2$_q$] that is well-studied is the situation where P is restricted to be a matroid. A

famous conjecture of Rota [Rot71], recently declared proven [GGW14], states that for every finite field $\mathbb{F}_q$, there are only a finite number of forbidden minors, a series of smaller matroids obtained through contraction and deletion, for a matroid to be representable over that field.

We now define the notion of representation of a representable polymatroid.

Definition 5. *A representation of a $\mathbb{F}_q$-representable polymatroid $([N], f)$ associated with a multiset $\mathscr{S} = \{V_1, \ldots, V_n\}$ of subspaces is a multiset of matrices $\{\mathbb{M}(i), i \in [N]\}$ where each subspace V_i is represented as a $f([N]) \times f(i)$ matrix $\mathbb{M}(i)$ whose columns are from the set $\mathsf{rep}(i)$ for all $i \in [N]$.*

In the context of the definition above, two representations of the same polymatroid are considered distinct if they correspond to different multisets of subspaces. We will consider two other notions of distinctness via isomorphism later in the manuscript. Note that each subspace in such a multiset may be represented by several different bases. For the sake of simplicity, we do not distinguish between two representations that differ only in this sense. For the special case of matroids of ground set size N, we will assume that a representation is a multiset of vectors $\{\mathbf{v}_1, \ldots, \mathbf{v}_N\}$, each of which generates a subspace of dimension at most 1. Let $\mathsf{Gr}_q(r,k), 0 \leq k \leq N$ be the set of all k-dimensional subspaces of $\mathbb{F}_q^r$. The set $\mathsf{Gr}_q(r,k)$ is also known as the Grassmannian. The size of $\mathsf{Gr}_q(r,k)$ is given by the q-ary Gaussian binomial coefficient:

$$|\mathsf{Gr}_q(r,k)| = \binom{r}{k}_q \triangleq \frac{(q^r-1)(q^{r-1}-1)\ldots(q^{r-k+1}-1)}{(q^k-1)(q^{k-1}-1)\ldots(q-1)} \tag{2.2}$$

For a set $K \subseteq [r]$, define $\mathsf{Gr}_q(r,K) \triangleq \bigcup_{k\in K} \mathsf{Gr}_q(r,k)$. We can associate with a polymatroid $P = ([N], f)$, a set K_P of distinct singleton ranks i.e. $K_P \triangleq \{f(i) \mid i \in [N]\}$. In the spirit of compactness, it is also important to consider *simple* polymatroids. A matroid is said to be simple if it has no parallel elements or loops [Oxl11], and likewise we have the following definition for a simple polymatroid

Definition 6. *A polymatroid (E, f) is said to be simple if it satisfies following conditions:*

1. *$\nexists e_1, e_2 \in E$ s.t. $f(\{e_1, e_2\}) = f(e_1) = f(e_2)$*
2. *For every $e \in E$, $f(e) > 0$*

If there exist elements e_1, e_2 satisfying condition 1 of definition 6 above, they are said to be *parallel* with respect to each other. In fact, there can exist multiple such elements forming a parallel class.

Definition 7. *For a polymatroid (E, f), $S \subseteq E$ is said to form a parallel class if*

$$f(S) = f(s)\ \forall s \in S \tag{2.3}$$

In this case, $|S|$ is said to be the degree of the parallel class.

If there exists an element violating condition 2 of definition 6, then it is called a *loop*. The number of ground set elements that are loops is called the loop degree of the polymatroid. Given a polymatroid $P = (E', f')$ with an order on the ground set E', we can associate it with a simple polymatroid (E, f), s.t. $E \subseteq E'$ and $f(S) = f'(S), \forall S \subseteq E$ where E is obtained by deleting all loops and all but one element of each parallel class from E'. Furthermore, we can decide to keep the smallest member of each parallel class under the aforementioned order on E'. We call (E, f) obtained in this manner the *unique simple polymatroid* associated with (E', f'), denoted as $\mathsf{us}(P)$, and define, via an abuse of notation, $|\mathsf{us}(P)| = |E|$. Having defined the parameters K_P and $\mathsf{us}(P)$ we can define the set $\mathscr{P}^q(N, r, K, s)$ as the set of representations of polymatroids with ground set size N, containing subspaces of $\mathbb{F}_q^r$ s.t. $K_P \subseteq K$ and $|\mathsf{us}(P)| = s$ over $\mathbb{F}_q$. Finally, we define the set $\mathscr{P}^q(N, (r_l, r_u), K, (s_l, s_u))$ of polymatroid representations as

$$\mathscr{P}^q(N, (r_l, r_u), K, (s_l, s_u)) \triangleq \bigcup_{\substack{r_l \le r \le r_u \\ s_l \le s \le s_u}} \mathscr{P}^q(N, r, K, s) \tag{2.4}$$

Henceforth, we shall refer to the tuple $\mathbf{c} = (N, (r_l, r_u), K, (s_l, s_u))$ as the *class tuple* and abbreviate $\mathscr{P}^q(N, (r_l, r_u), K, (s_l, s_u))$ to $\mathscr{P}^q(\mathbf{c})$. This leads us to state precisely what we mean by a class of linear codes for the purpose of this work.

Definition 8. *A class of linear codes is any set $\mathscr{P}^q(\mathbf{c})$ where $\mathbf{c} = (N, (r_l, r_u), K, (s_l, s_u))$ is the class*

tuple satisfying $r_l \leq r_u \leq N \cdot \max K, s_l \leq s_u \leq N$ *where* K *is a finite set containing non-negative integers.*

We are now ready to define the central problem adressed in this work. In several problems of practical interest, which will be described in the next section, rather than testing representability for a polymatroid specified by its rank function, it is of interest to determine whether there exists a polymatroid representable over $\mathbb{F}_q$ satisfying a specified collection of linear constraints on its rank function, and belonging to a particular class of linear codes. We will name this problem the constrained linear representability problem (CLRP) for polymatroids. More formally, let $\mathcal{I}$ be a collection of linear constraints on the rank function of a polymatroid with ground set $[N]$ and $\mathbf{c}$ be any valid class tuple. Then, the existential variant of CLRP can be stated as follows:

- [CLRP$_q$-EX] Given a system $\mathcal{I}$ of constraints on the rank function of a polymatroid with ground set $[N]$, determine if there exists a polymatroid that satisfies $\mathcal{I}$ in $\mathscr{P}^q(\mathbf{c})$.

Note that [E2$_q$] is a special case of [CLRP$_q$], as a pre-specified rank function f of a polymatroid (E, f) can be interpreted as a collection of 2^N linear constraints on the rank function with $N = |E|$. As we shall discuss in next section, it is sometimes of interest to find *all* polymatroids satisfying $\mathcal{I}$ and belonging to $\mathscr{P}^q(\mathbf{c})$. We now state CLRP$_q$-EN to be the problem of constructing the members of $\mathscr{P}(N, (r_l, r_u), K, (s_l, s_u))$ up to some notion of equivalence $\equiv$.

- [CLRP$_q$-EN] Given a system $\mathcal{I}$ of constraints on the rank function of a polymatroid with ground set $[N]$, list a representative from each equivalence class, under $\equiv$, of polymatroids in $\mathscr{P}(N, (r_l, r_u), K, (s_l, s_u))$ representable over $\mathbb{F}_q$, that satisfy $\mathcal{I}$.

An algorithm to solve CLRP$_q$-EN can also solve CLRP$_q$-EX, by halting as soon as it finds one (the first) such polymatroid representation. The design of finite-terminating algorithms to solve CLRP$_q$-EN is the subject of much of this chapter. The approach we use is that of using combinatorial generation techniques which are able to construct combinatorial objects satisfying certain desired properties systematically, exhaustively, and efficiently.

2.2 Network Coding, Secret Sharing and CLRP

We show that creating computer assisted achievability proofs and rate regions in network coding and secret sharing with finite length linear codes can be posed as variants of CLRP.

2.2.1 Network coding as CLRP

In the context of this chapter, for a HMSNC instance of size N, we are particularly interested in rate vectors $(\widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}}) \in \mathbb{Q}^N$ that are achievable with *linear* network codes. Given a class $\mathscr{P}^q(\mathbf{c})$ of linear codes, one can consider the following inner bound on Γ^*_N:

$$\Gamma_N^{\mathscr{P}^q(\mathbf{c})} = \mathsf{con}(\{\mathbf{h} \in \mathbb{R}^{2^N-1} \mid \mathbf{h} \in \mathscr{P}^q(\mathbf{c})\}) \tag{2.5}$$

yielding the inner bound $\mathcal{R}_{\mathscr{P}^q(\mathbf{c})} = \mathsf{proj}_{\boldsymbol{\omega},\mathbf{r}}(\Gamma_N^{\mathscr{P}^q(\mathbf{c})} \cap \mathscr{L} \cap \mathscr{L}')$ to the rate region $\mathcal{R}^*$.

A linear network code is a representable polymatroid that satisfies constraints $\mathcal{I}_A$ on its rank function. As an aside, note that a linear network code is said to be *scalar* if the associated polymatroid is in fact a matroid. From the perspective of achievability proofs for network coding rate regions, and the inner bounds associated with linear codes $\mathcal{R}_{\mathscr{P}^q(\mathbf{c})}$, there are two variants of the constrained linear representability problem that are of interest, an existential one building from CLRP$_q$-EX and an enumerative one building from CLRP$_q$-EN. The existential variant takes a specified rate vector $(\widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}})$ and asks whether there is a code over $\mathbb{F}_q$ which achieves it.

- [E3$_q$-EX] Given a HMSNC instance A, determine if there exists polymatroid of size N representable over $\mathbb{F}_q$ satisfying $\mathcal{I}_A$ achieving a rate vector $(\widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}})$.

Note that E3$_q$-EX is a special case of CLRP$_q$-EX, as we can interpret the requirement to achieve a rate vector $\mathbf{r}$ as additional linear constraints $\mathcal{I}_{(\widetilde{\boldsymbol{\omega}},\widetilde{\mathbf{r}})} = \{\{\mathbf{h}_i = \widetilde{\boldsymbol{\omega}}_i\}, i \in [k]\} \cup \{\{\mathbf{h}_i = \widetilde{\mathbf{r}}_i\}, i \in [N] \setminus [k]\}$. Hence, it is an instance of CLRP$_q$-EX with constraints $\mathcal{I} = \mathcal{I}_A \cup \mathcal{I}_{\widetilde{\boldsymbol{\omega}},\widetilde{\mathbf{r}}}$ and class tuple $\mathbf{c} = (N, (\sum_i r_i, \sum_i r_i), \mathsf{unique}(\mathbf{r}), (k, N))$, where $\mathsf{unique}(\widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}})$ is the set of unique values in vector $(\widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}})$.

The enumerative variant associated with achievability proofs in network coding rate regions aims

instead to find, up to isomorphism $\equiv$, all linear codes in a given class that satisfy the constraints of the network.

- [E3$_q$-EN] Given a class tuple $\mathbf{c}$ and a HMSNC instance A, list a representative from equivalence class of codes (under $\equiv$) yielding joint entropy vectors $\mathbf{h} \in \mathscr{P}^q(\mathbf{c})$ s.t. $\mathbf{h}$ satisfies $\mathcal{L}_1, \mathcal{L}_2$ and $\mathcal{L}_3$ associated with A.

Observe that E3$_q$-EN is likewise a special case of CLRP$_q$-EN associated with the same code class $\mathbf{c}$, where again the system of constraints is given by the constraints $\mathscr{L}$ built from the network coding problem A.

Having defined the two problem classes, some comparison and historical discussion is in order. The existing computation based information theory achievability proofs literature, while thin, is focussed on the former of these two problems E3$_q$-EX. This matches the situation with the computer aided converse proof literature, which, with the notable exception from our own previous work [Apt14, Con12, Con13, AW15b, Cona, Conb], aims to provide proofs verifying a putative given inequality [YR08, RES08, Tia14] (i.e. membership testing in the polar $\mathcal{R}_{\mathsf{out}}^{\circ}$), rather than generating that inequality as part of a full description of an outer bound $\mathcal{R}_{\mathsf{out}}$ in the first place. The seminal network coding paper of Koetter and Medard [KM03] provided an algebraic formulation of a slight variation of E3$_q$-EX which replaces $\mathbb{F}_q$ with its algebraic closure $\mathbb{F}_q^*$. Under this Koetter and Medard formulation, one can construct a system of polynomial equations with coefficients in $\mathbb{F}_q$ s.t. non-emptiness of the associated algebraic variety implies the existence of a polymatroid satisfying $\mathcal{I}$ that is representable over the algebraic closure of $\mathbb{F}_q$ and vice versa. The problem of determining the emptiness of the algebraic variety associated with these polynomial equations can be solved by computing the Gröbner basis [CLO07] of the ideal generated by them in $\mathbb{F}_q[\mathbf{x}]$. Shifting back to E3$_q$-EX, i.e. if one is interested in existence of solution over a specific finite field (and not the algebraic closure of it), one can instead compute the Gröbner basis in the quotient ring $\mathbb{F}_q[\mathbf{x}] \setminus \langle x_i^q = x_i \mid i \in [n] \rangle$. An important subsequent work of Subramanian and Thangaraj [ST10] refined the algebraic formulation of Koetter and Medard to switch to using path gain variables instead of variables that represent local coding coefficients. The benefit of this path gain formulation is that the polynomials contain

monomials of degree at most 2, which can have substantial complexity benefits over the original Koetter and Medard formulation in the Gröbner basis calculation. The details of how to adapt this method of Subramanian and Thangaraj to solve E3$_q$-EX are provided in algorithm 4 in the appendix. As we will show via computational experiments in §2.5, the new methods for solving E3$_q$-EX that we will provide in this work, which, in fact, are built from enumerative methods best suited to solving E3$_q$-EN, in some problems enable substantial reduction of runtime relative to the Subramanian and Thangaraj Gröbner basis based formulation. The reduction in runtime appears to be most pronounced when the rate vectors $\mathbf{r}$ being tested contain integers greater than one.

More broadly, when one has computed a polyhedral outer bound $\mathcal{R}_{\mathsf{out}}$ to $\mathcal{R}^*$, e.g. through polyhedral projection with the convex hull method [LL93, W. 08] as described in our previous work [Apt14, AW15b], one can utilize a series of E3$_q$-EX problems to determine if $\mathcal{R}_{\mathscr{P}^q(\mathbf{c})} = \mathcal{R}_{\mathsf{out}}$ as follows. For each extreme ray $(\widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}})$ of $\mathcal{R}_{\mathsf{out}}$, one uses E3$_q$-EX to determine if $(\widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}})$ is achievable with linear codes in the associated class $\mathbf{c}$. If the answer is yes for each of the extreme rays of $\mathcal{R}_{\mathsf{out}}$, then $\mathcal{R}^* = \mathcal{R}_{\mathsf{out}} = \mathcal{R}^* = \mathcal{R}_{\mathscr{P}^q(\mathbf{c})}$. However, if the answer to E3$_q$-EX is no for one or more of the extreme rays $\mathcal{R}_{\mathsf{out}}$, then it can be of interest to determine the region $\mathcal{R}_{\mathscr{P}^q(\mathbf{c})}$ achievable with this class of codes.

This more difficult problem of determining the inequality description of the polyhedral cone $\mathcal{R}_{\mathscr{P}^q(\mathbf{c})}$ can be solved with an algorithm providing a solution to E3$_q$-EN. Indeed, denoting the set of vectors forming the solution to E3$_q$-EN as $\mathscr{P}^q(\mathbf{c}, A)$, one can determine $\mathcal{R}_{\mathscr{P}^q(\mathbf{c})}$ with e.g. the following remaining steps:

1. Delete all co-ordinates of $\mathbf{h} \in \mathscr{P}^q(\mathbf{c}, A)$ other than singletons
2. Append the resulting list of vectors with $-\mathbf{e}_i, i \in [k]$ and $\mathbf{e}_i, i \in [N] \setminus [k]$, where $\mathbf{e}_i$ is the ith column of the identity matrix of dimension N
3. Compute the inequality description of the conic hull of the resulting set of N dimensional vectors

The resulting inequalities are the system of inequalities defining the polyhedral cone $\mathcal{R}_{\mathscr{P}^q(\mathbf{c})}$, and the

resulting extreme rays of the conic hull form the optimal codes which can be time-shared to achieve any point in $\mathcal{R}_{\mathscr{P}^q(\mathbf{c})}$ arbitrarily closely [Cona, Conb]. For the rest of the chapter, rate regions will be specified as inequalities amoungst source rate variables $\omega_i, i \in [k]$ and edge rate variables $R_j, j \in [N] \setminus [k]$, whereas the rate vectors will be specified as $(\boldsymbol{\omega}, \mathbf{r})$, where $\boldsymbol{\omega}$ and $\mathbf{r}$ are source and edge rate vectors of size k and $N-k$ respectively, to enhance readability.

2.2.2 Secret sharing as CLRP

The next application where the ability to solve CLRP-EX is useful is secret sharing. Secret sharing [Sha79, Bei11] is concerned with the sharing of a secret among a collection of people or entities such that only certain subsets of them can recover the secret. Let Δ be the set of participants. We assume that $|\Delta| = N$, with participants labeled by $[N]$. The secret sharer is called the dealer, and has label 1 while the rest of the participants, called parties have labels in $\{2, \ldots, N\}$. The dealer bears a secret, and gives each party a chunk of information, called a *share*. The subsets of $\Delta \setminus \{1\}$ that are allowed to reconstruct the secret are called the authorized sets. The set of all authorized subsets, called an *access structure*, and denoted by Γ is a monotone collection of sets i.e. $\mathscr{A} \in \Gamma \implies \mathscr{B} \in \Gamma, \forall \mathscr{B} \supset \mathscr{A}$. We associate a random variable X_1 with the dealer and random variables $X_2, \ldots, X_N$ with the parties. Given an access structure Γ containing authorized subsets of $\{2, \ldots, N\}$, every authorized set of participants must be able to recover the secret, imposing the following constraints on the entropy function.

$$\mathbf{h}_{\mathscr{S} \cup \{1\}} = \mathbf{h}_{\mathscr{S}}, \ \forall \mathscr{S} \in \Gamma \tag{2.6}$$

Moreover, an un-authorized set must not be able to gain any information about the secret, imposing the following constraints on the entropy function.

$$\mathbf{h}_1 + \mathbf{h}_{\mathscr{S}} = \mathbf{h}_{\mathscr{S} \cup \{1\}}, \ \forall \mathscr{S} \notin \Gamma \tag{2.7}$$

We denote the collection of constraints specified by (2.6) and (2.7) as $\mathcal{I}_\Gamma$.

Definition 9. *A secret sharing scheme (SSS) for a access structure* $\Gamma \subseteq 2^{\{2,\ldots,N\}}$ *is a collection of* N *random variables whose joint entropies satisfy the constraints* $\mathcal{I}_\Gamma$.

A SSS is linear if it is associated with a representable matroid with ground set size N satisfying $\mathcal{I}_\Gamma$. A SSS is said to be multi-linear if it is associated with a representable polymatroid satisfying $\mathcal{I}_\Gamma$. We can now formulate the achievability problem for secret sharing as follows.

- [E5$_q$] Given an access structure Γ, determine if there exists a polymatroid of size N representable over $\mathbb{F}_q$ satisfying $\mathcal{I}_\Gamma$ with secret size r_1 and share sizes $(r_2, \ldots, r_N)$.

Again, [E5$_q$] can be seen as special case of CLRP$_q$ respectively, by interpreting the requirement to have specific secret and share sizes as a collection of constraints $\mathcal{I}_{\mathbf{r}} = \{\{h_i = r_i\}, i \in [N]\}$, where $\mathbf{r} = (r_1, \ldots, r_N)$. The class of codes $\mathscr{P}^q(\mathbf{c})$ is specified by the class tuple $\mathbf{c} = (N, (\max_i r_i, \sum_i r_i - 1), \mathsf{unique}(\mathbf{r}), (2, N))$

Secret sharing schemes can be classified into two types: those constructed by putting together other secret sharing schemes (decomposition constructions) and those that are constructed from scratch without using existing schemes (basic constructions). Decomposition constructions of secret sharing schemes have been proposed by Blundo et al. [BDSSV95], Stinson [Sti94], van Dijk et al. [DJM98, vDKST06]. Basic constructions of linear schemes were proposed by Bertilsson et al. [BI93] and van Dijk [vD97]. In [vD97], van Dijk refers to a secret sharing scheme associated with a representable polymatroid as the generalized vector space construction. He also proposes a backtracking algorithm for determining whether a certain worst case information rate (in context of problem [E5$_q$], $\frac{\mathsf{max}_{i\in\{2,\ldots,N\}} r_i}{r_1}$ is the worst case information rate) can be achieved in a given access structure by means of the generalized vector space construction, thus providing an algorithm to solve a variant of (E5$_q$). Secure network coding (SNC) [CY02a] is a generalization of secret sharing. Again, one can form a variant of CLRP in the context of SNC.

2.3 Polymatroid Isomorphism, partial $\mathcal{I}$-feasibility, and extension

Algorithms for generating various types of combinatorial objects that are special-cases of polymatroids: graphs, linear codes, matroids and k-polymatroids, up to some equivalence relation $\equiv$, have

been studied in the literature, see e.g. [BCH73, MR08, MMIB12, Sav14, BBF+06]. A common theme in these algorithms is that of "orderly generation". Here, each combinatorial object X has an associated non-negative number "order" $o(X)$ which is the size of the object. This could be the number of edges for graphs, block length for linear codes or the ground set size for matroids/polymatroids. The aim of these algorithms is to list inequivalent objects of order up to some $N \geq 1$. The name orderly generation follows from the fact that these algorithms proceed in an orderly fashion, by constructing inequivalent objects of size $i \leq N$ from the list of inequivalent objects of size $i-1$, recursively in i. The key tool used for constructing an object of size i from an object of size $i-1$ is the notion of *augmentation*. For matroids or polymatroids, an example of such operation is extension, which augments a polymatroid with ground set size $i-1$ by adding a new element to the ground set, thereby constructing a polymatroid with ground set size i. In order to apply this orderly generation technique to solving CLRP$_q$-EN and CLRP$_q$-EX, three decisions must be made. The remaining subsections of this section describe the answers to these questions in detail, while we give a high level description here.

First, one must specify precisely the combinatorial objects whose generation is being attempted. In §2.3.1, given a specific collection of constraints $\mathcal{I}$ on the rank function of a polymatroid of ground set size N, we define the concept of linear $\mathcal{I}$-polymatroids which is a mathematical formalization of 'polymatroids in $\mathscr{P}^q(\mathbf{c})$, that satisfy $\mathcal{I}$' as mentioned in CLRP$_q$-EN. For a given class of codes $\mathscr{P}^q(\mathbf{c})$ and constraints $\mathcal{I}$ on a polymatroid of ground set size N, a $\mathbb{F}_q$-linear $\mathcal{I}$-polymatroid is a pair (P, ϕ) where P is a $\mathbb{F}_q$-representable polymatroid in $\mathscr{P}^q(\mathbf{c})$, having ground set size N and ϕ is a bijection $\phi : [N] \to [N]$. The domain of ϕ is understood to be the ground set of P while the range is understood to be the set associated with the set function that $\mathcal{I}$ is constraining (we choose to label both these sets with $[N]$). In particular, ϕ is a map under which P satisfies $\mathcal{I}$. ϕ is reminiscent of the network-matroid mapping of Dougherty, Freiling and Zeger [DFZ07], albeit being set up the other way around (from a polymatroid to a network, when $\mathcal{I}$ arises from a network coding instance). A $\mathbb{F}_q$-representable polymatroid for which such a mapping ϕ exists is said to have the property of $\mathcal{I}$-feasibility. In order to embed the constraint of $\mathcal{I}$-feasibility into an orderly generation

oriented algorithm gradually growing the polymatroid's ground set, we further expand this idea to polymatroids of ground set size $i \leq N$ via the property of partial $\mathcal{I}$-feasibllity or $p\mathcal{I}$-feasibility, to define linear $p\mathcal{I}$-polymatroids, which are $\mathbb{F}_q$-representable polymatroids with ground set size $i \leq N$ for which there exists an injective mapping $\phi : [i] \to [N]$, under which it satisfies a subset of $\mathcal{I}$ associated with $\phi([i])$. The mapping ϕ in this case is called a p-map, which is shorthand for partial map. Noting that the smaller polymatroid created via the deletion of any ground set elements of a $p\mathcal{I}$-feasibile polymatroid must also be $p\mathcal{I}$-feasible polymatroid, we arrive at the conclusion that the criterion of $p\mathcal{I}$-feasibility can be embedded into the orderly generation algorithm. Namely, if at any time in the extension process we encounter a polymatroid that is not $p\mathcal{I}$-feasible, we do not need to compute any extensions of this polymatroid, as they will also not be $p\mathcal{I}$-feasible.

In addition to the goal of generating only polymatroids which will ultimately obey the constraints $\mathcal{I}$, we are also interested in generating exclusively the *essentially different* ones. This leads to the next of the three decisions which must be made: the notion of equivalence. The equivalence relation $\equiv$ mentioned while defining CLRP captures this idea. In §2.3.2, we discuss two notions of isomorphism: strong isomorphism and weak isomorphism. Both of these notions have been used in literature to generate combinatorial objects related to polymatroids. Literature related to generation of matroids and 2-polymatroids [BCH73, MR08, MMIB12] uses strong isomorphism. The literature concerning the generation/classification of linear codes [BBF$^+$06] uses weak isomorphism as it is associated with the semi-linear isometry relation on linear codes of given block-length. We use words 'strong' and 'weak' to emphasize the fact that weak isomorphism is a refinement of strong isomorphism, which we elaborate on using example 1. The benefit of using weak isomorphism for representability polymatroids over strong isormorphism is that it enables group theoretic techniques to provide an algorithm that can exhaustively generate simple $p\mathcal{I}$-feasible polymatroids that are distinct under weak isomorphism. In order to generate representable polymatroids that are distinct under strong isomorphism, while also not necessarily simple, these weakly non-isomorphic can be subjected to further strong isomorphism testing among those pairs whose parameters admit the possibility of strong isomorphism. Ultimately, we settle on an algorithm which uses both of these notions of

isomorphism together, which is implemented in our software ITAP, as described in Section 2.4.2.

In section 2.3.3, we describe a general template for generating all members of a particular class of codes $\mathscr{P}^q(\mathbf{c})$ up to equivalence relation $\equiv$, which is of interest by itself, as it allows one to construct inner bounds $\Gamma_N^{\mathscr{P}^q(\mathbf{c})}$ on $\overline{\Gamma_N^*}$. While this template is motivated by algorithms to generate classes of polymatroids more general than linear polymatroids, we use a restrictive definition of polymatroid extension, that preserves $\mathbb{F}_q$-representability, thus avoiding the problem of handling non-representable extensions of representable polymatroids. §2.3.4 defines a notion of augmentation for p-maps, called *p-map extension*. This, along with our restrictive notion of polymatroid extension, provides an augmentation operation for $p\mathcal{I}$-polymatroids, settling the third key decision of choosing the augmentation operation for designing an orderly generation algorithm. We also provide an algorithmic description of how such augmentation can be performed in practice in §2.3.4, along with the details of symmetry exploitation while performing such augmentation in §2.3.5.

2.3.1 $p\mathcal{I}$-feasibility, $p\mathcal{I}$-polymatroids and $\mathcal{I}$-polymatroids

For a collection of constraints $\mathcal{I}$ we will first define the property of $p\mathcal{I}$-feasibility which plays a central role in algorithms discussed later in the chapter. For simplicity, we assume that $\mathcal{I}$ is presented as a set of linear constraints on the set function defined over $[N]$, with N being called the *size* of $\mathcal{I}$. For a polymatroid $P = ([i], f)$ with $i \leq N$, we can consider a partial map (p-map) that is an injective mapping ϕ from from $[i]$ to a $[N]$ that satisfies the relevant part of $\mathcal{I}$. More formally, given a subset $\mathscr{X} \subseteq [N]$ we denote by $\mathcal{I}(\mathscr{X})$ the subset of $\mathcal{I}$ that contains only those constraints whose involved sets, upon which the set function is evaluated, exclusively to include indices from $\mathscr{X}$.

Definition 10. *A polymatroid $([i], f)$ with $i \leq N$ is a $p\mathcal{I}$-feasible if there exists an injective mapping $\phi : [i] \to [N]$ s.t. $f(\phi^{-1}(\cdot))$ satisfies $\mathcal{I}(\phi([i]))$.*

We call a $p\mathcal{I}$-feasible polymatroid a $p\mathcal{I}$-polymatroid, while a $p\mathcal{I}$-polymatroid of size N is called a $\mathcal{I}$-polymatroid. If $\mathcal{I}$ arises from a HMSNC instance, and $([N], f)$ is a matroid, then ϕ^{-1} in above definition is known as the *network-matroid* mapping [DFZ07]. We shall denote the injective map associated with a $p\mathcal{I}$-polymatroid as the $p\mathcal{I}$-map and the bijection associated with an $\mathcal{I}$-polymatroid

as the $\mathcal{I}$-map. Note that the set of all bijections from $[N]$ to itself, denoted as Ω has size $N!$ while set of all maps from $[i]$ to $\{j_1, \ldots, j_i\} \subseteq [N]$ for $i \leq N$ denoted as Ω_p contains $\sum_{k=0}^{N-1} \binom{N}{k}(N-k)!$ maps with $\Omega \subseteq \Omega_p$. Fig. 2.1 and Fig.2.2 give examples of an $\mathcal{I}$-matroid and an $\mathcal{I}$-polymatroid respectively.

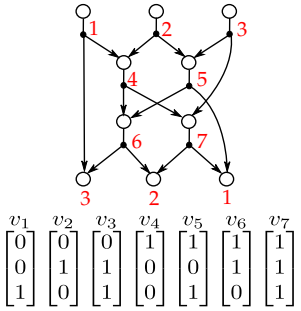

Figure 2.1: (top) the HMSNC instance Fano Network, and (bottom) a representation of the Fano matroid that is an $\mathcal{I}$-(poly)matroid with mapping ϕ defined as $\{1 \mapsto 1, 2 \mapsto 2, 3 \mapsto 4, 4 \mapsto 3, 5 \mapsto 6, 6 \mapsto 5, 7 \mapsto 7\}$.

$$\left\{ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \right\}$$
$$V_1 \quad V_2 \quad V_3 \quad V_4 \quad V_5$$

Figure 2.2: An $\mathcal{I}$-polymatroid for the collection of constraints $\mathcal{I}$ arising from rank vector $[2, 2, 4, 2, 4, 4, 4, 1, 2, 3, 4, 3, 4, 4, 4, 1, 3, 3, 4, 3, 4, 4, 4, 2, 3, 4, 4, 3, 4, 4, 4]$ with mapping ϕ defined as $\{1 \mapsto 4, 2 \mapsto 5, 3 \mapsto 1, 4 \mapsto 3, 5 \mapsto 2\}$. This polymatroid is an extreme ray of the cone of linear rank inequalities in 5 variables [DFZ09].

Thus, each $p\mathcal{I}$-polymatroid is a pair (P, ϕ) where P is a polymatroid representation and ϕ is a p-map. The combinatorial objects we want to systematically generate are $p\mathcal{I}$-polymatroids belonging to a particular class of codes $\mathscr{P}^q(\mathbf{c}), \mathbf{c} = (N, (r_l, r_u), K, (s_l, s_u))$ up to an equivalence relation $\equiv$. From orderly generation perspective, each of these has size N. The smaller objects of size $i < N$ belong to different classes of linear codes $\mathscr{P}^q(\mathbf{c}_i)$ where $\mathbf{c}_i = (i, (r_l, r_u), K, (\max(s_l, i), \min(i, s_u)))$.

2.3.2 Notions of polymatroid isomorphism and the equivalence relation $\equiv$

The most natural notion of isomorphism for polymatroids is the following one.

Definition 11 (Strong Isomorphism)**.** *Two polymatroids $P_1 = (E_1, f_2)$ and $P_2 = (E_2, f_2)$ are said*

to be isomorphic if there exists a bijection $\phi : E_1 \to E_2$ *s.t.* $f_1(S) = f_2(\phi(\mathfrak{S})), \forall \mathfrak{S} \subseteq E_1$*, denoted as* $P_1 \cong P_2$*.*

Given two polymatroids P_1 and P_2, the problem of determining if they are strongly isomorphic has received attention in the literature only in the special case where they are either representable matroids or graphic matroids. Testing if two matroids representable over $\mathbb{F}_q$ are strongly isomorphic is known to be no easier than the graph isomorphism problem [PR97], whose complexity status remains unresolved. There exist necessary conditions for isomorphism, that can be checked in polynomial-time, e.g. for binary matroids [Pen14]. The problem is somewhat easier if we know beforehand that $\mathsf{us}(P_1) = \mathsf{us}(P_2)$. Here, we can use the degree vector, as defined below.

Definition 12. *The degree vector* $(d_1, \ldots, d_{|\mathsf{us}(P)|+1})$ *of a polymatroid* $P = (E', f')$ *with a specified order on* E' *is an integer vector of size* $|E| + 1$ *where* $(E, f) = \mathsf{us}(P)$ *with the* i*th entry of the degree vector indicating the size of parallel class of the* i*th smallest non-loop element of* E *in* P *for* $i \in [|E|]$*, whereas the* $|E| + 1$*th element specifies the loop degree.*

An automorphism of a polymatroid is a strong isomorphism from a polymatroid to itself.

Lemma 2. *Let* $P_1 = (E_1, f_1)$ *and* $P_2 = (E_2, f_2)$ *be two polymatroids s.t.* $|E_1| = |E_2|$ *and* $\mathsf{us}(P_1) = \mathsf{us}(P_2)$*. Then* P_1 *and* P_2 *are strongly isomorphic if and only if their associated degree vectors are identical up to an automorphism of* $\mathsf{us}(P)$*.*

To introduce further notions of isomorphism, we first describe the group of semi-linear isometries of $\mathbb{F}_q^r$ and describe its action on the set of all distinct representations of simple linear polymatroids of specified size and rank. Then we explain how this action results in a equivalence relation that is a weakening of the notion of isomorphism in definition 11. We essentially generalize the notion of semilinear isometry among the generator matrices of projective linear codes over $\mathbb{F}_q$ [BBF+06] to that of weak isomorphism among the distinct representations of simple linear polymatroids.

Definition 13. *The mapping* $\sigma : \mathbb{F}_q^r \to \mathbb{F}_q^r$ *is called semilinear if there exists an automorphism* α *of*

$\mathbb{F}_q$ such that for all $\mathbf{u}, \mathbf{v} \in \mathbb{F}_q^r$ and all $x \in \mathbb{F}_q$, we have

$$\sigma(\mathbf{u}+\mathbf{v}) = \sigma(\mathbf{u}) + \sigma(\mathbf{v}), \;\; \sigma(x\mathbf{u}) = \alpha(x)\sigma(\mathbf{u}) \tag{2.8}$$

A semilinear isometry is a semilinear mapping that maps subspaces to subspaces. The set of all semilinear isometries forms a group known as the general semilinear group. Let $GL(r,q)$ be the general linear group containing all $r \times r$ invertible matrices over $\mathbb{F}_q$ and let $\mathsf{Gal}(q)$ be the Galois group of $\mathbb{F}_q$. $\mathsf{Gal}(q)$ is a cyclic group of order t where $q = p^t$ for a prime p, generated by the Frobenius automorphism $x \mapsto x^p$. Then the general semilinear group can be defined as follows.

Definition 14. *The semilinear isometries of $\mathbb{F}_q^N$ form the general semilinear group,*

$$\Gamma L(r,q) \triangleq \{(\mathbb{A}, \alpha) \mid \mathbb{A} \in GL(r,q), \alpha \in \mathit{Gal}(q)\} \tag{2.9}$$

The general semi-linear group $\Gamma L(r,q)$ is the semidirect product $GL(r,q) \rtimes \mathit{Gal}(q)$. The identity element is the pair $(\mathbb{I}_r, \mathsf{id})$, where id is the identity element in $\mathit{Gal}(q)$. The multiplication of two elements of $\Gamma L(r,q)$ is given by,

$$(\mathbb{A}_2, \alpha_2)(\mathbb{A}_1, \alpha_1) = (\mathbb{A} \cdot \alpha_2(\mathbb{A}_1), \alpha_2\alpha_1), \; \forall (\mathbb{A}_1, \alpha_1), (\mathbb{A}_2, \alpha_2) \in \Gamma L(r,q) \tag{2.10}$$

$\Gamma L(r,q)$ acts naturally on $\mathsf{Gr}_q(r,k)$ as follows

$$\Gamma L(r,q) \times \mathsf{Gr}_q(r,k) \to \mathsf{Gr}_q(r,k) : ((\mathbb{A}, \alpha), \langle \mathbf{v}_1, \ldots, \mathbf{v}_k \rangle) \mapsto \langle \alpha(\mathbf{v}_1)\mathbb{A}^t, \ldots, \alpha(\mathbf{v}_k)\mathbb{A}^t \rangle \tag{2.11}$$

The subgroup of $\Gamma L(r,q)$ that stabilizes each element of $\mathsf{Gr}_q(r,k)$ pointwise is $\{(\mathbb{A}, \mathsf{id}) \mid \mathbb{A} \in \mathcal{Z}_r\}$ where $\mathcal{Z}_r$ is the center of $GL(r,q)$. This stabilizer is isomorphic to $\mathcal{Z}_r$ itself. Hence, $\Gamma L(r,q)$ induces the action of *projective semilinear group* given as,

$$P\Gamma L(r,q) \triangleq \Gamma L(r,q)/\mathcal{Z}_r. \tag{2.12}$$

on $\mathsf{Gr}_q(r,k)$. The elements of $P\Gamma L(r,q)$ are of the form $(\mathbb{A}\mathcal{Z}_r,\alpha)$ for $\mathbb{A}\in GL(r,q)$. Furthermore, it can be expressed as the semidirect product

$$P\Gamma L(r,q) = PGL(r,q) \rtimes \mathsf{Gal}(q). \tag{2.13}$$

The identity element is $(\mathbb{I}_r\mathcal{Z}_r,\mathsf{id})$ where id is the identity element in $\mathsf{Gal}(q)$. The multiplication of two elements in $P\Gamma L(r,q)$ is given as

$$(\mathbb{A}_2\mathcal{Z}_r,\alpha_2)(\mathbb{A}_1\mathcal{Z}_r,\alpha_1) = ((\mathbb{A}_2\cdot\alpha_2(\mathbb{A}_1))\mathcal{Z}_r,\alpha_2\alpha_1) \tag{2.14}$$

The inverse element of $(\mathbb{A}\mathcal{Z}_r,\alpha)$ is $(\alpha^{-1}(\mathbb{A}))^{-1}\mathcal{Z}_r,\alpha^{-1}$. The action of $P\Gamma L(r,q)$ on $\mathsf{Gr}_q(r,k)$ is as follows

$$\gamma_k : P\Gamma L(r,q)\times\mathsf{Gr}_q(r,k)\to\mathsf{Gr}_q(r,k) : ((\mathbb{A}\mathcal{Z}_r,\alpha),\langle\mathbf{v}_1,\ldots,\mathbf{v}_k\rangle)\mapsto\langle\alpha(\mathbf{v}_1)\mathbb{A}^t\mathcal{Z}_r,\ldots,\alpha(\mathbf{v}_k)\mathbb{A}^t\mathcal{Z}_r\rangle \tag{2.15}$$

Let $k(V)$ denote the dimension of a subspace V. The above action can be naturally extended to an action on $\mathsf{Gr}_q(r,K)$ which further induces an action on $2^{\mathsf{Gr}_q(r,K)}$ as,

$$\gamma' : P\Gamma L(r,q)\times 2^{\mathsf{Gr}_q(r,K)}\to 2^{\mathsf{Gr}_q(r,K)} : \{V_1,\ldots,V_i\}\mapsto\{\gamma_{k(V_1)}(V_1),\ldots,\gamma_{k(V_i)}(V_i)\} \tag{2.16}$$

for any subset of size i of $\mathsf{Gr}_q(r,K)$. We shorten $\gamma'(g,\{V_1,\ldots,V_i\})$ as $\{V_1,\ldots,V_i\}g$, for $g\in P\Gamma L(r,q)$. Given a collection $K\subseteq[N]$ of distinct singleton ranks, let $\mathscr{S}^q(N,r,K)$ be the set of polymatroid representations in $\mathscr{P}^q(N,r,K,s)$ (described in §2.1) with $s=N$ i.e. the representations associated with simple polymatroids. One can easily verify that this set is fixed setwise under the above action, as parameters N,r,K remain unchanged. We now define the notion of weak isomorphism.

Definition 15. *(Weak Isomorphism) Two representations* $P_1=\{V_1^1,\ldots V_N^1\},P_2=\{V_1^2,\ldots,V_N^2\}\in$

$\mathscr{S}^q_{N,r,K}$ are said to be weakly isomorphic if there exists a $g \in P\Gamma L(r,q)$ s.t.

$$\{V_1^1, \dots V_N^1\}g = \{V_1^2, \dots, V_N^2\} \tag{2.17}$$

denoted as $P_1 \overset{W}{=} P_2$.

Lemma 3. *Let $P_1, P_2 \in \mathscr{S}^q(N,r,K)$ be the representations of two simple $\mathbb{F}_q$-representable polymatroids s.t. $P_1 \overset{W}{=} P_2$. Then, $P_1 \cong P_2$.*

The opposite of the above statement, however, is not true. It is possible for even the same $\mathbb{F}_q$-representable polymatroid can have several representations that are weakly non-isomorphic. The following example illustrates this phenomenon.

Example 1. Consider the following representations $P_1, P_2 \in \mathscr{S}^2_{5,5,\{2\}}$

$$P_1 \triangleq \left\{ \begin{matrix} V_1 \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \end{matrix}, \begin{matrix} V_2 \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \end{matrix}, \begin{matrix} V_3 \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix} \end{matrix}, \begin{matrix} V_4 \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix} \end{matrix}, \begin{matrix} V_5 \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \end{matrix} \right\} \tag{2.18}$$

$$P_2 \triangleq \left\{ \begin{matrix} V_1' \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \end{matrix}, \begin{matrix} V_2' \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \end{matrix}, \begin{matrix} V_3' \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix} \end{matrix}, \begin{matrix} V_4' \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \end{matrix}, \begin{matrix} V_5' \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \end{matrix} \right\} \tag{2.19}$$

One can computationally verify that every isomorphism between P_1 and P_2 must map V_5 to V_5' i.e.

fix V_5. The subgroup of $PGL(5,2)$ that stabilizes V_5 setwise contains matrices of the form $\begin{bmatrix} A & \mathbb{O} \\ \mathbb{O} & B \end{bmatrix}$ where $A \in PGL(2,2)$ and $B \in PGL(3,2)$. Hence, for P_1 and P_2 to be weakly isomorphic, we must have $P_1' \overset{W}{=} P_2'$ (eq. (2.20),(2.21)) i.e. there must exist a $B \in PGL(3,2)$ that maps P_1' to P_2'.

$$P_1' \triangleq \left\{ \begin{matrix} W_1 \\ \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \end{matrix}, \begin{matrix} W_2 \\ \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \end{matrix}, \begin{matrix} W_3 \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix} \end{matrix}, \begin{matrix} W_4 \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix} \end{matrix} \right\} \tag{2.20}$$

$$P_2' \triangleq \left\{ \begin{matrix} W_1' \\ \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \end{matrix}, \begin{matrix} W_2' \\ \begin{bmatrix} 0 & 1 \\ 0 & 0 \\ 1 & 0 \end{bmatrix} \end{matrix}, \begin{matrix} W_3' \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \end{bmatrix} \end{matrix}, \begin{matrix} W_4' \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{bmatrix} \end{matrix} \right\} \tag{2.21}$$

If such a B exists, then $W_4^t B = W_i'^t$ for some $i \in [4]$. However, the row reduced echelon form of W_4^t differs from those of $W_1'^t, \ldots, W_4'^t$, contradicting the fact that $B \in PGL(3,2)$. Hence $P_1 \overset{W}{\neq} P_2$.

Because of the difficulty of isomorphism problems for graphs and their generalizations, some of methods to generate such objects are designed to outright avoid explicit isomorphism testing. The choice of isomorphism relation could be motivated by this issue, as we discuss in §2.4.2.

2.3.3 Polymatroid extension and construction of all members of a class of codes

Polymatroid extension plays a central role in algorithms for exhaustive generation of polymatroids. It provides a means to construct polymatroids on ground set size $i+1$ from polymatroids on ground set size i. The following is a general definition of polymatroid extension which applies to arbitrary (not necessarily integer or representable) polymatroids.

Definition 16. *A polymatroid (E', f') is said to be an extension of polymatroid (E, f) if $E \subseteq E'$ and $f'(\mathfrak{S}) = f(\mathfrak{S})\ \forall \mathfrak{S} \subseteq E$. Furthermore, if $|E' \setminus E| = t$ then (E', f') is said to be a t-extension of*

(E, f).

If t is 1 and $E' \setminus E = \{e\}$, then we say that (E', f') is a 1-extension of (E, f) by an element e. The theory of unique 1-extensions for the special case of matroids was developed by Crapo [Cra65] where he provided a method to construct all unique 1-extensions of a matroid. This method, along with explicit isomorphism testing was used by Blackburn et al. [BCH73] to construct all non-isomorphic matroids on up to 8 elements. More recently Mayhew and Royle [MR08] constructed all matroids on 9 elements using the same method. Matsumoto et al. [MMIB12] extended Crapo's theory to avoid constructing isomorphic single element extensions and partially constructed matroids on 10 elements. Savitsky [Sav14] generalized Crapo's method of producing unique 1-extensions of matroids to integer k-polymatroids, which are integer polymatroids with the rank of every singleton bounded above by k and generated a catalog of 2-polymatroids on up to 7 elements. Unfortunately, techniques described by Crapo et. al. [Cra65] for matroids and Savitsky for [Sav14] for k-polymatroids are far too general for our purpose. Matroid extensions followed by representability checking style algorithm has been proposed in e.g. [ALW14], but has met with limited practical success for several reasons. Firstly, if (E, f) is a representable polymatroid, the extensions produced are not guaranteed to be representable, which means we must employ additional computation to find and weed out the ones that are non- representable. Secondly, these methods are only developed for polymatroids with specific sets of singleton ranks i.e. $K_P = \{0, 1\}$ in case of Crapo et al.'s work and $K_P = \{0, 1, \ldots, k\}$ for some $k \in \mathbb{N}$ in case of Savitsky's work. The third factor that discourages one from using such overly general techniques is the sheer rate at which number of general polymatroids or matroids grows as compared to their representable counterparts. For instance, in fig. 2.3 we can see the growth of number of all (non-isomorphic, integer valued) 2-polymatroids, all matroids, all ternary representable matroids and all binary representable matroids.

Bearing this in mind, we instead aim to enumerate representable polymatroids directly exclusively, and hence, we define a notion of extension for a polymatroid which we choose to use over the more general notion in def. 16, as it conveniently refers directly to the representation of the polymatroid.

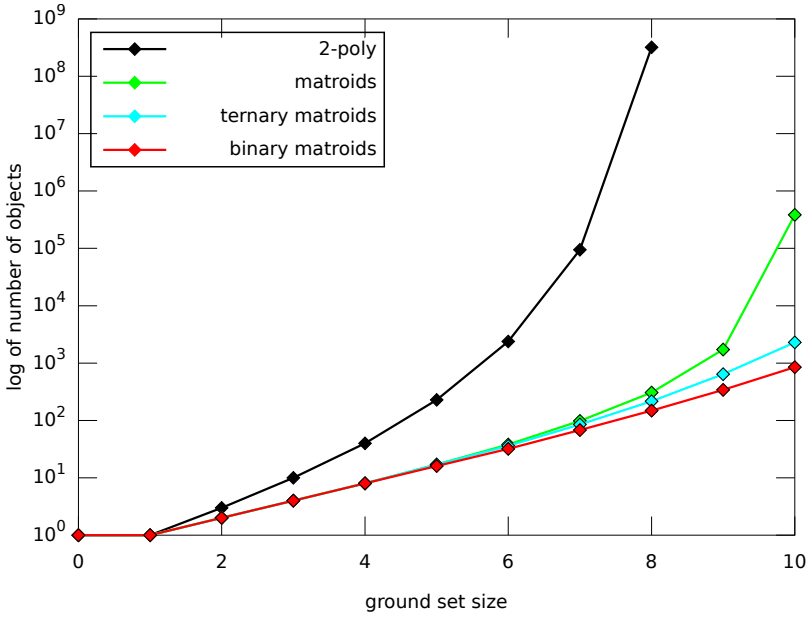

Figure 2.3: Log of number of all (non-isomorphic) 2-polymatroids [Max], matroids [Eri], ternary matroids [Marb], and binary matroids [Mara], plotted against the ground set size

Definition 17. *Let P be a polymatroid in $\mathscr{P}^q(i,(r,r),K,(0,i))$ represented as $\{V_1,\ldots,V_i\}$. Then, a 1-extension of P is any polymatroid P' represeted as $\{V_1,\ldots,V_i,V_{i+1}\}$ where $V_{i+1} \in \mathsf{Gr}_q(r,K)$.*

Note that the above definition augments a polymatroid made up of a multiset of subspaces of $\mathbb{F}_q^r$ by adding another subspace of $\mathbb{F}_q^r$ to the multiset, thus keeping the underlying vector space dimension r as a constant. We further classify 1-extensions into two types: simple 1-extensions and a non-simple 1-extensions. A 1-extension is simple if $|\mathsf{us}(P')| = |\mathsf{us}(P)|+1$ and non-simple otherwise. For an extension to be simple, V_{i+1} must be distinct from each of $V_1,\ldots,V_i$. On the other hand, an extension is non-simple if and only if V_{i+1} is a copy of one of $V_1,\ldots,V_i$ or it is an empty subspace with $f(i+1)=0$.

Now we describe our basic strategy for constructing all polymatroids in $\mathscr{P}^q(N,(r_l,r_u),K,(s_l,s_u))$ up to some notion of equivalence $\equiv$. A basic assumption about $\equiv$ is that for polymatroids belonging to the same equivalence class under $\equiv$, the parameters N,r,K and $|\mathsf{us}(P)|$ are identical, which is indeed the case with both strong and weak isomorphism relations discussed in 2.3.2. We also assume that we have access to procedures $\mathsf{se}(\cdot)$ and $\mathsf{nse}(\cdot)$ which take as input a list of $\equiv$-inequivalent

polymatroid representations and produce a list of $\equiv$-inequivalent simple and non-simple 1-extensions respectively. We defer the discussion of low-level details of how these procedures work in practice to the next section, where we point out several techniques in literature that can be used to implement such procedures. Fig. 2.4 describes how one can use procedures $\mathsf{se}(\cdot)$ and $\mathsf{nse}(\cdot)$ to construct all mem-

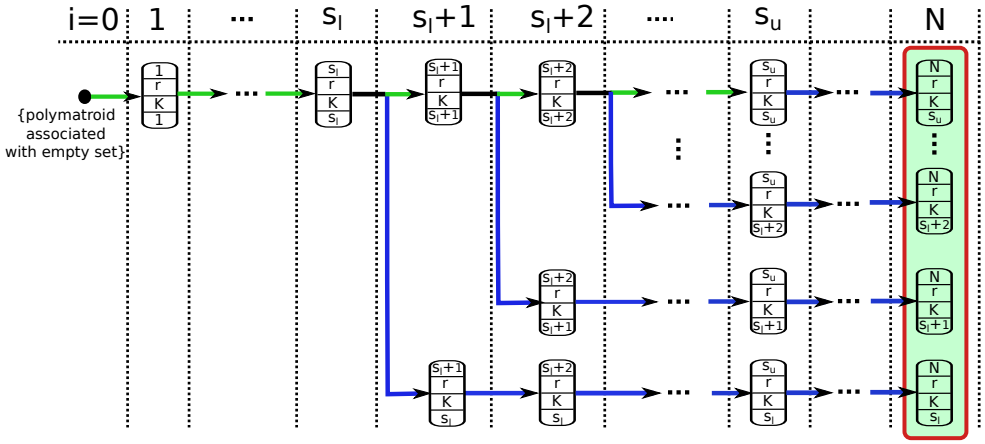

Figure 2.4: Construction of $\equiv$-inequivalent polymatroid representations belonging to the class $\mathscr{P}^q(N,(r_l,r_u),K,(s_l,s_u))$, for some $r_l \leq r \leq r_u$. The green arrows indicate the use of simple extensions $\mathsf{se}(\cdot)$ while blue arrows indicate the use of non-simple extensions $\mathsf{nse}(\cdot)$. Each box itself corresponds to a particular class of codes.The parameters specified from top to bottom are: (1) size or length i, (2) dimension r of vector space over $\mathbb{F}_q$ whose subspaces are used to build each polymatroid in the class, (3) set K of distinct singleton ranks, and (4) $|\mathsf{us}(P)|$ for each polymatroid P in the class.

bers of $\mathscr{P}^q(N,(r,r),K,(s_l,s_u))$ up to an equivalence relation $\equiv$. This strategy can be used repeatedly for different values of r s.t. $r_l \leq r \leq r_u$ to construct all members of $\mathscr{P}^q(N,(r_l,r_u),K,(s_l,s_u))$ up to $\equiv$. While our goal in this chapter is to solve variants of CLRPq, the strategy described in Fig. 2.4 can be used to construct inner bounds $\Gamma_N^{\mathscr{P}^q(\mathbf{c})}$, which might be of interest in their own right.

2.3.4 An augmentation operation for $p\mathcal{I}$-polymatroids

We now describe the construction of all $p\mathcal{I}$-polymatroids of size $i+1$ from non-isomorphic $p\mathcal{I}$-polymatroids of size $i < N$ by combining linear 1-extension of $p\mathcal{I}$-polymatroids and extension of the associated partial maps. In the background setting provided by the strategy described in Fig. 2.4, to generate linear $p\mathcal{I}$-polymatroids, the basic problem we must be able to solve is that of determining if a given polymatroid is a $p\mathcal{I}$-polymatroid:

- [X1] For a collection of constraints $\mathcal{I}$ of size N, given a polymatroid $P = ([i], f)$, $i \leq N$, determine if P is a $p\mathcal{I}$-polymatroid.

In order to settle [X1], we must look for a p-map ϕ under which P satisfies $\mathcal{I}$. If we are successful in finding a certificate p-map ϕ showing that P is a $p\mathcal{I}$-polymatroid, we decide to keep P. Otherwise we reject P' and all its t-extensions from contention.

Problem [X1] can be solved by using a backtracking approach. To see how, consider set of all possible p-maps Ω_p, ordered using the lexicographic order described as follows. Let δ be the map $\{1 \mapsto i_1, \ldots, |\delta| \mapsto i_{|\delta|}\}$ and γ be the map $\{1 \mapsto j_1, \ldots, |\gamma| \mapsto j_{|\gamma|}\}$ where $|\delta|$ and $|\gamma|$ denote the size of the domain of δ and γ respectively with $i_k \in [N], \forall k \in [|\delta|]$ and $j_k \in [N], \forall k \in [|\gamma|]$. Denote the ordinary lexicographic (dictionary) order on tuples of length $\leq N$ with elements from $[N]$ as $\overset{L}{<}$, then this ordering is directly extendable to partial maps as follows.

Definition 18. *Let $\delta, \gamma \in \Omega_p$ be distinct $p\mathcal{I}$-maps. Then δ is smaller than γ if $(i_t)_{t=1}^{|\delta|} \overset{L}{<} (j_t)_{t=1}^{|\gamma|}$.*

We now define the p-map tree which is a directed tree whose vertices are p-maps and the edges are defined using the notion of an *extension* of a p-map.

Definition 19. *A p-map $\delta' : [i+1] \to [N]$ is said to be an extension of a p-map $\delta : [i] \to [N]$ if $i < N$ and $\delta(k) = \delta'(k)$, $\forall k \in [i]$.*

We say that the p-map δ in the above definition is a deletion of p-map δ'. The p-map tree of order N is a directed graph $T_N = (V_N, E_N)$ where V_n is the set of all p-maps and $(u, v) \in E$ if v is an extension of u. Note that p-map extension and deletion provide a means to traverse the p-map tree in lexicographic order. Furthermore, given a vertex u (a p-map) in the tree, one can resume the traversal to produce all p-maps that are lexicographically greater than u. Given a p-map $\delta = \langle i_1, \ldots, i_k \rangle$ at depth $k < N$ in the p-map tree T_N, its immediate descendents can be computed as $\langle i_1, \ldots, i_k, j \rangle$ where $j \in [N] \setminus \{i_1, \ldots, i_k\}$ and can be visited in lexicographical order. The parent of δ can be obtained by deleting i_k. Thus, we need not explicitly store the p-map tree in order to traverse it.

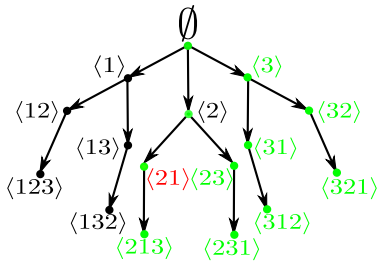

Figure 2.5: The p-map tree T_3 with the subtree $T_3^{>\langle 21\rangle}$. For a collection of constraints $\mathcal{I}$ of size 3, let a size 2 polymatroid (E, f) be a $p\mathcal{I}$-polymatroid with $p\mathcal{I}$-map $\{1 \mapsto 2, 2 \mapsto 1\}$ (shown in red), and let (E', f') be its 1-extension. Then we need only traverse the vertices shown in green in worst case to determine if (E', f') is also a $p\mathcal{I}$-polymatroid.

One merit of considering the lexicographic ordering of p-maps and thinking of the collection of all such p-maps as a tree ordered under this ordering is that it allows the problem of finding a p-map, or lack of existence thereof, to be posed as a tree search algorithm. One way to solve [X1] is to traverse T_N in a depth first manner to depth i, while testing constraints $\mathcal{I}(\delta([j+1])) - \mathcal{I}(\delta([j]))$ at each node associated with a p-map δ at depth j, $j \leq i$. In this instance, what is meant by depth first is that one extends a partial map adding mapped elements one by one until is it no longer possible to satisfy the constraints $\mathcal{I}$ using it, i.e., until all 1 extensions of the partial map no longer obey the constraints. At that time, a backtracking step deletes the most recent extension, and the partial map extension process continues. If ever a complete traversal to depth i succeeds, then a p-map of size i has been found, and provides a certificate showing that [X1] has been solved and that $P = ([i], f)$ is a $p\mathcal{I}$-polymatroid. Conversely, if the entire tree is exhausted, i.e. the algorithm terminates with no more backtracks or extensions possible, then [X1] has been answered in the negative. Of course, our primary interest in this manuscript is the construction of polymatroids with rank functions obeying a series of linear constraints through an extension process i.e. we would like be able to use the strategy in Fig. 2.4 while at the same time maintaining only those polymatroids that are $p\mathcal{I}$-feasible. The tree structure of the set of p-maps is also beneficial in this extension process. In particular, consider a $p\mathcal{I}$-polymatroid $P = ([i], f)$ of size $i < N$ with a p-map ϕ, and let $P' = ([i+1], f)$ be a 1-extension of P. Then, the $p\mathcal{I}$-polymatroid extension process must solve the following problem at each step.

- [X2] Determine if $P' = ([i+1], f)$, the 1-extension of a $p\mathcal{I}$-polymatroid $P = ([i], f)$ with lexicographically minimum p-map ϕ, is a $p\mathcal{I}$-polymatroid, and if so, find its lexicographically minimum p-map.

That is, we must determine if there exists a p-map ϕ' for P' s.t. it is a $p\mathcal{I}$-polymatroid. Given ϕ, we can use the following lemma to traverse only a subtree of T_N for determining if P' is a $p\mathcal{I}$-polymatroid.

Lemma 4. *Let P be a $p\mathcal{I}$-polymatroid with ϕ being the lexicographically smallest $p-$map associated with it and let P' be a 1-extension of P. If P' is a $p\mathcal{I}$-polymatroid with ϕ' being the p-map associated with it, then $\phi' \overset{L}{>} \phi$.*

This lemma shows that we can determine whether the polymatroid 1-extension P' is a $p\mathcal{I}$-polymatroid by traversing the p-map tree in depth-first fashion, resuming the tree traversal from ϕ. Let $V_i^{>\delta}$ be the set of all p-maps that are lexicographically greater than δ and $\hat{V}_i^{\delta}$ be the set of all ancestors of δ. Denote by $T_i(\delta)$ to be the subgraph of T_i induced by vertices $V_i^{>} \cup \hat{V}_i^{\delta}$ and the vertices. Hence, it suffices to traverse $T_i(\delta)$ to settle [X2].

2.3.5 Exploiting symmetry when augmenting $p\mathcal{I}$-polymatroids

In many instances, depending on the low level techniques used for implementing procedures $\mathsf{se}(\cdot)$ and $\mathsf{nse}(\cdot)$, either a symmetry group of the polymatroid $P = ([i], f)$ at the current step of the polymatroid extension process, or a symmetry group of the constraints $\mathcal{I}$ or both might be known. Knowledge of these groups can further reduce the amount of computation required in searching for p-maps in problem [X2]. In particular, we can assume that the symmetries of $P = ([i], f)$ are specified as a group $A \leq S_i$ where S_i is the group of all permutations of $[i]$: so that for each permutation $a \in A$ and for each set $\mathscr{E} \subseteq [i]$, $f(a(\mathscr{E})) = f(\mathscr{E})$. Similarly, the symmetries of $\mathcal{I}$ are provided as a group $B \leq S_N$ where S_N is the group of all permutations of $[N]$: for each element $b \in B$, if $([N], f')$ is $p\mathcal{I}$-polymatroid, then so is $([N], f' \circ b)$. Together, as each putative partial map for $P = ([i], f)$ is an injective map $\phi : [i] \to [N]$, the direct product $A \times B$ acts on a putative partial map ϕ via $((a, b), \phi) \mapsto b(\phi(a(\cdot)))$. When attempting to search for a p-map, then, one only needs to consider

single representatives from the equivalence class created on the p-maps under this group action.

Formally, the associated problem can be stated as follows.

- [X3] Given symmetry groups A, B determine if P is a $p\mathcal{I}$-polymatroid.

At depth $j \leq i$ in T_N, denote by $A_{[j]}$ the subgroup of A that stabilizes $[j]$ set-wise. We now consider the action of the direct product $G_{[j]} \triangleq A_{[j]} \times B \leq A \times B$ on V_j which is the set of vertices of T_N at depth $j \leq i$. The composition in $G_{[j]}$ is denoted as $((a_1, b_1) * (a_2, b_2)) = (a_1 a_2, b_2, b_2)$. One can see that $((a_1 a_2, b_2, b_2)\delta)(x) = b_1 b_2 \delta(x(a_1 a_2)) = b_1(b_2 \delta((x a_1) a_2)) = b_1((a_2, b_2) f)(x a_1) = (a_1, b_2)(((a_2, b_2) f)(x))$ for any $x \in [j], j \leq i$ where $[j]$ is the domain of p-map δ. Let V_j^* be the transversal of orbits in V_j under the aforementioned action formed by choosing the lexicographically smallest p-map from each orbit. Let T_i^* be the subgraph of T_i induced by $\cup_{j \leq i} V_j^*$.

Lemma 5. *It suffices to traverse T_i^* instead of T_i to determine if $P = ([i], f)$ is a $p\mathcal{I}$-polymatroid.*

Example 2. Consider the HMSNC instance shown in fig. 2.6. Let the associated constraints be $\mathcal{I}$. This network has a symmetry group of order 2 generated by permutations {(3,4)}. The polymatroid $\{V_1, V_2, V_3\}$ in (2.22) is a $p\mathcal{I}$-polymatroid obtained via extending $p\mathcal{I}$-polymatroids $P_1 = \{V_1\}$ and $P_2 = \{V_1, V_2\}$, in that order. Fig. 2.7 shows how the knowledge of network symmetry group, along with symmetries of the polymatroids P_i canbe used to traverse only a subset of the verticces of the p-map tree when determining their $p\mathcal{I}$ feasibility.

$$P_3 \triangleq \left\{ \begin{matrix} V_1 \\ \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \end{matrix}, \begin{matrix} V_2 \\ \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \end{matrix}, \begin{matrix} V_3 \\ \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \end{matrix} \right\} \tag{2.22}$$

Problems [X1]-[X3] can be combined to form problem [X4] below:

- [X4] Determine if $P' = ([i+1], f)$ is a $p\mathcal{I}$-polymatroid given symmetry groups A, B and the p-map ϕ associated with $p\mathcal{I}$-polymatroid $P = ([i], f)$ where P' is a 1-extension of P.

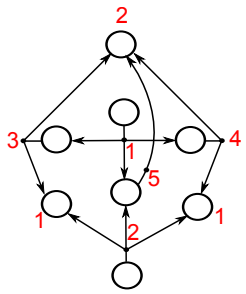

Figure 2.6: A HMSNC instance with $N = 5$. The symmetry group A of this instance is of order 2 generated by $\{(3,4)\}$.

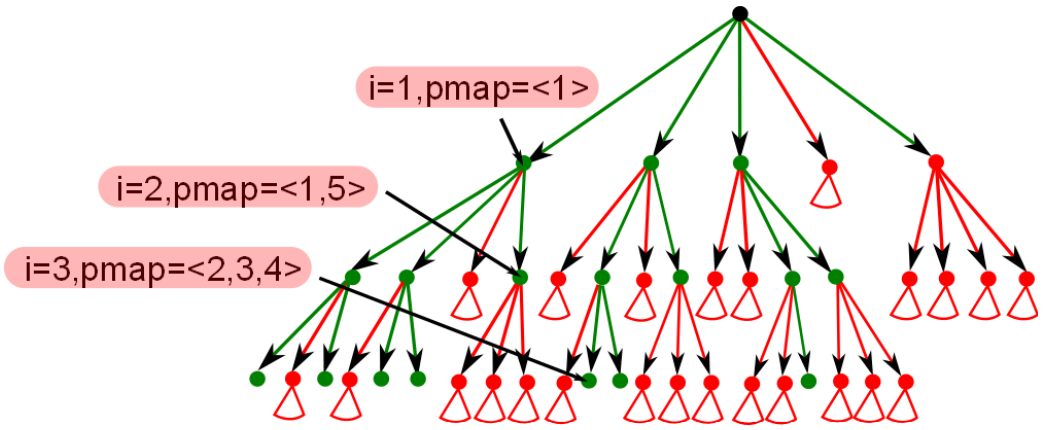

Figure 2.7: The nested p-map trees T_i, $i \leq 3$ for HMSNC instance in fig. 2.6. The subtrees T_i^* are shown in green. Each T_i^* is the subgraph of T_i induced by vertices associated p-maps that are with lexicographically smallest in their respective orbits under the the action of $A \times B$ where B is the trivial group for $i = 1$, and is generated by $\{(1,2)\}$,$\{(1,2),(1,3,2)\}$ for $i = 2,3$ respectively. The vertices at every level are drawn in lexicographic order (ascending from left to right).

The above problem can be solved by combining the approaches to [X1]-[X3] i.e. it suffices to traverse $T^*_{i+1}(\phi)$, where, using induction on i, we assume that ϕ is part of T^*_{i+1}.

The functionality of p-map extension, for generation of $\mathcal{I}$-polymatroids, is provided by the procedure extend_pmap(P', pc, G) that accepts a 1-extension P' of a $p\mathcal{I}$-polymatroid P, the p-map pc and a group G of symmetries of $\mathsf{us}(P')$. Note that symmetries of P' can be directly deduced from those of $\mathsf{us}(P')$. If P' is indeed a $p\mathcal{I}$-polymatroid, then it returns a p-map c associated with P' s.t. $\mathsf{c} \overset{L}{>} \mathsf{pc}$ which serves as a certificate. Otherwise, it returns empty p-map, denoted as ϕ_{null}. Note that p-map c produced in this manner will itself be the lexicographically smallest such map associated

with P'.

2.4 An algorithm for solving CLRP$_q$-EN

In this section, we build on various concepts described in previous section, to provide the high level description of an algorithm to solve CLRP$_q$-EN via exhaustive generation of $p\mathcal{I}$-polymatroids up to equivalence relation $\equiv$. The description is generic enough so that $\equiv$ can be interpreted as either strong or weak isomorphism. We also discuss how procedures $\mathsf{se}(\cdot)$ and $\mathsf{nse}(\cdot)$ could be implemented in §2.4.2.

2.4.1 High-level description of the algorithm

In §2.3.3 we discussed how to generate all members of a class of codes $\mathscr{P}^q(\mathbf{c})$ up to an equivalence relation $\equiv$. We intend to use fig. 2.4 as a template for generating $p\mathcal{I}$-polymatroids. To that end, we establish some facts about the property of $p\mathcal{I}$-feasibility, that allow us to restrict the strategy in fig 2.4 to $p\mathcal{I}$-polymatroids only. The two main facts that help us achieve this are the *inheritedness* and *isomorph-invariance* of the property of $p\mathcal{I}$-feasibility.

Definition 20. *A property χ of polymatroids is said to be inherited if a polymatroid P has χ then all polymatroids obtained from $\mathcal{P}$ via deletion of ground set elements also have χ.*

Let P be a $p\mathcal{I}$-polymatroid with associated p-map ϕ. One can form a p-map for polymatroid P' obtained by deletion from P by simply deleting mappings associated with deleted ground set elements. Such a p-map serves as a certificate of $p\mathcal{I}$-feasibility of P', thus showing that property of $p\mathcal{I}$-feasibility is inherited. As an implication, every $p\mathcal{I}$polymatroid of size i can be obtained via polymatroid extension and p-map extension from some $p\mathcal{I}$-polymatroid on ground set of size $i-1$.

Definition 21. *A property χ of polymatroids is said to be $\equiv$-invariant wrt equivalence relation $\equiv$, if a polymatroid P has χ, then all polymatroids equivalent to it under χ also have χ.*

For both equivalence relations $\cong$ and $\overset{W}{=}$ discussed in this work, members of the same equivalence class have the same rank function up to a permutation of the ground set. Hence, $p\mathcal{I}$-feasibility is both $\cong$-invariant and $\overset{W}{=}$-invariant, as given a certificate ϕ of $p\mathcal{I}$-feasibility of one member of the

equivalence class, we can construct such a certificate for every member of the said equivalence class, simply by applying an appropriate permutation to the domain of ϕ. Fig. 2.8 describes our algorithm

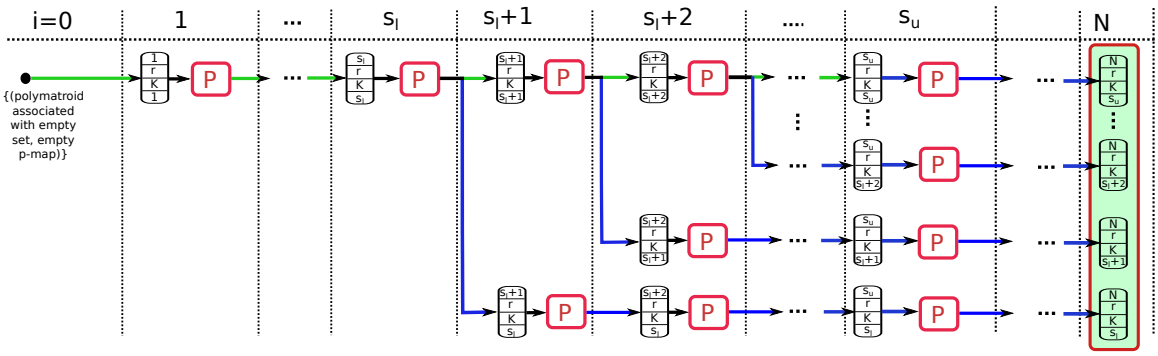

Figure 2.8: Construction of $\equiv$-inequivalent $p\mathcal{I}$-polymatroids belonging to the class $\mathscr{P}^q(N,(r_l,r_u),K,(s_l,s_u))$, for some $r_l \leq r \leq r_u$. The green arrows indicate the use of simple extensions $\mathsf{se}(\cdot)$ while blue arrows indicate the use of non-simple extensions $\mathsf{nse}(\cdot)$. Each box itself corresponds to polymatroids belonging a particular class of codes. The parameters specified from top to bottom are: 1) size or length i, dimension r of vector space over $\mathbb{F}_q$ whose subspaces are used to build each polymatroid in the class, 3) set K of distinct singleton ranks, and 4) $\mathsf{us}(P)$ for each polymatroid P in the class. The red box labeled P corresponds to procedure $\mathsf{extend\_pmap}(\cdot)$ that extends p-maps of the parents of polymatroids to find certificates of $p\mathcal{I}$-feasibility while filters out polymatroids for which no such extension exists.

to solve CLRP$_q$-EN. This is obtained essentially by adding the functionality of p-map extension to the strategy described in fig. 2.4. Any polymatroids that are found to be not $p\mathcal{I}$-feasible are filtered out, a convenient feature is made possible by inheriteness and isomorph-invariance of the property of $p\mathcal{I}$-feasibility.

2.4.2 Low-level details: simple and non-simple polymatroid extensions

This section is concerned with the implementation of procedures $\mathsf{se}(\cdot)$ and $\mathsf{nse}(\cdot)$. The two main aspects one must pay attention to are: a) the construction of *distinct* 1-extensions of a polymatroid and b) isomorph rejection. In case of $\mathsf{se}(\cdot)$, we are given a list of $\equiv$-inequivalent polymatroids belonging to a class $\mathscr{P}^q(i-1,(r,r),K,(s,s))$ and we want to construct a list of $\equiv$-inequivalent 1-extensions belonging to a class $\mathscr{P}^q(i,(r,r),K,(s+1,s+1))$. The exact techniques used will depend heavily on the choice of $\equiv$. There are techniques in literature that allow the construction of distinct 1-extensions linear extensions for $K=\{0,1\}$, i.e. the case of $\mathbb{F}_q$-representable matroids. These procedures use *chains* of a matroid and are implemented in sage-matroid package of SageMath

[PV+13]. Isomorph rejection via pairwise isomorphism testing is then employed to reject strongly isomorphic 1-extensions. Note that approach using the chains of a matroid can also be extended to procedure $\mathsf{nse}(\cdot)$.

An alternative approach, which allows arbitrary sets K of singleton ranks (and hence, to representably polymatroids as opposed to only representable matroids) is based on Leiterspiel, or the algorithm of snakes and ladders. This is a very general approach, designed for determining representatives of the orbits under action of a group G in the power set $2^{\mathscr{X}}$ of a set $\mathscr{X}$. It can be used for generating any combinatorial objects that can be described as subsets of a set up to an equivalence relation that arises from the orbits of a group action on the set in question. It was first described in a purely group-theoretic language by Schmalz [Sch90], whereas an interpretation more suitable to combinatorial generation can be found in [BBF+06]. Each member of $\mathscr{P}^q(i,(r,r),K,(s+1,s+1))$ can be described as an i-subset of $\mathsf{Gr}_q(r,K)$. Furthermore, weak isomorphism is in fact defined via a group action on $\mathsf{Gr}_q(r,K)$, as seen in §2.3.2. To authors' knowledge, weak isomorphism is maximal amoungst all equivalence relations that can be defined via a group action on $\mathsf{Gr}(q,K)$, in a terms of coarseness of the partition induced on subsets of size i of $\mathsf{Gr}_q(r,K)$. In exchange for weakness of the isomorphism relation, Leiterspiel allows us avoid explicit pairwise isomorphism testing along with providing access to subgroups of automorphism groups of polymatroids, which are constructed naturally in the process. These symmetries of polymatroids can be used in the determination of $p\mathcal{I}$-feasibility as described in §2.3.5. Owing to its generality, and other advantages, the implementation ITAP accompanying this thesis, uses Leiterspiel for the procedure $\mathsf{se}(\cdot)$.

Next, for implementing the procedure nse, one can first form all distinct $|\mathsf{us}(P)|$ non-simple 1-extensions of each polymatroid P in the input list and resort to pairwise isomorphism testing with respect to the chosen equivalence relation. Note that lemma 2, dictates how explicit strong isomorphism testing for two polymatroids P_1 and P_2 with $\mathsf{us}(P_1)=\mathsf{us}(P_2)$ can be performed. The statement of lemma 2 can be modified when one is interested in weak isomorphism testing, by substituting words 'weak' for 'strong' and by restring the meaning of automorphism group to be a subgroup of appropriate projective semilinear group. On the flipside, if we know beforehand that

$\mathsf{us}(P_1) \neq \mathsf{us}(P_2)$, we can directly conclude that $P_1 \ncong P_2$ (alternatively, $P_1 \overset{W}{\ncong} P_2$). This allows us to restrict the explicit isomorphism testing to only the polymatroids with the same underlying simple polymatroid. Note that if Leiterspiel is used to construct simple polymatroids (as is done in ITAP), we have access to the automorphism group w.r.t. weak isomorphism of every simple polymatroid we construct. This gives us a head start in the application of lemma 2 as we are already aware of the automorphism group of $\mathsf{us}(P_1)$ $(= \mathsf{us}(P_2))$ w.r.t weak isomorphism, while we have a subgroup of automorphism group w.r.t. strong isomorphism.

When we are solving CLRP$_q$-EN to compute achievable network coding rate regions, we can also use a hybrid approach of switching between isomorphism relations, which is implemented in ITAP. This approach uses Leiterspiel to perform simple extensions (green arrows in the first horizontal level in Fig. 2.8) and uses explicit isomorph testing w.r.t strong ismorphism relation when performing non-simple extensions (blue arrows in Fig. 2.8). In this case the automorphism groups provided by Leiterspiel can be used in explicit strong isomorphism testing.This hybrid technique ultimately answers CLRP$_q$-EN with the strong isomorphism (i.e. equality of polymatroid rank vectors) notion of equivalence, even though intermediate stages – the simple extensions – make use of weak isomorphism equivalence relations. This approach suffices for computing the achievable rate regions, because all rate vectors achievable with codes in $\mathscr{P}^q(\mathbf{c})$ can be obtained from the set of all strongly non-isomorphic $p\mathcal{I}$-polymatroids in $\mathscr{P}^q(\mathbf{c})$.

Algorithm 1 provides a detailed description of how to implement the general strategy in fig. 2.8. It closely matches our implementation in ITAP. We assume that we are given a collection of constraints $\mathcal{I}$, size of finite field q, and a class tuple $\mathbf{c} = (N, (r_l, r_u), K, (s_l, s_u))$. Algorithm 1 describes the construction for a specific $r_l \leq r \leq r_u$. The output is the a list of all weakly non-isomorphic $\mathcal{I}$-polymatroids in $\mathscr{P}^q(N, (r, r), K, (s_l, s_u))$. A list of weakly non-isomorphic polymatroids is denoted as $\mathfrak{P}_{i,j}$, where i is the size of polymatroids in the list and j is the size of underlying simple polymatroid of every polymatroid in the list. At the ith iteration of the algorithm, a collection of such lists of polymatroids of size i are created from a collection of lists of polymatroids of size $i-1$. For every list $\mathfrak{P}_{i,j}$, a certificate map $\mathsf{cert}_{i,j}$ is also maintained, which maps members of $\mathfrak{P}_{i,j}$

Input: $\mathcal{I}$, a prime power q, $\mathbf{c} = (N, (r_l, r_u), K, (s_l, s_u))$ and vector space dimension r
Output: Lists $\mathfrak{P}_{N,s}, s_l \leq s \leq s_u$ of codes in $\mathscr{P}^q(\mathbf{c})$ and respective certificate maps $\mathsf{cert}_{N,s} : \mathfrak{P}_{N,s} \to T_N$

```
1  𝔓_{0,0} ← {P_null}
2  cert_{0,0}(P_null) ← φ_null
3  for 1 ≤ i ≤ s_u do
4  |   (𝔓_{i,i}, σ_{i,i}, φ_{i,i}) ← leiterspiel(𝔓_{i-1,i-1}, σ_{i-1,i-1}, φ_{i-1,i-1})
5  |   (𝔓_{i,i}, cert_{i,i}) ← pmapfilter(P_{i,i}, cert_{i-1,i-1}, σ_{i-1,i-1})
6  |   if i ≥ s_l + 1 then
7  |   |   for s_l ≤ j ≤ i − 1 do
8  |   |   |   𝔓_{i,j} ← nse(𝔓_{i-1,j})
9  |   |   |   (𝔓_{i,j}, cert_{i,j}) ← pmapfilter(P_{i,j}, cert_{i-1,j}, σ_{j,j})
10 |   |   end
11 |   end
12 |   for s_u + 1 ≤ i ≤ N do
13 |   |   for s_l ≤ j ≤ s_u do
14 |   |   |   𝔓_{i,j} ← nse(𝔓_{i-1,j})
15 |   |   |   (𝔓_{i,j}, cert_{i,j}) ← pmapfilter(P_{i,j}, cert_{i-1,j}, σ_{j,j})
16 |   |   end
17 |   end
18 end
19 return {𝔓_{N,s_l}, ..., 𝔓_{N,s_u}}, {cert_{N,s_l}, ..., cert_{N,s_u}}
```

Algorithm 1: An algorithm to solve CLRP$_q$-EN as implemented in ITAP

to the vertices of the p-map tree T_i at depth i, that correspond to the certificates of $p\mathcal{I}$-feasibility. The procedure leiterspiel$(\cdot)$ (line 4) refers to the Leiterspiel algorithm as described in [BBF$^+$06] (Algorithm 9.6.10), which serves as a concrete implementation of procedure se$(\cdot)$, which we mentioned previously, without giving any internal details (green arrows in fig. 2.8). The input to this procedure is the orbits datastructure, consisting of a list $\mathfrak{P}_{i,i}$ for some $1 \leq i \leq N$, stabilizer map $\sigma_{i,i}$ and the transporter map $\varphi_{i,i}$. The stabilizer map $\sigma_{i,i}$ maps the members of $\mathfrak{P}_{i,i}$ to subgroups of $P\Gamma L(r, q)$ that are their automorphism groups. The transporter map $\varphi_{i,i}$ maps a subset of size i of $\mathsf{Gr}_q(r, K)$ that is a $p\mathcal{I}$-polymatroid to the repesentative of its weak isomorphism class present in the list $\mathfrak{P}_{i,i}$. The procedure pmapfilter$(\cdot)$ in lines 5,9 and 15 corresponds to the red boxes in fig 2.8. The input to this procedure is a list of polymatroids $\mathfrak{P}_{i,j}$, the associated certificate map $\mathsf{cert}_{i,j}$ and the stabilizer map $\sigma_{j,j}$ which maps the underlying simple polymatroids of members of $\mathfrak{P}_{i,j}$ to the respective stabilizers. It uses extend_pmap$(\cdot)$ (line 3) to extend the certificate p-map of the parent polymatroid obtained using function $\mathsf{parent}(\cdot)$, and rejects any polymatroids for which no such extension exists. Note that extend_pmap$(\cdot)$ also takes the stabilizer subgroup of the underlying

simple polymatroid of the polymatroid being tested, in order to exploit symmetry as described in §2.3.5. Note that Leiterspiel, as described in [BBF+06], allows one to reject some of the generated objects if they do not satisfy an *inherited test function*, which is an indicator function for a any inherited property of the objects being generated. Hence, in actual implementation, rejection of poymatroids that are not $p\mathcal{I}$-feasible in line 5 is performed naturally in procedure leiterspiel(·) itself. The procedure nse(·) (lines 8 and 14) takes as input a list of polymatroids and outputs all strongly non-isomorphic non-simple extensions of the polymatroids in the list. For each polymatroid in the input list, it constructs all non-simple extensions (line 4 of nse) and rejects isomorphs using explicit strong isomorphism testing for polymatroids with identical underlying simple polymatroid (line 8 of nse).

```
1  𝔓′ ← ∅
2  for P ∈ 𝔓 do
3  |  φ ← extend_pmap(P, cert(parent(P)), σ(us(P)))
4  |  if φ ≠ φ_null then
5  |  |  𝔓′ ← 𝔓′ ∪ {P}
6  |  |  cert′(P) ← φ
7  |  end
8  end
9  return 𝔓′, cert′
```

Procedure pmapfilter($\mathfrak{P}$, cert, σ)

```
1  𝔓′ ← ∅
2  for P ∈ 𝔓′ do
3  |  i ← size of us(P)
4  |  𝔓″ ← i + 1 non-simple extensions of P
5  |  for P_ext ∈ 𝔓″ do
6  |  |  badpoly ← false
7  |  |  for P′_ext ∈ 𝔓′ do
8  |  |  |  if us(P_ext) = us(P′_ext) ∧ P_ext ≅ P′_ext then
9  |  |  |  |  badpoly ← true
10 |  |  |  end
11 |  |  end
12 |  |  if badpoly = false then
13 |  |  |  𝔓′ ← 𝔓′ ∪ {P}
14 |  |  end
15 |  end
16 end
17 return 𝔓′
```

Procedure nse($\mathfrak{P}$)

2.5 Computational Experiments

Accompanying this thesis is an implementation of the strategy described in Fig. 2.8 written in GAP [GAP15] that is available in form of a GAP4 package named ITAP (Information Theoretic Achievability Prover) [JJ15]. It uses Leiterspiel [BBF+06] to perform simple linear extensions. In this section we first consider a simple approach to measure the difficulty of solving a variant of CLRP$_q$-EN for a specific collection of constraints $\mathcal{I}$. We also consider several examples from literature to describe the functionality of ITAP via sample sessions in ITAP.

We will assume that every polymatroid generated using strategy in fig. 2.8 is endowed with a *rank oracle* i.e. a computer program that provides the rank of a subset of subspaces in time $\mathcal{O}(1)$. Denote by $\mathsf{RE}_{\mathcal{I}}(i, r, q, K)$, the number of evaluations of the rank oracle performed while constructing $p\mathcal{I}$-polymatroids at iteration i from $p\mathcal{I}$-polymatroids constructed at iteration $i-1$, for specific values of r, q and K, for a specific collection of constraints $\mathcal{I}$. $\mathsf{RE}_{\mathcal{I}}(i, r, q, K)$ can be written as,

$$\mathsf{RE}_{\mathcal{I}}(i, r, q, K) = \mathsf{RE}'_{\mathcal{I}}(i, r, q, K) \times \mathcal{N}_{i-1}^{r,q,K}, i \in [N] \tag{2.23}$$

where, $\mathsf{RE}'_{\mathcal{I}}(i, r, q, K)$ is the number of evaluations of the rank oracle per object and $\mathcal{N}_{i-1}^{r,q,K}$ denotes number of $p\mathcal{I}$-polymatroids at iteration i. Note that the first term in the expression above depends on the low level implementation and the collection of constraints $\mathcal{I}$. The second term, however, depends completely on the constraints $\mathcal{I}$. Hence, we are motivated to discuss the number of $p\mathcal{I}$-polymatroids constructed by ITAP as a measure of difficulty of solving a CLRP$_q$ variant for constraints $\mathcal{I}$. A closed form expression for the numbers of strongly non-isomorphic $p\mathcal{I}$-polymatroids in a particular class of codes for a particular collection of constraints $\mathcal{I}$ is unknown. Note that the numbers of $p\mathcal{I}$-polymatroids in a particular class of codes constructed by ITAP upper bounds the number of strongly non-isomorphic $p\mathcal{I}$-polymatroids in a particular class of codes, due to the hybrid approach of switching between weak and strong isomorphism adopted by ITAP (see §2.4.2). Interestingly, we observe that these numbers are much smaller than the number of all codes up to strong isomorphism in the same class of codes, at least in cases where such upper bounds are known.

The examples we consider, along with many others, are inbuilt in ITAP as part of the catalog of examples. The first example we consider is that of enumeration of all rate vectors achievable with a specified class of codes. The rest of the examples are concerned with the existential questions arising in varied contexts ranging from achievability proofs in network coding and secret sharing to proving linear rank inequalities. For each example, we state the associated constrained linear representability problem, plot the number of polymatroid representations constructed at the ith iteration of the algorithm along with known upper bounds, specify the time required to compute the answer, and describe the associated sample session with ITAP. All computations are performed on a 2 GHz Xeon CPU E5-2620 running Ubuntu 12.04 OS.

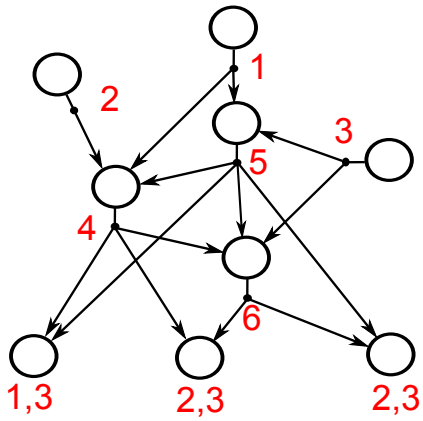

Figure 2.9: A HMSNC instance HN1 with 6 random variables considered in example 3

Example 3. Consider the 6-variable HMSNC instance HN1 in Fig. 2.9. It consists of 3 source random variables and 3 edge random variables. The exact rate region of this network is also shown

in below, which happens to be polyhedral.

$$\mathcal{R}_{\mathsf{HN1}} = \left\{ (\boldsymbol{\omega}, \mathbf{r}) \in \mathbb{R}^6 \;\middle|\; \begin{array}{r} R_4 \geq \omega_1 \\ R_4 + R_5 \geq \omega_1 + \omega_2 + \omega_3 \\ R_6 \geq \omega_1 \\ R_4 + R_6 \geq h_2 + \omega_3 \\ R_4 + R_5 + 2R_6 \geq \omega_1 + 2\omega_2 + 2\omega_3 \\ R_5 + R_6 \geq \omega_2 + \omega_3 \end{array} \right\} \tag{2.24}$$

The network constraints associated with this network are,

$$\mathcal{I}_{\mathsf{HN1}} = \left\{ \begin{array}{r} \{h_1 + h_2 + h_3 = h_{1,2,3}\}, \{h_{1,2,3} = h_{1,2,3,4}\}, \\ \{h_{1,3,4} = h_{1,3,4,5}\}, \{h_{3,4,5} = h_{3,4,5,6}\}, \\ \{h_{4,5} = h_{1,3,4,5}\}, \{h_{4,6} = h_{2,3,4,6}\}, \\ \{h_{5,6} = h_{2,3,5,6}\} \end{array} \right\} \tag{2.25}$$

We can construct achievable rate region that matches the exact rate region in (2.24) if we use codes from class $\mathscr{P}^2(6, (1,4), \{0,1,2\}, (1,6))$, as shown in the sample ITAP session. This computation takes about 387 sec. Constructing an achievable rate region from class $\mathscr{P}^q(6, (1,3), \{0,1,2\}, (1,6))$, however, results in a smaller rate region, shown in (2.27), computation that takes about 32 sec. The tree of $p\mathcal{I}_{\mathsf{HN1}}$-polymatroids generated by ITAP in the latter case is shown in Fig. 2.10. The leaves of this tree correspond to all weakly non-isomorphic network codes that belong to the family of representable integer polymatroids determined by the aforementioned parameters. Each leaf comes with a p-map that determines the rate vector achieved by the associated code, which is also shown in the figure. This p-map has the property that it is the lexicographically smallest map amoung all valid p-maps. By traversing rest of the p-map tree for these $\mathcal{I}_{\mathsf{HN1}}$-polymatroids, we recover the

following collection of achievable rate vectors:

$$\left\{ \begin{array}{c} [1,1,1,1,2,2],[1,1,1,2,1,2], \\ \\ [1,1,1,2,2,1],[1,1,1,2,2,2] \end{array} \right\} \tag{2.26}$$

To obtain the inequality description of the achievable rate region, we follow the procedure mentioned at the end of §2.2.1. This completes the computation of achievable rate region.

$$\mathcal{R}_{\text{in}} = \left\{ (\boldsymbol{\omega}, \mathbf{r}) \in \mathbb{R}^6 \left| \begin{array}{r} \omega_i \geq 0, i \in [3] \\ \\ R_i \geq \omega_k, i \in \{4,5,6\}, k \in [3] \\ \\ R_i + R_j \geq 3\omega_k, \forall \{i,j\} \subset \{4,5,6\} \text{ and } k \in [3] \\ \\ R_4 + R_5 + R_6 \geq 5\omega_i, i \in [3] \end{array} \right. \right\} \tag{2.27}$$

ITAP session with example 3

```
gap> N:=HyperedgeNet1();
[ [ [ [ 1, 2, 3 ], [ 1, 2, 3, 4 ] ], [ [ 1, 3, 4 ], [ 1, 3, 4, 5 ] ],
      [ [ 3, 4, 5 ], [ 3, 4, 5, 6 ] ], [ [ 4, 5 ], [ 1, 3, 4, 5 ] ],
      [ [ 4, 6 ], [ 2, 3, 4, 6 ] ], [ [ 5, 6 ], [ 2, 3, 5, 6 ] ] ], 3, 6 ]
gap> rlist:=proveregion(N,2,GF(2),[4]);; # k=2,opt_dmax=4=max. code dimension
gap> Size(rlist[1]); # number of distinct achievable rate vectors found
122
gap> rlist[1][78]; # an achievable rate vector
[ 2, 0, 1, 2, 1, 1 ]
gap> lrs_path:="/home/aspitrg3-users/jayant/lrslib-061/";; # path to lrslib
gap> rrcompute(rlist[1],N[2],N[3],lrs_path); # compute achievable rate region

*redund:lrslib v.6.1 2015.11.20(lrsgmp.h gmp v.5.0)
*Copyright (C) 1995,2015, David Avis   avis@cs.mcgill.ca
*Input taken from file /tmp/tmxYdXYT/file1.ext
*Output sent to file /tmp/tmxYdXYT/file1red.ext

*0.056u 0.004s 648Kb 0 flts 0 swaps 0 blks-in 8 blks-out


*lrs:lrslib v.6.1 2015.11.20(lrsgmp.h gmp v.5.0)
*Copyright (C) 1995,2015, David Avis   avis@cs.mcgill.ca
*Input taken from file /tmp/tmxYdXYT/file1red.ext
H-representation
begin
***** 7 rational
 0  0  0  0  1  0  0
 0  1  0  0  0 -1  0
 0  0  0  0  0  1  0
 0  0  0  0  0  0  1
 0  0  0  1  0  0  0
 0  1  1  0 -1 -1 -1
 0  0  1  1  0 -1 -1
```

```
 0  0  1  0  0  0  0
 0  1  1  2 -1 -2 -2
 0  1  0  1  0 -1 -1
end
*Totals: facets=10 bases=22
*Dictionary Cache: max size= 6 misses= 0/21   Tree Depth= 5
*lrs:lrslib v.6.1 2015.11.20(32bit,lrsgmp.h)
*0.000u 0.000s 648Kb 0 flts 0 swaps 0 blks-in 0 blks-out
```

Example 4. This is a collection of examples from network coding literature: the so called matroidal and discrete polymatroidal networks (see [DFZ07,MR14]). These networks are constructed to mimic the dependencies of matroids and integer polymatroids respectively. As a result, the known results regarding the representability of these polymatroids carry forward to the networks, thus providing us with the networks for which achievability or non-achievability of certain rate vectors is established by construction. The matroidal networks we consider are the Fano, Non-Fano and Vámos networks, whereas the discrete polymatroidal network we use as example is the network constructed from the polymatroid associated with a scaled version of U_4^2 matroid. Equations below describe the constraints associated with each of these networks. The Fano network is size $N = 7$ network associated with the Fano matroid. Rate vector $(\omega_i = 1, R_j = 1, i \in [3], j \in [7] \setminus [3])$ is achievable for this network using linear network coding only over a finite field of even characteristic, as the Fano matroid is only representable over such fields.

$$\mathcal{I}_{\mathsf{Fano}} = \left\{ \begin{array}{c} \{h_1 + h_2 + h_3 = h_{1,2,3}\}, \{h_{1,2} = h_{1,2,4}\}, \\ \{h_{2,3} = h_{2,3,5}\}, \{h_{4,5} = h_{4,5,6}\}, \\ \{h_{3,4} = h_{3,4,7}\}, \{h_{1,6} = h_{3,1,6}\}, \\ \{h_{6,7} = h_{2,6,7}\}, \{h_{5,7} = h_{1,5,7}\} \end{array} \right\} \tag{2.28}$$

ITAP session with Fano Network in Example 4

```
gap> FanoNet();
[ [ [ [ 1, 2 ], [ 1, 2, 4 ] ], [ [ 2, 3 ], [ 2, 3, 5 ] ],
     [ [ 4, 5 ], [ 4, 5, 6 ] ], [ [ 3, 4 ], [ 3, 4, 7 ] ],
     [ [ 1, 6 ], [ 3, 1, 6 ] ], [ [ 6, 7 ], [ 2, 6, 7 ] ],
     [ [ 5, 7 ], [ 1, 5, 7 ] ] ], 3, 7 ]
gap> rlist:=proverate(FanoNet(),[1,1,1,1,1,1,1],GF(2),[]);;
gap> rlist[1]; # Fano matroid is representable over GF(2)
true
gap> DisplayCode(rlist[2]);
1->1
```

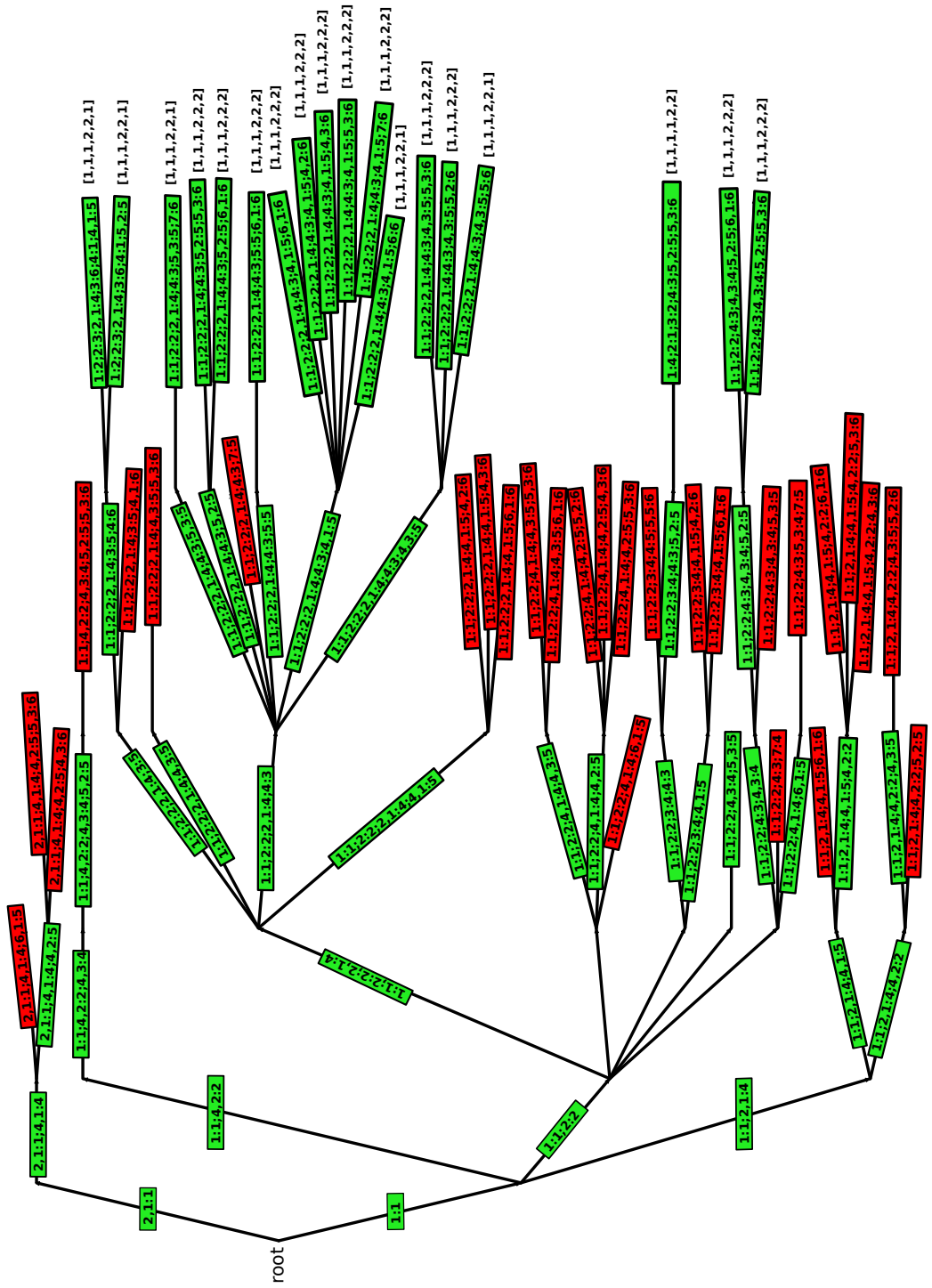

Figure 2.10: The generation tree for example 3 with class of codes $\mathscr{P}^2((6,(3,3)),1,2,(6,6)))$. The strings associated with edges encode the polymatroids and associated p-maps in a compact form. A string i,j:k;l,m:n is to be interpreted as the polymatroid associated with the subspace arrangement {{i,j},{l,m}} where numbers i,j,l,m correspond to integer representation of binary vectors in $\mathbb{F}_2^3$ and numbers k, n correspond to subscripts of the random variables associated with the network.

```
 . . 1
============================
2->2
 . 1 .
============================
3->4
 . 1 1
============================
4->3
 1 . .
============================
5->6
 1 . 1
============================
6->5
 1 1 .
============================
7->7
 1 1 1
============================
gap> rlist:=proverate(FanoNet(),[1,1,1,1,1,1,1],GF(3),[]);;
gap> rlist[1]; # Fano matroid is not representable over GF(3)
false
```

The second matroidal network we consider is the Non-Fano network, which is also a size $N = 7$ network, for which the rate vector $(\omega_i = 1, R_j = 1, i \in [3], j \in [7] \setminus [3])$ is achievable via linear network coding only over a finite field of odd characteristic.

$$\mathcal{I}_{\mathsf{NonFano}} = \left\{ \begin{array}{r} \{\{h_1 + h_2 + h_3 = h_{1,2,3}\}, \\ \{h_{1,2,3} = h_{1,2,3,4}\}, \{h_{1,2} = h_{1,2,5}\}, \\ \{h_{1,3} = h_{1,3,6}\}, \{h_{2,3} = h_{2,3,7}\}, \\ \{h_{4,5} = h_{3,4,5}\}, \{h_{4,6} = h_{2,4,6}\}, \\ \{h_{4,7} = h_{1,4,7}\}, \{h_{5,6,7} = h_{1,2,3,5,6,7}\} \end{array} \right\} \tag{2.29}$$

The third matroidal network we consider is the Vámos network of size $N = 8$ for which the rate

vector $(\omega_i = 1, R_j = 1, i \in [4], j \in [8] \setminus [4])$ is not achievable via linear coding over any finite field.

$$\mathcal{I}_{\mathsf{Vámos}} = \left\{ \begin{array}{r} \{h_1 + h_2 + h_3 + h_4 = h_{1,2,3,4}\}, \{h_{1,2,3,4} = h_{1,2,3,4,5}\}, \\ \{h_{1,2,5} = h_{1,2,5,6}\}, \{h_{2,3,6} = h_{2,3,6,7}\}, \\ \{h_{3,4,7} = h_{3,4,7,8}\}, \{h_{4,8} = h_{2,4,8}\}, \\ \{h_{2,3,4,8} = h_{1,2,3,4,8}\}, \{h_{1,4,5,8} = h_{1,2,3,4,5,8}\}, \\ \{h_{1,2,3,7} = h_{1,2,3,4,7}\}, \{h_{1,5,7} = h_{1,3,5,7}\} \end{array} \right\} \tag{2.30}$$

The last network we consider is the $2U_4^2$ network of size $n = 4$, whose network constraints mimic the dependencies of the U_4^2 matroid i.e. the 4 point line. This matroid is a forbidden minor for matroid representability over $\mathbb{F}_2$ [Oxl11], which means that the rate vector $(\omega_i = 1, R_j = 1, i \in [2], j \in [4] \setminus [2])$ is not achievable for this network via linear network coding over $\mathbb{F}_2$. However, the polymatroid $2U_4^2$, which is obtained by scaling the rank function of U_4^2 by 2, is linearly representable over $\mathbb{F}_2$, implying that rate vector $(\omega_i = 2, R_j = 2, i \in [2], j \in [4] \setminus [2])$ is achievable for this network using linear coding over $\mathbb{F}_2$.

$$\mathcal{I}_{2U_4^2} = \left\{ \begin{array}{r} \{h_1 + h_2 = h_{1,2}\}, \{h_{1,2} = h_{1,2,3}\}, \\ \{h_{1,3} = h_{1,2,3}\}, \{h_{2,3} = h_{1,2,3}\}, \\ \{h_{1,2} = h_{1,2,4}\}, \{h_{1,4} = h_{1,2,4}\}, \\ \{h_{3,4} = h_{1,3,4}\}, \{h_{3,4} = h_{2,3,4}\}, \{h_{2,4} = h_{1,2,4}\} \end{array} \right\} \tag{2.31}$$

Now that we have described the four network coding instances, we can ask ITAP some questions whose answers we already know.

For Fano, Non-Fano and Vámos networks, we test whether rate vector $(\omega_i = 1, R_j = 1, i \in [k], j \in [N] \setminus [k])$ is achievable over $\mathbb{F}_2$ and $\mathbb{F}_3$. The sample session with for Fano network with ITAP is also shown. The numbers of $p\mathcal{I}$-polymatroids are shown in figures 2.11 and 2.12. The upper bounds in these figures are the numbers of rank ≤ 3 and rank ≤ 4 non-isomorphic binary and ternary representable matroids respectively, which were first obtained by Wild (see [Wil94, Mara]). One can see from the plots that the number $p\mathcal{I}$-polymatroids maintained by ITAP in each instance is much

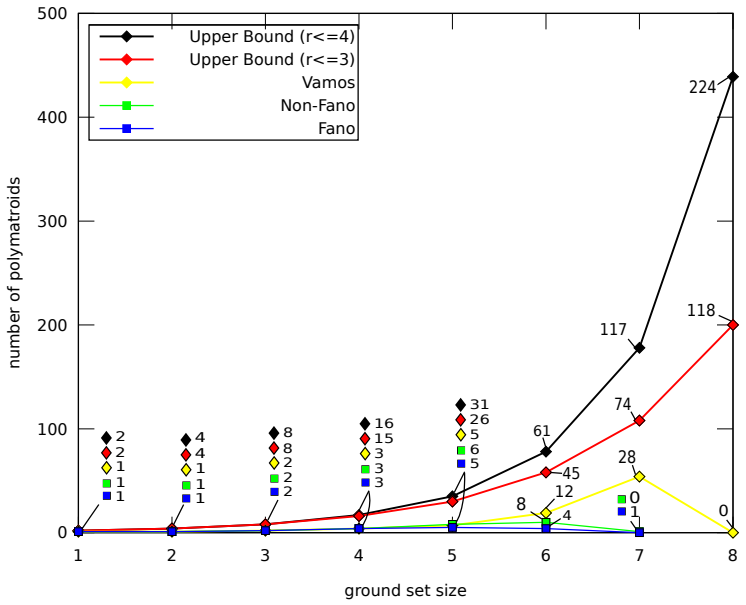

Figure 2.11: Number matroid representations over $\mathbb{F}_2$ maintained by ITAP at different iterations for Fano (blue), Non-Fano (green) and Vámos (yellow) networks along with upper bound which is the number of all non-isomorphic $\mathbb{F}_2$-representable matroids of rank ≤ 3 for Fano and Non-Fano networks while it is the number of $\mathbb{F}_2$-representable matroids of rank ≤ 4 for the Vámos network

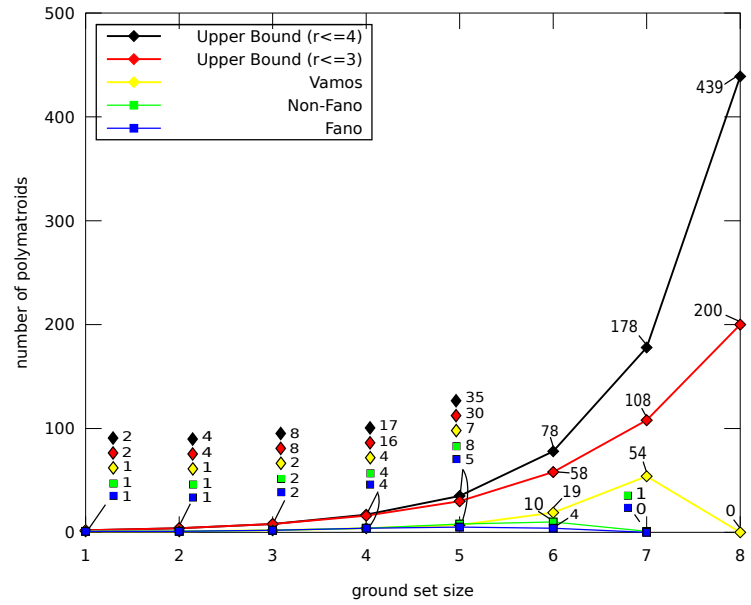

Figure 2.12: Number of weakly non-isomorphic matroid representations over $\mathbb{F}_3$ at different iterations for Fano, Non-Fano and Vámos networks along with upper bound (the number of all non-isomorphic $\mathbb{F}_3$-representable matroids of suitable rank)

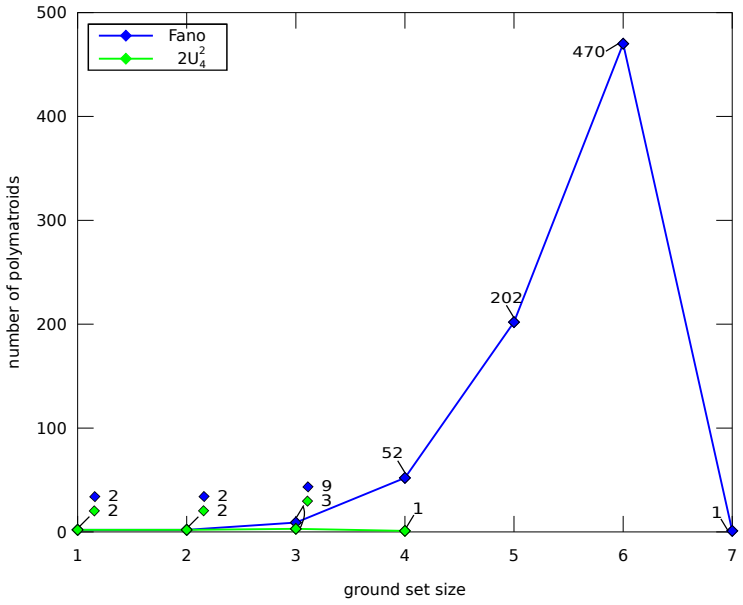

Figure 2.13: Number of polymatroid representations over $\mathbb{F}_2$ maintained by ITAP at different iterations for Fano and $2U_4^2$ networks

smaller than the respective upper bounds. The time required for testing scalar solvability of Fano and Non-Fano networks is 1 seconds and 2 seconds respectively over $\mathbb{F}_2$ whereas it is 5 seconds and 4 seconds respectively over $\mathbb{F}_3$. Timing results for Vámos network are discussed in detail in example 7.

For Fano and $2U_4^2$ networks, we test if rate vector $(\omega_i = 2, R_j = 2, i \in [k], j \in [N] \setminus [k])$ is achievable. We know that the answer is affirmative for both of these instances. This rate vector dictates that for Fano network, the class of codes to search for achievability construction is $\mathscr{P}^q(7,(6,6),\{2\},(3,7))$. Whereas, for $2U_4^2$ network, we have the class $\mathscr{P}^q(4,(4,4),\{2\},(2,4))$. The result, i.e. the numbers of weakly non-isomorphic representations constructed at each iteration by ITAP is shown in figure 2.13. This plot is left without any upper bounds, as the only known upper bound is the number of general 2-polymatroids obtained via Savitsky's enumeration [Sav14], already shown in Fig. 2.3. The time required in this case is 142 minutes and 1 seconds for Fano and $2U_4^2$ networks respectively. ■

Example 5. ([BL90, Pad13]) Let $\Gamma = \{\{2,3\},\{3,4\},\{4,5\}\}$ be an access structure for sharing a

secret with 4 parties labeled $\{2,3,4,5\}$ with the dealer labeled 1. This example appears in [BL90], whereas an explicit multi-linear secret sharing scheme for this access structure with secret size $r_1 = 2$ and share sizes $r_2 = 2, r_3 = 3, r_4 = 3, r_5 = 2$ can be found in [Pad13]. For the purpose of reproducing this scheme with ITAP, equations (2.6) and (2.7) (constraints $\mathcal{I}_\Gamma$) along with the requirement to have specified share sizes form collection of constraints $\mathcal{I}$. The constraints imply that the class of codes in which an achievable construction might exist is $\mathscr{P}^q(5,(3,12),\{2,3\},(2,5))$. Equation (2.32) gives the representation constructed by ITAP, with p-map $\{1\mapsto 1, 2\mapsto 2, 3\mapsto 3, 4\mapsto 5, 5\mapsto 4\}$. The sample ITAP session is also shown. This computation takes about 32 min. ■

$$P \triangleq \left\{ \overset{s_1}{\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}}, \overset{s_2}{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}}, \overset{s_3}{\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}}, \overset{s_4}{\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}, \overset{s_5}{\begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}} \right\} \tag{2.32}$$

ITAP session for Example 5

```
gap> B:=BenalohLeichter();
[ [ 1, 2 ], [ 2, 3 ], [ 3, 4 ] ]
gap> rlist:=provess(B,5,[2,2,3,3,2],GF(2),[]);;
gap> rlist[1];
true
gap> DisplayCode(rlist[2]);
1->1
 . . . . 1 .
. . . . . 1
============================
2->2
. . 1 . . .
. . . 1 . .
============================
3->3
. 1 . . . .
. . 1 . . 1
. . . 1 1 .
============================
4->5
1 . . . . .
. 1 . . . .
============================
5->4
```

```
1 . . . . 1
. 1 . . 1 .
. . 1 . . .
============================
```

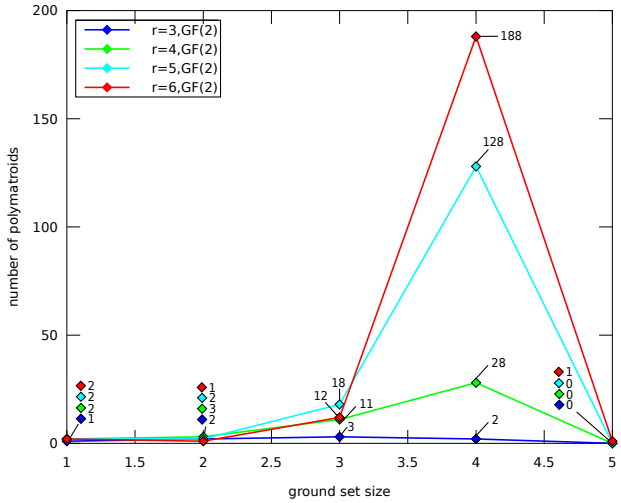

Figure 2.14: Number of weakly non-isomorphic $p\mathcal{I}$-polymatroid representations over $\mathbb{F}_2$ of various ranks constructed by ITAP at different iterations for example 5.

Example 6. This example is concerned with proving linear rank inequalities amoungst N random variables. Dougherty, Freiling and Zeger use a computational technique [Dou14] to find non-redundant linear rank inequalities in N random variables (linear rank inequalities that cannot be expressed as conic combinations of the others). The idea is to start with the Shannon outer bound $\Gamma_{N'}$ for $N' > N$, intersect it with equality constraints obtained by enforcing certain pairs of random variables to have common information equal to the mutual information between them (currently, at most 3 pairs have been used for $N = 6$ and $N' = 7$) and then projecting down onto the dimensions associated with subsets of N variables. The extreme rays of this cone that are linearly representable are also the extreme rays of the cone of the linear rank inequalities for N random variables. This is where a computer based representability checker comes into picture (i.e. a computer program capable of solving problem (E2) described in §2.1). A computer program like ITAP, in principle, is able to prove linear representability of such an extreme ray over $\mathbb{F}_q$ by solving problem (E2$_q$) described in §2.1. Consider the polymatroid $([6], f)$ where f is specified by the vector

$[1, 1, 2, 1, 2, 2, 2, 2, 3, 3, 4, 3, 4, 4, 4, 2, 3, 3, 4, 3, 4, 4, 4, 4, 5, 5, 6, 5, 6, 6, 6, 4, 5, 5, 6, 5, 6, 6, 6, 6, 6, 6, 6,$

$6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6]$, which is one of the extreme rays of the cone of linear rank inequalities [Dou14] for $N = 6$. The representation constructed by ITAP is shown in equation (2.33) (with p-map $\{\{i \mapsto i\},\ \forall i \in [6]\}$).

$$P \triangleq \left\{ \begin{matrix} s_1 \\ \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{matrix}, \begin{matrix} s_2 \\ \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{matrix}, \begin{matrix} s_3 \\ \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{matrix}, \begin{matrix} s_4 \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \end{matrix}, \begin{matrix} s_5 \\ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \end{matrix}, \begin{matrix} s_6 \\ \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \end{matrix} \right\} \tag{2.33}$$

Figure 2.15 shows the number of weakly non-isomorphic representations of $p\mathcal{I}$-polymatroids con-

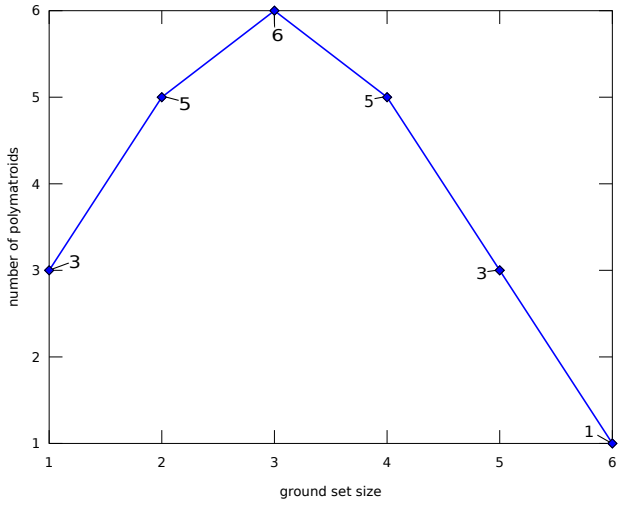

Figure 2.15: Number of $p\mathcal{I}$-polymatroid representations over $\mathbb{F}_2$ of various ranks constructed by ITAP at different iterations for example 6.

structed by ITAP at each iteration while solving $(\mathrm{E2}_q)$ w.r.t. the aforementioned ray. This computation takes about 375 sec. A similar strategy has been used by authors in [ACW16] to prove tightness of symmetric parts of Γ_N on symmetric parts of $\overline{\Gamma_N^*}$, ITAP was used to prove entropicness of the extreme rays of symmetric parts of Γ_N under the action of groups permuting the random variables. ■

Table 2.1: Time in seconds to test scalar solvability of Vámos network w.r.t. Field size q

q	Comb. Gen.	Gröbner
2	130	4
2	1330	5
4	5400	7
5	>2 hrs	5
7	>2 hrs	5
8	>2 hrs	7
9	>2 hrs	7

Example 7. In this example, we compare the computational performance of methods developed in this work and the Gröbner basis compution basis based methods, in particular the path gain construction of Subramanian and Thangaraj [ST10], for solving CLRP_q-EX. This method is implemented in ITAP via the GAP interface to singular [CdG12] which is available as an add on package to GAP, and interfaces to popular Gröbner basis computation software Singular [DGPS15]. The algorithm that transforms a given HMSNC instance to an instance of Network Coding over Direct Acyclic Multi-graphs (NCDAMG) is provided in the appendix. The first instance for which we compare the two methods is the Vámos network.Being a 'no' instance of CLRP_q-EX, this is expected to trigger the worst behaviour out of a combinatorial generation based algorithm (in a sense that the algorithm must traverse the entire search tree, in order to conclude that a solution does not exist). Our results are summarized in table 2.1, which suggest the same. One the other hand, we also find type of instances that incite bad behaviour from Gröbner basis computation based method. The number of indeterminates in path gain formulation depends on the number of paths. If the NCDAMG instance produced by algorithm 4 has p source-sink paths for a rate vector $(\omega_i = 1, R_j = 1), i \in [k], j \in [N] \setminus [k]$, doubling each rate produces a NCDAMG instance with 2^p paths. The instance we compare the performance of the two methods is a Multilevel Diversity

Table 2.2: Time in seconds to test achievability of a given rate vector over $\mathbb{F}_2$

Rate vector	Type of instance	Comb. Gen.	Gröbner
$[1,1,1,1,1,1,1]$	no	4	1
$[1,1,1,2,1,1,1]$	no	8	1
$[1,1,1,2,2,1,1]$	yes	9	>1 hr
$[1,1,1,2,2,2,2]$	yes	3	>1 hr
$[1,1,1,2,1,1,2]$	yes	5	>1 hr

Coding System (MDCS) [Yeu08] instance with $N = 7$, described as follows:

$$\mathcal{I}_{\mathsf{MDCS}} = \left\{ \begin{array}{r} \{\{h_1 + h_2 + h_3 = h_{1,2,3}\}, \{h_{1,2,3} = h_{1,2,3,4}\}, \\ \{h_{1,2,3} = h_{1,2,3,5}\}, \{h_{1,2,3} = h_{1,2,3,6}\}, \\ \{h_{1,2,3} = h_{1,2,3,7}\}, \{h_4 = h_{1,4}\}, \{h_5 = h_{1,5}\}, \\ \{h_{4,5} = h_{1,2,4,5}\}, \{h_{6,7} = h_{1,2,6,7}\}, \\ \{h_{4,6} = h_{1,2,3,4,6}\}, \{h_{5,7} = h_{1,2,3,5,7}\} \end{array} \right\} \tag{2.34}$$

The comparison results are shown in table 2.2.

Chapter 3: Outer Bound Algorithms

The entropy function region for N discrete random variables, or Γ_N^* was introduced by Yeung in his work *A Framework for Linear Information Inequalities* [Yeu97], as a tool for proving converse coding theorems for multi-terminal information theory problems. ITIP [YR08] and XITIP [RES08] are well-known software tools based on Yeung's framework, that allow one to use a computer to test whether a putative linear information inequality is implied by a collection of known linear information inequalities and a collection of linear constraints on the entropy function. Given a putative inequality, claimed to hold for the rate region of a multi-source network coding instance, one can determine if it is implied by a collection of linear information inequalities and the network coding constraints on the entropy function, in an identical fashion. While this is a neat scheme that allows us to verify putative inequalities for rate regions, it is not clear how to find the said putative inequalities in the first place. We consider the problem of computing explicit polyhedral outer bounds (EPOBs) on network coding rate regions, which is the problem of computing the tightest collection of inequalities implied by the given set of linear information inequalities, together with the constraints on the entropy function arising from a multi-source network coding instance. Yeung's framework has been used to simultaneously invoke known information inequalities via linear programming, to yield bounds on various scalar quantities of interest, such as capacity [DFZ07] and weighted sum rate [TGC11]. The problem of computing an EPOB has a similar flavor, however, the object we are interested in bounding is no longer a scalar, but a set of rate vectors. Note that, if we are to restrict ourselves to use only linear network codes, then, instead of known linear information inequalities, we would use known linear rank inequalities (see [Kin11, Dou14, DFZ09]), to obtain outer bounds on linear network coding rate regions.

The authors have amassed experience with computation of EPOBs for thousands of network coding instances. The classical technique used for polyhedral projection is that of Fourier-Motzkin elimination (FME), which removes one variable at every iteration. In case of EPOBs, however, we

need to eliminate exponentially many variables in the network coding instance size, which means FME has to pass through as many intermediate polyhedra to compute the projection. Thus, FME performs miserably when number of random variables involved is greater than 4. We propose an algorithm to compute EPOBs that builds upon an existing polyhedral computation technique, namely the Convex Hull Method (CHM), which is more suitable to such projections, due to its ability to work directly in the projection space. In a related problem of finding new non-Shannon type inequalities for a collection of N discrete random variables, one is faced with an analogous problem of eliminating a number of dimensions that is exponential in N. The convex hull method, and the closely related Benson's outer approximation algorithm have been the most successful in finding new non-Shannon type information inequalities, as evidenced by the work of Xu, Wang and Sun [W. 08] and Csirmaz [Csi13], respectively.

In order to push the limits of computation, in terms of the size of network coding instances that can be handled using this algorithm, we add the ability to exploit the symmetries of network coding instances [AW15a], which reduces the complexity of computing EPOBs. This work is accompanied by an implementation of the techniques for computing EPOBs, called Information Theoretic Converse Prover (ITCP) [JJ16] which is a GAP [GAP15] package. In addition to symmetry-aware computation of EPOBs, this package supports symmetry-aware computation of bounds on weighted sum rate in network coding, worst case information ratio in secret sharing, and guessing number of directed graphs.

In §3.1, we state the main problem considered in this work, which is the problem of computing explicit polyhedral outer bounds (EPOBs) on network coding rate regions. §3.2 states basic facts about polyhedra, and describes our implementation of the convex hull method, along with a discussion of its complexity, and its performance for computing EPOBs for a family of network coding instances. §3.3 establishes a connection between the network symmetry group of a network coding instance and the symmetries of the polyhedra involved in the computation of an EPOB on its rate region. In §3.4, we discuss the symmetry exploiting variant of the convex hull method, called symCHM, and describe the various complexity reductions it enables, along with a brief discussion

of how symmetry can be exploited in the related problem of computing weighted sum-rate bounds. Finally, in §3.6, we describe several sample sessions with ITCP, with interesting examples from the literature.

3.1 Explicit Polyhedral Outer Bounds

Γ_N^* is not yet fully characterized for $N \geq 4$. The known partial characterizations are in the form of linear information inequalities. An information inequality for N discrete random variables is an inequality of the following form,

$$\sum_{A \subseteq [N]} \lambda_A \mathbf{h}_A \geq 0 \tag{3.1}$$

where $\lambda_A \in \mathbb{R}$ for all $A \subseteq [N]$, that is satisfied by every entropic vector $\widetilde{\mathbf{h}} \in \Gamma_N^*$. Entropy function is known to satisfy polymatroidal axioms since Fujishige [Fuj78] and any consequences of them, which are also called the Shannon-type information inequalities. Inequalities not implied by the polymatroidal axioms, that are satisfied by all entropic vectors exist and are called the non-Shannon type information inequalities. The first such inequality was found by Zhang and Yeung [Z. 98]. Hundreds of linear non-Shannon type inequalities have been found since the Zhang-Yeung inequality [DFZ11, Csi13]. Furthermore, Matús [F. 07] has shown that for $N \geq 4$, an infinite number of linear information inequalities is necessary to determine $\overline{\Gamma}_N^*$. Hence, despite the implicit characterization described in chapter 1, most of the HMSNC rate region characterizations known so far have been found using the method of *sandwich bounds* [DFZ07, Conb], which is based on substitution of polyhedral inner and outer bounds in place of Γ_N^* in (1.5). In this work, we are interested in the EPOBs $\mathcal{R}_{\mathsf{out}}$ on the network coding rate regions, which can be defined implicitly as follows.

Definition 22. *An EPOB for the rate region of a HMSNC instance A is any outer bound $\mathcal{R}_{out}$ on the rate region, given as,*

$$\mathcal{R}_{out} = \mathit{proj}_{\boldsymbol{\omega},\mathbf{r}}(\Gamma_{out} \cap \mathscr{L} \cap \mathscr{L}') \tag{3.2}$$

where Γ_{out} is the region enclosed by a finite collection of linear information inequalities.

When Γ_{out} consists of all Shannon-type information inequalities, it is denoted as Γ_N, and the

resultant outer bound on the network coding rate region is known as the LP (Linear Programming) bound, denoted as $\mathcal{R}_{LP}$ [Yeu08]. Due to the use of only finitely many information inequalities, $\mathcal{R}_{\mathsf{out}}$ is naturally a polyhedral set. We say that $\mathcal{R}_{\mathsf{out}}$ is an *explicit polyhedral outer bound* (EPOB), if it is stated as a set of inequalities involving source rates $\boldsymbol{\omega}$ and edge capacities $\mathbf{r}$. We can now state the problem of computing EPOBs:

- [EPOB] Given a collection of linear information inequalities bounding Γ_{out} and a HMSNC instance A, explicitly determine the associated EPOB.

EPOBs for a handful of instances of the HMSNC problem are known in the literature. The works of Roche et. al. [RYH97], Hau and Yeung [HY97] and more recently Mohajer [MTD08] are representative of EPOBs obtained via human intuition. On the other hand much less attention has been paid to automating the derivation of EPOBs with the notable exceptions of Tian [Tia14] and the previous work of some of the authors and their collaborators [LWW12, LAWW13, Cona, Conb].

Note that, when interested only in linear network coding, one can replace Γ_N^* in the above discussion with subspace inner bound $\Gamma_N^{\mathsf{space}}$, and the phrase 'information inequalities' with 'rank inequalities'. $\Gamma_N^{\mathsf{space}}$ is fully characterized only for $N \leq 5$ [Kin11, DFZ09], while it is partially characterized for $N \geq 6$ [Dou14], via linear rank inequalities. The region Γ_{out}, in this context, will be an outer bound on $\Gamma_N^{\mathsf{space}}$, formed by a collection of known linear rank inequalities.

3.2 Polyhedra and their projection via CHM

In this section, after reviewing some of the basic terminology related to polyhedra, we give a procedural description of our implementation [Apt14] of the Convex Hull Method (CHM) in §3.2.1, which features several improvements over the original version proposed by Lassez et. al [LL93], while §3.2.2 and §3.2.3 discuss the low level details of CHM and a transformation that exables CHM to work with polyhedral cones, respectively.

A polyhedron $\mathcal{P} \subseteq \mathbb{R}^d$ [Zie95] can be represented either as the intersection of a finite number of closed halfspaces, $\mathcal{P} = \mathcal{P}(\mathbf{H}, \mathbf{z}) = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{H}\mathbf{x} + \mathbf{z} \geq \mathbf{0}\}$, referred to as its inequality representation, or, equivalently, as the sum of the convex hull of a finite set of extreme points and the conic hull

of a finite set of extreme rays, $\mathcal{P} = \mathsf{conv}(\mathbf{V}) + \mathsf{con}(\mathbf{Y}), \mathbf{V} \in \mathbb{R}^{d\times t}, \mathbf{Y} \in \mathbb{R}^{d\times t'}$, referred to as its extremal representation. Here $\mathsf{conv}(\cdot)$ and $\mathsf{con}(\cdot)$ refer to the convex and conic hull of the column vectors of $\mathbf{V}$ and $\mathbf{Y}$, respectively. If, in the extremal representation, $\mathbf{Y}$ is empty, the polyhedron is called a *polytope*, while if $\mathbf{V}$ is empty or exclusively the all zero-vector, the polyhedron is called a *polyhedral cone*. In the inequality representation, a polyhedral cone can be represented as $\mathcal{P}(\mathbf{H}, \mathbf{0})$ i.e. using only homogeneous inequalities. Any polyhedron can be viewed as a polyhedral cone of one higher dimension via a process known as *homogenization* [Zie95]. The inequality representation of the homogenization of $\mathcal{P}(\mathbf{A}, \mathbf{z})$, denoted by $\mathsf{homog}(\mathcal{P})$ is,

$$\mathcal{C}(\mathcal{P}) \subseteq \mathbb{R}^{d+1} := \mathcal{P}\left(\begin{bmatrix} -1 & \mathbf{0}^T \\ -\mathbf{z} & \mathbf{A} \end{bmatrix}, \begin{bmatrix} 0 \\ \mathbf{0} \end{bmatrix}\right) \tag{3.3}$$

while the extremal representation of the homogenization of a polyhedron with extreme points $\mathbf{V}$ and extreme rays $\mathbf{Y}$ is,

$$\mathcal{C}(\mathcal{P}) \subseteq \mathbb{R}^{d+1} := \text{cone}\left(\begin{bmatrix} \mathbf{1}^T & \mathbf{0}^T \\ \mathbf{V} & \mathbf{Y} \end{bmatrix}\right) \tag{3.4}$$

A key concept related to passing between the two descriptions of a polyhedral cone, and hence, via homogenization, the two descriptions of a general polyhedron, is that of the polar cone. The polar $\mathcal{C}^\circ$ ([Roc70], §14) of a convex cone $\mathcal{C}$ is $\mathcal{C}^\circ = \left\{\mathbf{x} \in \mathbb{R}^n \,\middle|\, \mathbf{x}^T\mathbf{y} \leq 0 \quad \forall \mathbf{y} \in \mathcal{C}\right\}$. For a polyhedral cone, polars swap the roles between the inequality and extremal descriptions, and $\mathcal{P}^{\circ\circ} = \mathcal{P}$. In particular, for a polyhedral cone $\mathcal{P} = \mathcal{P}(\mathbf{A}, \mathbf{0}) = \text{cone}(\mathbf{Y})$, the polar $\mathcal{P}^\circ = \mathcal{P}(-\mathbf{Y}^T, \mathbf{0}) = \text{cone}(-\mathbf{A}^T)$. For such a cone, $\mathbf{A}, \mathbf{Y}$ are said to form a *double descriptions (DD) pair* [FP96]. The projection of a polyhedron $\mathcal{P} \subseteq \mathbb{R}^d$ onto its first k dimensions, is another polyhedron $\mathsf{proj}_k(\mathcal{P})$ defined as,

$$\mathsf{proj}_k(\mathcal{P}) \triangleq \left\{\mathbf{x} \in \mathbb{R}^k \,\middle|\, \exists \mathbf{y} \in \mathbb{R}^{d-k} \text{ s.t. } (\mathbf{x}, \mathbf{y}) \in \mathcal{P}\right\} \tag{3.5}$$

Observe that the EPOB problem, as described in the previous section essentially asks us to construct the inequality representation of a projection of a polyhedron (Γ_{out} interscted with some constraints)

specified by its inequality representation.

3.2.1 Convex Hull Method: high level idea

CHM [LL93] is an algorithm to project polytopes by building successively better inner bounds to the projected polytope $\mathsf{proj}_k(\mathcal{P})$ via the solution of carefully selected linear programs over $\mathcal{P}$. The pseudocode for our implementation of [Apt14] is listed as algorithm 2. Note that we assume that the input polyhedron $\mathcal{P}$ is bounded and full-dimensional, so that the dimension of its affine hull is d. Tests for full-dimensionality and methods for the elimination of any redundant variables and inequalities of the input can be be implemented based on the guidelines in [Fuk04]. The algorithm

Input: Polyhedron $\mathcal{P}$ and k, the dimension of projection
Output: Vertex-facet set pair (V, H) of $\mathsf{proj}_k(\mathcal{P})$
1 $(V, H) \leftarrow \mathsf{initialhull}(\mathcal{P}, k)$
2 **while** $\exists\{\mathbf{hx} \geq \mathbf{b}\} \in H$ *s.t.* *isterminal*$(\mathcal{P}, \mathbf{h}, \mathbf{b}) = 0$ **do**
3 $\quad \mathbf{v} \leftarrow \mathsf{extremepoint}(\mathcal{P}, \mathbf{h})$
4 $\quad H \leftarrow \mathsf{updatehull}(\mathbf{v}, V, H)$
5 $\quad V \leftarrow V \cup \mathbf{v}$
6 **end**
7 **return** (V, H)

Algorithm 2: Convex Hull Method

relies on the fact that, if $\mathbf{c} \in \mathbb{R}^k$, then a point on the boundary of $\mathrm{proj}_k(\mathcal{P})$ that attains the solution to the linear program with cost vector $\mathbf{c}$ over $\mathrm{proj}_k(\mathcal{P})$, can be found by projecting the extreme point in $\mathcal{P}$ attaining the solution of the linear program with cost vector $[\mathbf{c}^T, \mathbf{0}_{d-k}^T]^T$ over $\mathcal{P}$, so that

$$\min_{\mathbf{x} \in \mathrm{proj}_k \mathcal{P}} \mathbf{c}^T \mathbf{x} = \min_{\mathbf{y} \in \mathcal{P}} [\mathbf{c}^T, \mathbf{0}_{d-k}^T] \mathbf{x}, \quad \text{and,} \tag{3.6}$$

$$\operatorname*{argmin}_{\mathbf{x} \in \mathrm{proj}_k \mathcal{P}} \mathbf{c}^T \mathbf{x} = \mathrm{proj}_k \left(\arg\min_{\mathbf{y} \in \mathcal{P}} [\mathbf{c}^T, \mathbf{0}_{d-k}^T] \mathbf{x} \right). \tag{3.7}$$

Furthermore, any extreme point of $\mathsf{proj}_k(\mathcal{P})$ can be described as a projection of some extreme point of $\mathcal{P}$, meaning that finding all extreme points of $\mathsf{proj}_k(\mathcal{P})$ is simply a matter of solving a series of linear programs over $\mathcal{P}$ and projecting the point attaining the optimum solution.

3.2.2 Convex Hull Method: low-level details

The first step in CHM is the construction of an inner bound to $\mathsf{proj}_k(\mathcal{P})$, which is the convex hull of $d+1$ points on the boundary of $\mathsf{proj}_k(\mathcal{P})$, that is itself full-dimensional. In order to obtain the initial

inner bound for the process (proc. initialhull), we first obtain two boundary points of $\text{proj}_k\mathcal{P}$ by solving two linear programs, $\min_{\mathbf{y}\in\mathcal{P}}[-1, \mathbf{0}_{d-1}^T]\mathbf{y}$ and $\min_{\mathbf{y}\in\mathcal{P}}[1, \mathbf{0}_{d-1}^T]\mathbf{y}$. We then select the normal vector $\mathbf{c}$ of any hyperplane containing these points (proc. hyperplane), and new boundary points of the projection are obtained as $\text{proj}_k \arg\min_{\mathbf{y}\in\mathcal{P}}[\mathbf{c}, \mathbf{0}_{d-k}]\mathbf{y}$ and $\text{proj}_k \arg\min_{\mathbf{y}\in\mathcal{P}}[-\mathbf{c}, \mathbf{0}_{d-k}]\mathbf{y}$. This process of finding a hyperplane containing all previously known boundary points and finding new boundary points is repeated until we obtain $k+1$ boundary points of the projection that give a full dimensional initial inner bound of the projection (in other words, until we obtain $k+1$ convex-independent boundary points in $\mathbb{R}^k$). This process needs to be repeated atmost $k+1$ times, owing to the full-dimensionality of $\mathcal{P}$. Given the set V containing $k+1$ boundary points of the projection forming a full-dimensional inner bound, computing the associated inequality description $\mathsf{conv}(V)$ corresponds to a $k+1 \times k+1$ matrix inversion (proc. facets). This gives us a double description pair of the first full dimensional inner bound of the projection.

At each stage of the algorithm, a DD pair is maintained for the current inner bound to $\text{proj}_k\mathcal{P}$. Each inequality in the inequality description of the inner bound carries with it a label, indicating whether or not it is terminal or non-terminal. The initial inner bound has all of its inequalities labelled as non-terminal. A non-terminal inequality $\mathbf{c}^T\mathbf{x} \geq b$ is then selected (proc. isterminal), and the linear program $\min_{\mathbf{x}\in\mathcal{P}}[\mathbf{c}^T\mathbf{0}_{d-k}^T]\mathbf{x}$ is solved over the high dimensional polyhedron. If the solution obtained is b, the inequality is marked as terminal. Otherwise, the extreme point of $\mathcal{P}$, obtained in the process, i.e. $\mathbf{v} = \arg\min_{\mathbf{x}\in\mathcal{P}}[\mathbf{c}^T\mathbf{0}_{d-k}^T]\mathbf{x}$ attaining the solution is projected to get a new boundary point $\text{proj}_k\mathbf{v}$ of $\text{proj}_k\mathcal{P}$. The DD pair of the inner bound is then updated (proc. updatehull) by adding the new boundary point, viewing it as a new inequality in the polar cone, via a single DD algorithm update step [FP96] (proc. DDiteration), and any new inequalities thus introduced are marked as non-terminal. Then a new non-terminal facet is selected and the process is repeated until all of the facets are labelled as terminal, at which point the inner bound has been proven equal to $\text{proj}_k\mathcal{P}$. The procedure $\mathsf{extremepoint}(\mathcal{P}, \mathbf{c})$ returns $\text{proj}_k\left(\arg\min_{\mathbf{y}\in\mathcal{P}}[\mathbf{c}^T, \mathbf{0}_{d-k}^T]\mathbf{x}\right)$.

Procedure $\mathsf{DDiteration}(\cdot)$ corresponds to an iteration of the DD method, returning the extreme rays of the cone which is formed as the intersection of the input cone and the input inequality.

Input: Polyhedron $\mathcal{P}$ and k, the dimension of projection
Output: (V, H) corresponding to initial hull of projection

$V \leftarrow \phi$
$\mathbf{p}_1 \leftarrow \mathsf{extremepoint}(\mathcal{P}, [1, 0, \ldots 0]_{1\times k})$
$\mathbf{p}_2 \leftarrow \mathsf{extremepoint}(\mathcal{P}, [-1, 0, \ldots 0]_{1\times k})$
$V \leftarrow V \cup \{\mathbf{p}_1, \mathbf{p}_2\}$
$\mathbf{h} \leftarrow \mathsf{hyperplane}(V)$
$i \leftarrow 3$
while $i \leq k$ **do**
  $\mathbf{p}_i \leftarrow \mathsf{extremepoint}(\mathcal{P}, \mathbf{h})$
  $V \leftarrow V \cup \{\mathbf{p}_i\}$
  $\mathbf{h} \leftarrow \mathsf{hyperplane}(V)$
  $i \leftarrow i + 1$
end
$H \leftarrow \mathsf{facets}(V)$
return (V, H)

Procedure initialhull$(\mathcal{P}, k)$

Procedure ineq returns the facet of the inner bound on projection that corresponds to the input ray, which is an extreme ray of polar of homogenization of that inner bound.

Input: Set of vertices V and corresponding set of inequalities H, new vertex $\mathbf{v}$
Output: Set of inequalities H' corresponding to $V' = V \cup \{\mathbf{v}\}$

$\mathcal{C} \leftarrow \mathsf{homog}(H)^\circ$
$\{\mathbf{r}_1, ..., \mathbf{r}_t\} \leftarrow \mathsf{DDiteration}(\mathcal{C}, \{[1 \;\; v_1, \ldots, v_d]\mathbf{y} \leq 0\})$
$H' \leftarrow \{\mathsf{ineq}(\mathbf{r}_1), \ldots, \mathsf{ineq}(\mathbf{r}_t)\}$
return H'

Procedure updatehull$(\mathbf{v}, V, H)$

3.2.3 Convex Hull Method: boundedness transformation

Note that CHM is made to project polytopes, but when calculating the bounds on network coding rate regions, we will be interested in projecting pointed polyhedral cones in $\mathbb{R}^d_{\geq \mathbf{0}}$, as all linear information inequalities are homogeneous inequalities while constraints arising from a network coding instance are homogeneous equations. Fortunately, an unbounded polyhedron $\mathcal{C}$ can be transformed to create a polytope $\mathcal{B}(\mathcal{C})$ such that the projection of the unbounded polyhedron $\mathcal{C}$ can be obtained from projection of $\mathcal{B}(C)$. While there are several such transformations, including the one in [LL93], we describe a transformation that is more efficient in terms of dimension of $\mathcal{B}(\mathcal{C})$ $(=d)$. For an arbitrary polyhedron in $\mathcal{P} \subseteq \mathbb{R}^d_{\geq \mathbf{0}}$, one can apply the transformations described in this section to $\mathsf{homog}(\mathcal{P}) \subseteq \mathbb{R}^{d+1}_{\geq \mathbf{0}}$. Let $\mathbf{Hx} \geq \mathbf{0}$ be the inequality description associated with a polyhedral cone $\mathcal{C}$.

This cone can be transformed into a polytope $\mathcal{C}' = \{\mathbf{x} \in \mathbb{R}^d_{\geq \mathbf{0}} \mid \mathbf{H}\mathbf{x} \geq \mathbf{0} \ \wedge \ (\mathbf{1}_k^T, \mathbf{0}_{d-k}^t)\mathbf{x} \leq 1\}$.

Lemma 6. *The inequality representation of $\mathsf{proj}_k(\mathcal{C}')$ is equal to the inequality description of $\mathsf{proj}_k(\mathcal{C})$ plus the additional inequality $\mathbf{1}_k^T\mathbf{x}_{1:k} \leq 1$.*

Proof: The inequalities in the inequality representation of $\mathsf{proj}_k(\mathcal{C})$ must be the only homogeneous inequalities in the representation of $\mathcal{C}'$, as adding anon-homogeneous inequality $[\mathbf{1}_k^T, \mathbf{0}_{d-k}^t]\mathbf{x} \leq 1$ to $\mathcal{C}$ cannot affect their minimality. Next, we must show that $\mathbf{1}_k^T\mathbf{x}_{1:k} \leq 1$ is the only non-homogeneous inequality that is non-redundant for $\mathsf{proj}_k(\mathcal{C}')$. Assuming the contrary, there must be an inequality $\mathbf{a}^T\mathbf{x}_{1:k} \leq b$ that is also non-redundant for $\mathsf{proj}_k(\mathcal{C}')$. Every inequality bounding $\mathsf{proj}_k(\mathcal{C}')$ can be obtained as a conic combination of inequalities in the inequality representation of $\mathcal{C}'$. Hence, $(b, \mathbf{a}^T) = \boldsymbol{\lambda}\mathbf{H} + \lambda_0(1, \mathbf{1}_k^T)$, where $\boldsymbol{\lambda}$ is a non-negative real vector and λ_0 is a non-negative real number. Furthermore, since $(\mathbf{1}_k^T, \mathbf{0}_{d-k}^t)\mathbf{x} \leq 1$ is the only non-homogeneous inequality in the inequality representation of $\mathcal{C}'$, $\lambda_0 \neq 0$. Now, $\boldsymbol{\lambda}\mathbf{H} \geq 0$ is a homogeneous inequality satisfied by projection, and hence can be written as a conic combination of non-redundant homogeneous inequalities in the inequality representation of $\mathsf{proj}_k(\mathcal{C}')$. This means, $\mathbf{a}^T\mathbf{x} \leq b$ can be written as a conic combination of inequalities bounding $\mathsf{proj}_k(\mathcal{C})$ and $\mathbf{1}_k^T\mathbf{x}_{1:k} \leq 1$, contradicting our assumption that it is non-redundant w.r.t. $\mathsf{proj}_k(\mathcal{C}')$. ■ Using lemma 6, given the inequality description of $\mathsf{proj}_k(\mathcal{C}')$, we can obtain the inequality representation of $\mathsf{proj}_k(\mathcal{C})$ by simply deleting the only non-homogeneous inequality present in the representation.

3.2.4 Convex Hull Method and EPOBs

In this subsection, we consider a simple family of HMSNC instances and guage the practical performance of CHM for computation of the associated EPOBs, along with a discussion of worst case complexity of CHM and its various components. When computing an EPOB for a HMSNC instance with N random variables, with $\Gamma_{\mathsf{out}} = \Gamma_N$, one starts with the polytope defined by the inequality representation containing elemental Shannon-type inequalities [Yeu97], network constraints and the bounding inequality $\sum_{i\in[k]} \boldsymbol{\omega}_i + \sum_{j\in[N]\setminus[k]} R_j \leq 1$. This polytope is then projected to $\boldsymbol{\omega}, \mathbf{r}$ co-ordinates using CHM and from the resultant inequality representation we simply remove the

bounding inequality, giving us the inequality representation of the EPOB.

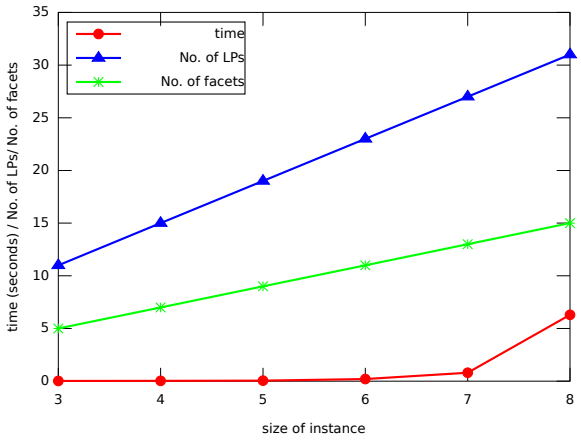

Figure 3.1: The runtime, number linear programs solved, and no. of facets of rate region outer bound, for computing EPC for a U_k^2 network vs k, the size of network, with $\Gamma_{\mathsf{out}} = \Gamma_k$ i.e. the Shannon outer bound.

Example 8. U_k^2 networks [HLWY13] are a family of matroidal networks, that can be constructed from uniform matroids of rank 2 on ground set of size k, using the construction of Dougherty, Freiling and Zeger [DFZ07]. Fig. 3.1 shows the plot of no. of linear programs solved by CHM and time taken by CHM as a function of k, which is the size of a U_k^2 network. The number of linear programs solved by CHM, which in this case, is the sum of number of facets and number of vertices of the EPOB, grows linearly w.r.t. k. On the other hand, the runtime grows exponentially w.r.t. k. The growth in runtime reflects the exponential growth in the dimension of the polyhedra being projected, as the outer bound Γ_k, which is used to compute these EPOBs, lives in a $2^k - 1$ dimensional space. ■

Note that the rate region outer bounds computed in example 8 are very *well-behaved* when compared to McMullen's upper bound of $\mathcal{O}(m^{\lfloor \frac{k}{2} \rfloor})$ [Zie95] on the no. of vertices of a k dimensional polytope specified by m inequalities. Simplex method, which is used in CHM to solve linear programs exactly by employing rational arithmetic, has exponential worst case complexity d. Hence, in the worst case, assuming that the only points of projection obtained during the course are the extreme points of projection, CHM could solve $\mathcal{O}(m^{\lfloor \frac{k}{2} \rfloor}) + m$ linear programs, one per vertex and facet of projection, with each having exponential runtime, given that the projection has m facets. Furthermore, the double description method, used in CHM to incrementally build the projection

polytopes, does not have an output sensitive time complexity [Bre96], as it is known to be susceptible to the insertion order of inequalities. In case of CHM, this means that the order in which we find the extreme points of the projection, greatly affects the number of rays in intermmediate double description pairs. Every iteration of DD method, the number of extreme rays can get squared, in the worst case. While the worst case analysis of the building blocks of CHM paints an ominous picture, both simplex method and double description method have been practically successful in solving moderate to large problem instances, which is essentially what the authors expect of CHM.

3.3 Polyhedral Symmetries and Network Symmetries

In this section, we first review the symmetry groups of network coding instances (§3.3.1), and the symmetry groups of polyhedra (§3.3.2). Finally, in §3.3.3, we establish a connection between the two in a way that the symmetries of the graph underlying the network coding instance can be interpreted as the symmetries of the polyhedra involved in the computation of the associated EPOB.

3.3.1 Network Symmetry Groups

Symmetry of the network coding instances was considered by the authors in [AW15a], where the problem model under consideration was that of network coding over directed acyclic graphs (as opposed to hypergraphs). While the treatment of network coding symmetry provided here is in the context of the HMSNC problem, the spirit, however remains the same. A symmetry of a HMSNC instance A is a permutation of the set [N] of subscripts of random variables associated with the network i.e. a bijection $\sigma : [N] \to [N]$. Any such permutation induces an action on network constraints $\mathcal{I}_A$ of a HMSNC instance A, by permuting the sets of random variables whose entropies appear in the network constraints. The network symmetry group (NSG), is then the subgroup of the symmetric group S_N which stabilizes $\mathcal{I}_{\mathsf{A}}$ setwise. The source independence constraint i.e. equation (1.2) is fixed pointwise by the NSG, implying that the set $[k]$ of source random variable subscripts is stabilized setwise. Hence, when defined as above, the NSG is always a direct product of a subgroup of $S_{[k]}$ with a subgroup of $S_{[N]\setminus[k]}$ which are symmetric groups permuting source labels and edge labels respectively.

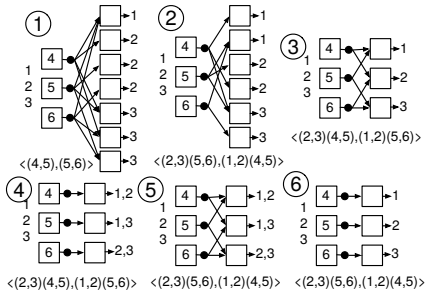

Figure 3.2: Six instances of 3-source 3-encoder IDSC problem that have NSG of order 6. The generators of respective NSGs are specified below each figure, written in the form of permutations of subscripts of random variable associated with sources ($\{1, 2, 3\}$) and encoders ($\{4, 5, 6\}$)

Fig. 3.2 shows NSGs of several instances of the Independent Distributed Source Coding Coding (IDSC) problem, which is a special case of HMSNC. The number of random variables associated with these instances is 6. A rather direct way of computing NSGs of these instances using GAP, is by defining the action of a permutation group on network constraints via the inbuilt action of permutation groups on sets of sets of sets i.e. on members of $2^{2^{2^{[N]}}}$. Indeed, the set of all network constraints minus the source independence constraint is a set of sets of sets, which means we can compute the NSGs as stabilizer subgroups of $S_{[k]} \times S_{[N]\setminus[k]}$, using this action. However, the size of the action domain grows very rapidly w.r.t. the size of the network coding instance. Hence, one can utilize algorithms for computing automorphism groups of graphs, as described in [AW15a]. The NSGs are also produced as a byproduct of the algorithm used for enumeration of HMSNC instances [Conb] .

3.3.2 Polyhedral Symmetries

In its most general form, a polyhedral symmetry is an automorphism of the face lattice (see e.g. [Zie95], pg. 57) of a polyhedron. For a d dimensional polyhedral cone with K extreme rays in the directions $\{\mathbf{r}_1, \ldots, \mathbf{r}_K\}$, these face lattice automorphisms can be identified as permutations on the set of rays that leave the faces intact, or equivalently bijections $\sigma : \{1, \ldots, K\} \to \{1, \ldots, K\}$ such that cone$(\{\mathbf{r}_{i_1}, \ldots, \mathbf{r}_{i_\ell}\})$ is a face of $\mathcal{P}$ if and only if cone$(\{\mathbf{r}_{\sigma(i_1)}, \ldots, \mathbf{r}_{\sigma(i_\ell)}\})$ is a face of $\mathcal{P}$ for every

$\{i_1, \ldots, i_\ell\} \subseteq \{1, \ldots, K\}$ and for every $\ell \in \{1, \ldots, K\}$. The group of all such symmetries, which can naturally be thought of as a subgroup of the symmetric group S_K, is called the *combinatorial symmetry group*. Calculating this group requires representation conversion [KS03] and knowledge of adjacency relationships, which makes it computationally prohibitive. A smaller subgroup of the combinatorial symmetries that is important for the solution of linear programs is the set of restricted affine symmetries, or simply *restricted symmetries* [Reh10, BSS09]), which can be defined as those bijections $\sigma : \{1, \ldots, K\} \to \{1, \ldots, K\}$ such that there exists a $d \times d$ matrix $\mathbf{T} \in \mathbb{R}^{d \times d}$ for which $\mathbf{r}_{\sigma(i)} = \mathbf{T}\mathbf{r}_i$ for all $i \in \{1, \ldots, K\}$. The label *restricted* comes from the fact that, that the vectors representing extreme rays of the cone are only defined up to a positive scalar multiple, but we have dictated a particular choice of these representative vectors $\mathbf{r}_i$ and that under the transformation this scalar multiple must be one. Alternatively, a rectricted symmetry can also be defined in terms of the inequality representation $\mathbf{Hx} \geq 0$ of the cone, as those permutations of rows of $\mathbf{H}$ a matrix $\mathbf{T} \in \mathbb{R}^{d \times d}$ s.t. $\mathbf{h}_i^T \mathbf{T} = \mathbf{h}_{\sigma(i)}$, where $\mathbf{h}_i^T$ denotes the ith row of $\mathbf{H}$.

The set of all such restricted symmetries of a polyhedral cone $\mathcal{P}$ form a group, with matrix multiplication as its operation, called the *restricted symmetry group* (RSG). The RSG can be computed via the automorphism group of an appropriately constructed edge-colored graph [Reh10,BSS09], and can be applied to general polyhedra through their homogenization into polyhedral cones. For full dimensional polytopes $\mathcal{P}$, with extreme points specified as the columns of matrix $\mathbf{V}$, by default, one can choose vector representatives for the extreme rays of their homogenization to be the columns of $\left[\mathbf{1}_t \mathbf{V}^T\right]^T$. Then, the RSG of homog($\mathcal{P}$), denoted by $G_{rs,\text{homog}(\mathcal{P})}$, is equivalent (isomorphic) to the *affine symmetry group* (ASG) of $\mathcal{P}$, defined as $G_{a,\mathcal{P}} = \left\{[\mathbf{b}, \mathbf{T}] \in \mathbb{R}^{d \times d+1} \,|\mathbf{T} \in GL_d(\mathbb{R}),\ \mathbf{T}\mathcal{P} + \mathbf{b} = \mathcal{P}\right\}$. Here $GL_d(\mathbb{R})$ is the general linear group of invertible $d \times d$ matrices with entries in $\mathbb{R}$, equipped with the operation of matrix multiplication. The relationship between $G_{rs,\mathsf{homog}(\mathcal{P})}$ and $G_{a,\mathcal{P}}$ can be stated as,

$$[\mathbf{b}, \mathbf{T}] \in G_{a,\mathcal{P}} \iff \begin{bmatrix} 1 & \mathbf{0}_d^T \\ \mathbf{b} & \mathbf{T} \end{bmatrix} \in G_{rs,\text{homog}(\mathcal{P})} \tag{3.8}$$

We can further constrain the notion of polyhedral symmetries by considering only those members

of $G_{rs,\mathsf{homog}(\mathcal{P})}$ that are permutation matrices i.e. matrices obtained from identity matrix by permuting the columns. These symmetries correspond to the permutations of co-ordinate dimensions under which the polytope $\mathcal{P}$ remains invariant.

3.3.3 NSGs and polyhedral symmetries

The action of S_N on subsets of subscripts of random variables, that was used to define the NSG, further induces an action on the vector space $\mathbb{R}_M$. This action can be written as a map $S_N \times \mathbb{R}^M \to \mathbb{R}^M$, such that, under permutation $\sigma \in S_N$, a vector $[\boldsymbol{\omega'}^T, \boldsymbol{r'}^T, \boldsymbol{h'}^T]^T$ is mapped to a new vector $\sigma([\boldsymbol{\omega'}^T, \boldsymbol{r'}^T, \boldsymbol{h'}^T]^T) = [\boldsymbol{\omega''}^T, \boldsymbol{r''}^T, \boldsymbol{h''}^T]^T$ with $\boldsymbol{\omega'}_i = \boldsymbol{\omega''}_{\sigma(i)}$, $\forall i \in [k]$, $\boldsymbol{r'}_j = \boldsymbol{r''}_{\sigma(j)}$, $\forall j \in [N] \setminus [k]$ and $\boldsymbol{h'}_{\mathcal{A}} = \boldsymbol{h''}_{\sigma(\mathcal{A})}$, $\forall \mathcal{A} \subseteq [N]$. The Shannon outer bound on the entropy region Γ_N is invariant under random variable permutations. Furthermore, it is reasonable to assume that an outer bound obtained by augmenting Γ_N with non-Shannon type information inequalities also satisfies this property, as we can add all permuted forms of any non-Shannon type inequalities to create such a bound. Hence, NSG stabilizes $\Gamma_{\mathsf{out}} \cap \mathcal{L}' \cap \mathcal{L}''$ setwise, and it forms a subgroup of the RSG of $\Gamma_{\mathsf{out}} \cap \mathcal{L}' \cap \mathcal{L}''$ associated with affine transformations $[\mathbf{b}, \mathbf{A}]$ that are linear (i.e. $\mathbf{b} = \mathbf{0}$) and with $\mathbf{A}$ a permutation matrix such that $\sigma([\boldsymbol{\omega}^T, \boldsymbol{r}^T, \boldsymbol{h}^T]^T) = [\tilde{\boldsymbol{\omega}}^T, \tilde{\boldsymbol{r}}'^T, \tilde{\boldsymbol{h}}^T]^T$ as previously defined is equal to $[\tilde{\boldsymbol{\omega}}^T, \tilde{\boldsymbol{r}}'^T, \tilde{\boldsymbol{h}}^T]^T = \mathbf{A}[\boldsymbol{\omega}^T, \boldsymbol{r}^T, \boldsymbol{h}^T]^T$. Thus, by computing the NSG, we automatically detect some symmetries of $\Gamma_{\mathsf{out}} \cap \mathcal{L}' \cap \mathcal{L}''$ as well as $\mathsf{proj}_{\boldsymbol{\omega},\mathbf{r}}(\Gamma_{\mathsf{out}} \cap \mathcal{L}' \cap \mathcal{L}'')$.

The Convex Hull Method, as described in §3.2 is able to project polytopes, and uses a boundedness transformation to handle general polyhedra. This boundedness transformation, when applied to $\Gamma_{\mathsf{out}} \cap \mathcal{L}' \cap \mathcal{L}''$ gives us a polytope $\mathcal{P} = \Gamma_{\mathsf{out}} \cap \mathcal{L}' \cap \mathcal{L}'' \cap \{(\mathbf{1}_N^T, \mathbf{0}_{2^N-1}^T)(\boldsymbol{\omega}^T, \boldsymbol{r}^T, \boldsymbol{h}^T)^T \leq 1\}$. Each non-zero extreme point of $\mathsf{proj}_{\boldsymbol{\omega},\mathbf{r}}(\mathcal{P})$ is the direction of an extreme ray of $\mathcal{R}_{\mathsf{out}}$ scaled to sum to one, and vice versa. Since the sum of an extreme ray direction is invariant under a permutation from the NSG, each member of NSG yields a permutation of the extreme points of $\mathsf{proj}_{\boldsymbol{\omega},\mathbf{r}}(\mathcal{P})$. Hence, the NSG gives a subgroup of the ASG of $\mathsf{proj}_{\boldsymbol{\omega},\mathbf{r}}(\mathcal{P})$, based on the action of the network symmetries on the vectors $(\widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}})$. In this manner, the symmetries imparted by the NSG are valid, despite the boundedness transformation. These symmetries can be exploited in CHM by employing known symmetry exploitation techniques in polyhedral computation, so that EPOBs can be computed with

lower complexity when such symmetries are present.

3.4 Symmetry Exploitation

Techniques for exploiting the knowledge of RSG, to reduce the complexity of polyhedral computation procedures have been studied in literature. Bremner et. al. [BSS09] consider the problem of exploiting the knowledge of RSG in polyhedral representation conversion, which is the problem of computing the inequality representation of a polyhedron, given the extreme ray representation (or vice versa). On the other hand, Bödi et. al [BH09, BHJ13] study how subgroups of RSG consisting of permutation matrices can be utilized to reduce the dimension of linear and integer programs.

In this section, we apply Bremner et. al.'s techniques to CHM, that allow one to compute the polyhedral projection by solving less linear programs. Bödi et. al's techniques can be applied to weighted sum-rate computation in network coding, computation of lower bounds on worst case information ratio in secret sharing and computation of upper bounds on guessing numbers of directed graphs. Note that such reduction in number and size of LPs solved is very important when one is interested in *exact solutions*, where the best known algorithm is the simplex method employing rational arithmetic, which has exponential worst case complexity. The authors believe that the emphasis on exact solutions is well- justified, especially for our cause of proving mathematical statements about HMSNC instances using a computer.

As discussed in the previous section, NSG determines a subgroup of RSG of $\Gamma_{\mathsf{out}} \cap \mathcal{L}' \cap \mathcal{L}''$ as well as the RSG of its projection $\mathcal{R}_{\mathsf{out}}$. Given the generality of RSG, it is safe to assume that the network symmetry group covers only a small fraction of RSG. For example, we have computed the RSG of Γ_4 using software sympol [Reh], which is a group of order 1152 with 8 generators, whereas permutations of random variables provide a subgroup of the RSG of order at most $4! = 24$ implying that at most only $\frac{1}{48}$th of the symmetries in RSG arise from random variable permutations. However, determining the RSG of Γ_{out} and $\Gamma_{\mathsf{out}} \cap \mathcal{L}' \cap \mathcal{L}''$ seem like daunting tasks for number of random variables ≥ 4, even when Γ_{out} is the Shannon outer bound.

Algorithm 3 provides a procedural description of symmetry exploiting CHM (symCHM). Just as in CHM, symCHM builds a sequence of progressively better inner bounds to $\mathrm{proj}_k(\mathcal{P})$. However,

in symCHM, each inner bound obtained is selected to be symmetric under the action of G i.e. the symmetry group of $\mathsf{proj}_k(\mathcal{P})$. The symmetry exploitation in symCHM happens at three different levels. First, memory is saved by storing only the vertices and facets inequivalent under G, secondly, we solve fewer linear programs in the process, and finally, the process of updating the inequality description of the intermediate inner bounds is replaced by a symmetric updating process that is more efficient. The following sections discuss these complexity reductions in detail.

3.4.1 symCHM: using symmetry to save on space

Let V and H be the set of vertices and facets respectively of $\mathsf{proj}_k(\mathcal{P})$. For an affine symmetry $g \in G$ and a vertex $\mathbf{v}$ of $\mathsf{proj}_k(\mathcal{P})$ denote by $\mathbf{v}^g$ to be the vertex to which $\mathbf{v}$ maps to under action of g and let $\mathbf{v}^G$, the *orbit* of $\mathbf{v}$ under G, be the set of all vertices to which $\mathbf{v}$ can map to under action of G. $\mathbf{v}^G$ contains all vertices that are G-*equivalent* to $\mathbf{v}$. The set of all orbits in V under action of G forms a partition of V and is denoted as $\mathcal{O}_V$. Since each facet is simply the convex hull of a set of vertices, the action of the ASG can be extended to H i.e. we define the orbit of a facet $h \in H$ denoted as h^G and $\mathcal{O}_H$ to be set of all orbits of facets. The transversal $\mathcal{T}$ of a set of orbits $\mathcal{O}$ is a set containing one representative per orbit in $\mathcal{O}$, and transversals of $\mathcal{O}_V$ of $\mathcal{O}_H$ are denoted as $\mathcal{T}_V$ and $\mathcal{T}_H$ respectively. Based on the aforementioned McMullen's Upper Bound, it is possible that the size of the double description of $\mathsf{proj}_k(\mathcal{P})$ is prohibitively large. In this case, we can trade space requirement for orbit computation, which is a basic procedure in computational group theory [Ser03], by storing only the transversals of the inequality and vertex orbit sets.

Input: Polyhedron $\mathcal{P} \subseteq \mathbb{R}^n$, projection dimension $d < n$ and symmetry group $G \leq S_k$ of $\mathsf{proj}_k(\mathcal{P})$
Output: Transversal pair $(\mathcal{T}_V, \mathcal{T}_H)$ of $\mathsf{proj}_d(\mathcal{P})$

$(\mathcal{T}_V, \mathcal{T}_H) \leftarrow \mathsf{syminitialhull}(\mathcal{P}, k)$
while $\exists\{\mathbf{hx} \geq \mathbf{b}\} \in \mathcal{T}_H$ *s.t.* *isterminal*$(\mathcal{P}, \mathbf{h}, \mathbf{b}) = 0$ **do**
  $\mathbf{v} \leftarrow \mathsf{extremepoint}(\mathcal{P}, \mathbf{h})$
  $\mathcal{T}_H \leftarrow \mathsf{symupdatehull}(\mathbf{v}, V, H)$
  $\mathcal{T}_V \leftarrow \mathcal{T}_V \cup \{\mathbf{v}\}$
end
return $(\mathcal{T}_V, \mathcal{T}_H)$

Algorithm 3: Symmetry exploiting CHM

Input: Transversal pair $(\mathcal{T}_V, \mathcal{T}_H)$, vertex $\mathbf{v}$ and group G
Output: Transversal $\mathcal{T}'_H$ associated with $\mathcal{T}'_V = \mathcal{T}_V \cup \{\mathbf{v}\}$
$(\mathcal{T}_{V_\mathcal{C}}, \mathcal{T}_{H_\mathcal{C}}) \leftarrow \mathsf{homog}(\mathcal{T}_H)^\circ$
$\{\mathbf{r}_1, ..., \mathbf{r}_t\} \leftarrow \text{symDD}(\mathcal{T}_{V_\mathcal{C}}, \mathcal{T}_{H_\mathcal{C}}, \{(1 \;\; \mathbf{v}^T)\mathbf{y} \leq 0\}, G)$
$\mathcal{T}'_H \leftarrow \{\mathsf{ineq}(\mathbf{r}_1), \ldots, \mathsf{ineq}(\mathbf{r}_t)\}$
return $\mathcal{T}'_H$

Procedure symupdatehull$(\mathbf{v}, V, H)$

Input: Transversal pair $\mathcal{T}_{V_\mathcal{C}}, \mathcal{T}_{H_\mathcal{C}}$ of the set of extreme rays and facets of a cone $\mathcal{C} \subseteq \mathbb{R}^{d+1}$, inequality $\mathbf{a}^T\mathbf{x} \leq 0$, group G
Output: Transversal of set of extreme rays of $\mathcal{C} \bigcap \cap_{g \in G} \{\mathbf{a}^T\mathbf{x} \leq 0\}^g$
$V_\mathcal{C} \leftarrow \mathcal{T}^G_{V_\mathcal{C}}$, $H_\mathcal{C} \leftarrow \mathcal{T}^G_{H_\mathcal{C}}$
$(P, N, Z) \leftarrow \mathsf{DD}(V_\mathcal{C}, H_\mathcal{C}, \mathbf{a})$
$V_{\mathcal{C}_{\mathbf{v}\leq}} \leftarrow P \cup Z$
$H_{\mathcal{C}_{\mathbf{v}\leq}} \leftarrow H_\mathcal{C} \cup \{\mathbf{a}^T\mathbf{x} \geq 0\}$
$(V_{\mathcal{C}_{\mathbf{v}=}}, H_{\mathcal{C}_{\mathbf{v}=}}) \leftarrow \mathsf{tightenfacet}(P \cup Z, H_{\mathcal{C}_{\mathbf{v}\leq}})$
$\mathcal{A} \leftarrow \mathsf{repDD}((V_{\mathcal{C}_{\mathbf{v}=}}, \mathbf{a}, G)$
for $\mathbf{a}' \in \mathcal{A}$ **do**
  $(P, N, Z) \leftarrow \mathsf{DD}(V_{\mathcal{C}_{\mathbf{v}=}}, H_{\mathcal{C}_{\mathbf{v}=}}, \mathbf{a}')$
  $V_{\mathcal{C}_{\mathbf{v}=}} \leftarrow P \cup Z$
  $H_{\mathcal{C}_{\mathbf{v}=}} \leftarrow H_{\mathcal{C}_{\mathbf{v}=}} \cup \{\mathbf{a}'^T\mathbf{x} \leq 0\}$
end
$\mathcal{T}_{V_{\mathcal{C}'}} \leftarrow \mathsf{nonisomorphic}((P \setminus N^G) \cup \mathsf{lift}(V_{\mathcal{C}_{\mathbf{v}=}}, \mathbf{a}), G)$
return $\mathcal{T}_{V_{\mathcal{C}'}}$

Procedure symDD$(\mathcal{T}_{V_\mathcal{C}}, \mathcal{T}_{H_\mathcal{C}}, \mathbf{a}^T\mathbf{x} \leq \mathbf{0}, G)$

3.4.2 symCHM: using symmetry to solve fewer linear programs

The transversal $\mathcal{T}_H$ of the facets of the inner bound carries with it an indicator variable indicating if it is terminal or non-terminal. At an intermediate step in the algorithm, a non-terminal facet $\mathbf{c}^T\mathbf{x} \geq b$ is selected from the current inner bound's facet transversal $\mathcal{T}_H$. Just as in CHM, the linear program $\min_{\mathbf{y} \in \mathcal{P}} [\mathbf{c}^T, \mathbf{0}^T_{d-k}]\mathbf{y}$ is solved, and if the result is b, the facet is marked as terminal. If the result is not b, the projection of the extreme point attaining the minimum, $\text{proj}_k \arg\min_{\mathbf{y} \in \mathcal{P}} [\mathbf{c}^T, \mathbf{0}^T_{d-k}]\mathbf{y}$, is added to the transversal $\mathcal{T}_V$. This act of adding this single extreme point $\mathbf{v}$ to the inner bound's vertex transversal has the same effect as having added the entire orbit $\mathbf{v}^G$ to the full list of extreme points in CHM. In ordinary CHM, to find these extreme points, $|\mathbf{v}^G|$ linear programs would have had to be solved, whereas in symCHM, only one LP is required to obtain all of them, followed by an orbit computation. Similarly, if a facet of an inner bound is found to be terminal, all the facets is its orbit under G can be labeled as terminal, amounting to further reduction in the number of LPs

solved.

Remark. *The framework of Bödi et. al. (see [BH09, BHJ13]) allows reduction in the dimensionality of a linear program, given the knowledge of the symmetries of the linear program. The symmetries of linear programs are the affine symmetries of the polytope associated with the constrained region, that additionally keep the cost vector fixed. The reduced dimensional LP is solved over a subset of the constrained region, obtained by intersecting it with an appropriate subspace determined by the group of symmetries of the original linear program. The caveat, however, is that, while the optimum values of the original and reduced dimensional LPs are identical, the points which attain the optimum could be different. In case of CHM, we project the vertex attaining the optimum value, under a guarantee that it yields a point on the boundary of the projection. More often than not, the projected point is in fact an extreme point of the projection (e.g. in example 8 every projected point is in fact an extreme point). On the other hand, it is possible that the optimum vertex associated with the reduced dimensional linear program projects down to a boundary point of the projection that is not an extreme point, which introduces redundancy. However, the ability to find the optimum value by solving a reduced dimensional LP is still useful for determining whether a facet of the projection is terminal, a situation where the optimum vertex is of no further use. Fig. 3.3 shows the number of linear programs solved by CHM and sym CHM for computing EPOBs for the six IDSC instances in fig. 3.2, along with the dimension of the LPs, if one needed to know only the optimum value and not the optimum vertex. Thus, Bödi et. al.'s framework [BH09, BHJ13] finds limited use in symCHM. However, it is fully applicable to computation of weighted sum-rate bounds, which we elaborate on in §3.5.*

3.4.3 symCHM: efficient inner bound updates using symmetry

When a new extreme point $\mathbf{v}$ is added to the transversal $\mathcal{T}_V$, the transversal of the inequalities $\mathcal{T}_H$ must be updated (proc. symupdatehull) to reflect the new inequalities that the addition of the extreme points $\mathbf{v}^G$ to the symmetric inner bound creates (we call this new polytope the *symmetric improvement*). In ordinary CHM, this would have been done through of $|\mathbf{v}^G|$ steps of the DD method applied to the complete inequality description of the symmetric inner bound. However, based on

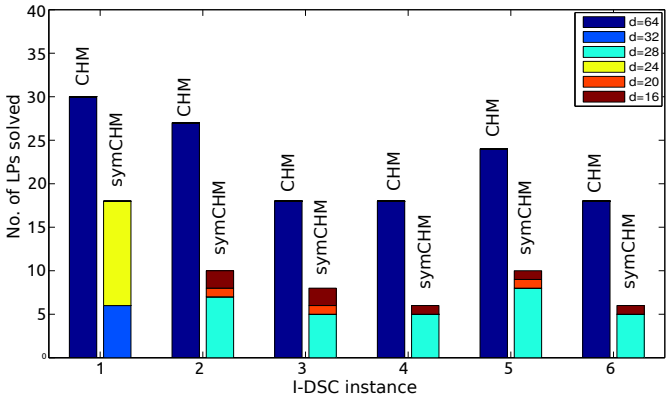

Figure 3.3: Comparison of number of LPs solved in chm vs symchm for computing EPOBs w.r.t. Γ_6, for non-isomorphic IDSC instances in fig 3.2. The dimensionality reduction brought about by the NSGs of these instances is also depicted, by classifying the LPs according to the dimension d of the reducted dimensional LP, if one is only interested in the optimal value and not the vertex attaining it.

Lemma 7, which is the same insight from which the incidence decomposition method [BSS09] for representation conversion of symmetric polyhedra is derived, we can perform DD steps of smaller size (proc. symDD) to obtain the new facets that must be added to the transversal. The size of a DD step, in this context is the number of extreme rays in the input double description step.

Lemma 7. *Let $\mathcal{P}_k^{(\ell)}$ be an inner bound on $\mathsf{proj}_k(\mathcal{P})$ whose ASG has G_p as a subgroup, and let $\mathbf{v}$ be a new vertex of a symmetric improvement $\mathcal{P}_k^{(\ell+1)}$. If, in $\mathcal{P}_k^{(\ell+1)}$, $\{f_1, \ldots, f_t\}$ is the set of facets incident to $\mathbf{v}$ then, $\{f_1^g, \ldots, f_t^g\}$ is the set of facets incident to $\mathbf{v}^g$.*

Lemma 7 ensures that as long as we calculate the facets of $\mathcal{P}_k^{(\ell+1)}$ incident to $\mathbf{v}$ correctly, and include any of these facets that are G-inequivalent into the new transversal $\mathcal{T}_H$ after removing those non-terminal inequalities that the new extreme points violate, the new facet transversal will reflect all of the G-inequivalent facets of $\mathcal{P}_k^{(\ell+1)}$. The key issue in calculating the facets incident to $\mathbf{v}$ in $\mathcal{P}_k^{(\ell+1)}$ correctly is that there may be some vertices in $\mathbf{v}^G \setminus \{\mathbf{v}\}$ that are adjacent to $\mathbf{v}$.

To address this issue, we first determine the double description pair of cone $\mathcal{C}_{\mathbf{v}^=}$, given as,

$$\mathcal{C}_{\mathbf{v}^=} \triangleq \mathcal{C} \cap \{[1 \ \mathbf{v}^T]\mathbf{x} = 0\}, \tag{3.9}$$

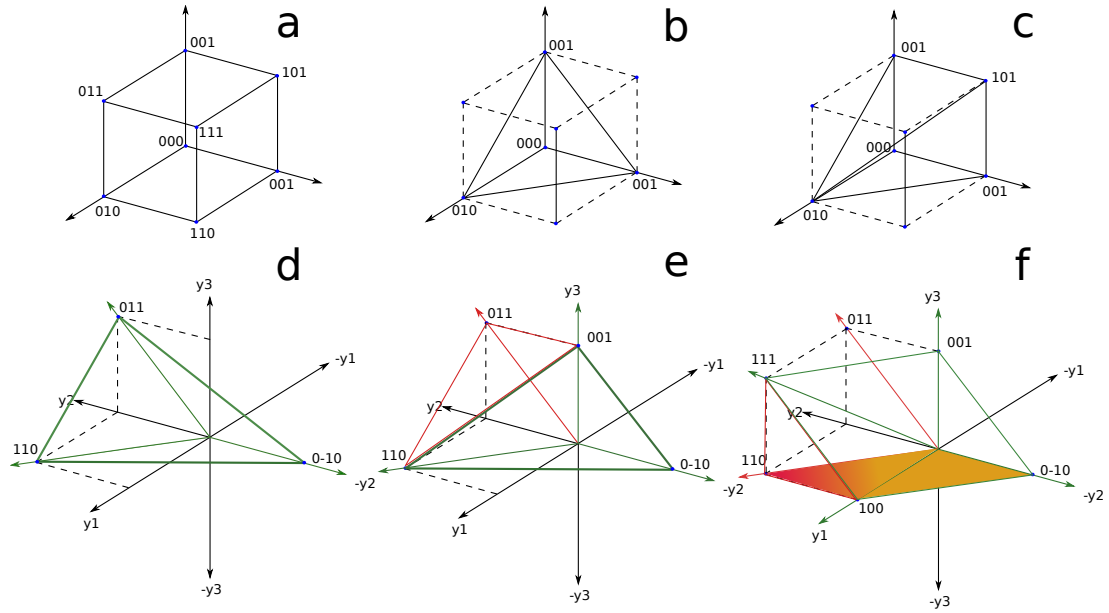

Figure 3.4: Symmetric update of an inner bound of a 3D cube. Part (a) shows the 3D cube in question with symmetry group S_3 which is the projection polytope to be computed, part (b) shows a 3D simplex that forms a symmetric inner bound to the projection, by the virtue of a symmetry group S_3. The inequalities and extreme points of (c) are shown in (3.13) and (3.14) respectively. Part (c) shows the updated inner bound obtained by adding vertex $\mathbf{v} = (1, 0, 1)$ to the description. Polar of homogenization of the polytope in (c) lies in $(\mathbb{R}^4)^\circ$, which is referred to as $\mathcal{C}_{\mathbf{v}\leq}$, whose rays are shown in (3.17). The cone in part (d) is $\mathcal{C}_{\mathbf{v}=}$, that lives in $(\mathbb{R}^3)^\circ$, whose rays are shown in (3.19). Parts (e) and (f) show the lower dimentional double description steps performed to obtain the symmetric update. The extreme rays of the cone in part (f) are shown in (3.20)

where $\mathcal{C}$ is the homogenized polar of the current inner bound $\mathcal{C} = \text{homog}(\mathcal{P}_k^\ell)^\circ$. The double description pair associated with $\mathcal{C}$ is computed from the associated transversals in line 1 of proc. symDD. The double description pair of $\mathcal{C}_{\mathbf{v}=}$ is determined by first determining the double description pair of $\mathcal{C}_{\mathbf{v}\leq} = \mathcal{C} \cap \{(1, \mathbf{v}^T)\mathbf{x} \leq 0\}$ through an iteration of double description step, in lines 2 and 4 of proc. symDD. The procedure DD, used for this purpose, returns sets P, Z and N of extreme rays, where P and N are the extreme rays of $\mathcal{C}$ that strictly satisfy and violate the inequality $\{(1, \mathbf{v}^T)\mathbf{x} \leq 0\}$ respectively, while the set Z contains the extreme rays of $\mathcal{C}$ that evaluate to zero w.r.t. $\{(1\ \mathbf{v}^T)\mathbf{x} \leq 0\}$ in addition to the new extreme rays that are computed as conic combinations of rays in P and N. The set of extreme rays of $\mathcal{C}_{\mathbf{v}=}$ is the set Z, i.e. the rays of $\mathcal{C}_{\mathbf{v}\geq}$ that have inequality $\{(1, \mathbf{v}^T)\mathbf{x} = 0\}$ incident to them, while its inequality representation is formed by the subset of inequalities in $\hat{H}_{\mathcal{C}}$

that are adjacent to $\{(1, \mathbf{v}^T)\mathbf{x} \leq 0\}$. The computation of double description pair of $\mathcal{C}_{\mathbf{v}=}$ is condensed into proc. tightenfacet on line 5 of proc. symDD. Note that cone $\mathcal{C}_{\mathbf{v}=}$ has dimension one lower that of $\mathcal{C}_{\mathbf{v}\leq}$ along with having only a subset of its extrem rays and facets. The symmetry exploitation is achieved by using only $\mathcal{C}_{\mathbf{v}=}$ from this poin onwards, to compute the symmetric update. To check to see if any vertices in $\mathbf{v}^G \setminus \{\mathbf{v}\}$ are adjacent to $\mathbf{v}$, and if so, which ones, we can determine the set $\mathcal{A}$, defined as,

$$\mathcal{A} = \left\{ (1, \mathbf{z}^T) \,\middle|\, \mathbf{z} \in \mathbf{v}^G \setminus \{\mathbf{v}\} \min_{\mathbf{x} \in \mathcal{C}_{\mathbf{v}=}} (1 \ \mathbf{z}^T)\mathbf{x} < 0 \right\} \tag{3.10}$$

Procedure repDD (line 6 of symDD) is used to compute the set $\mathcal{A}$. The rays of $V_{\mathcal{C}_{\mathbf{v}=}}$ are further refined by adding the inequalities $\{\mathbf{w}^T\mathbf{x} \leq 0\}$ for each $\mathbf{w} \in \mathcal{A}$ if any, through $|\mathcal{A}|$ further DD steps. This refinement is carried out in line 8 of proc. symDD. The new inequality transversal of $\mathcal{P}_k^{(\ell+1)}$ is can be created by removing any G-equivalent inequalities from $\mathcal{P}_k^{\ell}$'s inequality description (i.e. rays in the homogenized polar) in these $|\mathcal{A}| + 1$ DD steps, and by adding the representatives of the new rays introduced at the end of these $|\mathcal{A}| + 1$ DD steps, in line 12 of proc. symDD, where proc. lift is used to embed the refined set of rays of $\mathcal{C}_{\mathbf{v}=}$ into the higher dimensional space in which $\mathcal{C}$ existed. The consideration of symmetry in this step of updating the inequality description of the inner bound, reduced both the number of DD steps required for CHM and their size. Indeed, only $|\mathcal{A}| + 1$ DD steps must be performed to find the result of adding $|\mathbf{v}^G|$ new extreme points. Also, the size of each these DD steps is substantially smaller, since the cone $\mathcal{C}_{\mathbf{v}=}$ is being dealt with in the $|\mathcal{A}|$ latter DD steps, rather than $\mathcal{C}$.

Example 9. This is a simple example elaborating how procedure symDD works. Consider a cube $\mathcal{P}_3 \subseteq \mathbb{R}^3$ (fig. 3.4-a) with inequality representation,

$$H = \{0 \leq \mathbf{x}_i \leq 1, i \in [3]\}, \tag{3.11}$$

and extreme points,

$$V = \left\{ \begin{array}{l} (0,0,0),(0,0,1),(0,1,0),(0,1,1), \\ \\ (1,0,0),(1,0,1),(1,1,0),(1,1,1) \end{array} \right\} \tag{3.12}$$

$\mathcal{P}_3$ is the projection of any hypercube $\mathcal{P} \subseteq \mathbb{R}^d, d \geq 4$. The symmetric group S_3 is a group of symmetries of $\mathcal{P}_3$, as $\mathcal{P}_3$ is stabilized setwise under any permutation of co-ordinate dimensions. A simplex $P_3^{(l)} \subseteq \mathcal{P}_3 \subseteq \mathbb{R}^3$ (fig. 3.4-b) is the convex hull of $V^{(l)} \subseteq V$,

$$V^{(l)} = \{(0,0,0),(0,0,1),(0,1,0),(1,0,0)\}, \tag{3.13}$$

and has inequality representation,

$$H^{(l)} = \{0 \leq \mathbf{x}_i, i \in [3]\} \cup \{\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 \leq 1\}. \tag{3.14}$$

Let $\mathcal{P}_3^{(l)}$ be the inner bound at the lth iteration of symCHM. Now consider the problem of computing the inequality representation $H^{(l+1)}$ of the symmetric update $P_3^{(l+1)}$, i.e. the convex hull of $V^{(l)} \cup \mathbf{v}^{S_3}$ where $\mathbf{v}$ is the vertex $(1,0,1)$, which is presumably found by solving a linear program over the aforementioned hypercube $\mathcal{P}$ (line 3 of symCHM). The first input to procedure symDD is the transversal of the orbits of the rays of $\mathcal{C} = \mathsf{homog}(P_3^{(l)})^\circ \subseteq (\mathbb{R}^4)^\circ$, given as,

$$\mathcal{T}_\mathcal{C} = \{(0,-1,0,0),(-1,1,1,1)\}, \tag{3.15}$$

which bears one to one correspondance with transversal of orbits of S_3 in $H^{(l)}$. The second input is the inequality $\mathbf{y}_0 + \mathbf{y}_1 + \mathbf{y}_3 \leq 0$, which corresponds to vertex $\mathbf{v}$, after homogenization and polar. The initial double description step inserts this inequality in $\mathcal{C}$, giving the double description pair of cone $\mathcal{C} \cap \{\mathbf{y} \mid \mathbf{y}_0 + \mathbf{y}_1 + \mathbf{y}_3 \leq 0\} \subseteq (\mathbb{R}^4)^\circ$ which corresponds to the polytope shown in fig. 3.4-c, in

the original space. $\mathcal{C} \cap \{\mathbf{y} \mid \mathbf{y}_0 + \mathbf{y}_1 + \mathbf{y}_3 \leq 0\}$ has inequality representation,

$$\begin{aligned} H^{\mathcal{C}_{\mathbf{v}^{\geq}}} &= T_1 \cup T_2 \cup T_3 \text{ where,} \\ T_1 &= \{\mathbf{y}_0 + \mathbf{y}_i \leq 0, \forall i \in [3]\} \\ T_2 &= \{\mathbf{y}_0 \leq 0\} \\ T_3 &= \left\{ \sum_{i \in \{0,1,3\}} \mathbf{y}_i \leq 0 \right\} \end{aligned}, \tag{3.16}$$

where the last inequality is the newly added inequality, and extreme rays,

$$V^{\mathcal{C}_{\mathbf{v}^{\leq}}} = \left\{ \begin{array}{l} (0,-1,0,0),(0,0,0,-1),(0,0,-1,0), \\ (-1,0,1,1),(-1,1,1,0) \end{array} \right\} \tag{3.17}$$

The first two rays in $V^{\mathcal{C}_{\mathbf{v}^{\leq}}}$ belong to the set P in line 2 of symDD and last three rays belong to set Z, while the set N contains the ray $(-1,1,1,1)$, which in original space corresponds to the inequality $\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 \leq 1$, is not part of the inequality representation of the symmetric updates, and is to be removed from consideration along with all its permutations in line 12 at the end of proc. symDD($N^{S_3} = N$ in this case). The cone $\mathcal{C}_{\mathbf{v}^{=}}$ is obtained by making last inequality in $H^{\mathcal{C}}$ tight, and has inequality representation,

$$H^{\mathcal{C}_{\mathbf{v}^{=}}} = \left\{ \begin{array}{l} -\mathbf{y}_1 \leq 0, -\mathbf{y}_1 + \mathbf{y}_2 - \mathbf{y}_3 \leq 0, \\ -\mathbf{y}_3 \leq 0, -\mathbf{y}_1 - \mathbf{y}_2 \leq 0 \end{array} \right\}, \tag{3.18}$$

which is obtained by substituting $\mathbf{y}_0 = -\mathbf{y}_1 - \mathbf{y}_3$ in the first 4 inequalities of $H^{\mathcal{C}}$. Note that the last inequality in $H^{\mathcal{C}_{\mathbf{v}^{=}}}$ is redundant, i.e. the cone $\mathcal{C}_{\mathbf{v}^{=}}$ remains unchanged even after removing this inequality. Such redundancy can be detected, using e.g. the algebraic adjacency oracle described by Fukuda et al. in [FP96]. In the framework of Bremner et. al. [BSS09], this would require solving of a linear program, a situation which is alleviated in our case by the knowledge of both descriptions of $\mathcal{C}$. Cone $\mathcal{C}_{\mathbf{v}^{=}}$ is shown in fig. 3.4-d, and has extreme rays obtained from set Z in line 2 of symDD,

$$V^{\mathcal{C}_{\mathbf{v}^=}} = \{(1,1,0),(0,-1,0),(0,1,1)\}. \tag{3.19}$$

Next, we form inequalities $\mathbf{y}_0+\mathbf{y}_2+\mathbf{y}_3 \leq 0$ and $\mathbf{y}_0+\mathbf{y}_1+\mathbf{y}_2 \leq 0$, corresponding to other vertices in the orbit of $\mathbf{v}$ i.e. $\mathbf{v}^G \setminus \{\mathbf{v}\} = \{(0,1,1),(1,1,0)\}$. Substituting $\mathbf{y}_0 = -\mathbf{y}_1 - \mathbf{y}_3$ in these inequalities, we get inequalities $-\mathbf{y}_1+\mathbf{y}_2 \leq 0$ and $\mathbf{y}_2 - \mathbf{y}_3 \leq 0$. The set $\mathcal{A}$ is formed by checking if any rays in $V^{\mathcal{C}_{\mathbf{v}^=}}$ fail to satisfy these inequalities. In this case, ray $(0,1,1)$ fails inequality $-\mathbf{y}_1+\mathbf{y}_2 \leq 0$ and ray $(1,1,0)$ fails inequality $\mathbf{y}_2 - \mathbf{y}_3 \leq 0$, implying that $\mathcal{A} = \{(1,0,1,1),(1,1,1,0)\}$. Standard DD steps are now performed (line 8 of symDD), for inserting inequalities $-\mathbf{y}_1+\mathbf{y}_2 \leq 0$ (obtained by substituting $\mathbf{y}_0 = -\mathbf{y}_1 - \mathbf{y}_3$ in $\mathbf{y}_0+\mathbf{y}_2+\mathbf{y}_3 \leq 0$) and $\mathbf{y}_2 - \mathbf{y}_3 \leq 0$ in $\mathcal{C}_{\mathbf{v}^=}$ (obtained by substituting $\mathbf{y}_0 = -\mathbf{y}_1 - \mathbf{y}_3$ in $\mathbf{y}_0+\mathbf{y}_1+\mathbf{y}_2 \leq 0$), as shown in fig. 3.4-e and fig. 3.4-f respectively. The cone in fig. 3.4-f has extreme rays,

$$\{(0,-1,0),(0,0,1),(1,1,1),(1,0,0)\}, \tag{3.20}$$

which after prepending co-ordinate $\mathbf{y}_0 = -\mathbf{y}_1 - \mathbf{y}_3$ become,

$$\left\{ \begin{matrix} (0,0,-1,0),(-1,0,0,1), \\ (-2,1,1,1),(-1,1,0,0) \end{matrix} \right\} \subseteq (\mathbb{R}^4)^\circ \tag{3.21}$$

In the original space, these correspond to the inequalities,

$$\{-\mathbf{x}_2 \leq 0, \mathbf{x}_3 \leq 1, \mathbf{x}_1+\mathbf{x}_2+\mathbf{x}_3 \leq 2, \mathbf{x}_1 \leq 1\} \tag{3.22}$$

which are the inequalities incident to vertex $\mathbf{v} = (1,0,1)$ in the symmetric update. The inequalities incident to rays in $\mathbf{v}^G \setminus \{\mathbf{v}\}$ are all obtained as permutations of those incident to $\mathbf{v}$, according to lemma 7. Finally, the inequality description of the symmetric update is obtained by permuting

inequalities in (3.22) under S_3 and taking union with the inequalities associated with set $P \setminus N^{S_3}$.

$$H^{(l+1)} = \{0 \leq \mathbf{x}_i \leq 1, \forall i \in [3]\} \cup \left\{ \sum_{i \in [3]} \mathbf{x}_i \leq 2 \right\} \tag{3.23}$$

Example 10. When polytope $\mathcal{P}$ is a hypercube in $\mathbb{R}^{12}$, which we want to project down to $\mathbb{R}^9$, the sizes of DD steps under varying knowledge of symmetry is shown in fig. 3.5. In this case, the number of double description steps does not reduce i.e. $|\mathcal{A}| = |\mathbf{v}^G| - 1$, in line 6 of proc. symdd, for every symmetric update. The main tool to guage the efficiency of the symmetric updates deescribed in this section, we consider the sizes of extreme ray descriptions input to the double description steps. The average stepsizes under knowledge of no symmetries, cyclic group C_9 and symmetric group S_9 are 978.97, 245.38 and 120.54 respectively. Fig. 3.7 shows the number of LPs solved under varying knowledge of symmetry for projecting $\mathcal{P}$ to different dimensions, while fig. 3.6 shows the time required for projection vs the dimension under varying knowledge of symmetry.

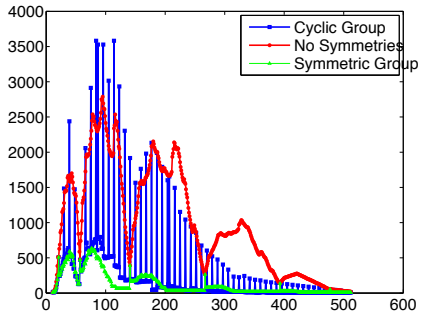

Figure 3.5: Sizes of double description steps for finding ith vertex of the projection of a 12-dimensional hypercube to 9 dimensions versus i, under varying knowledge of symmetry.

3.5 Symmetry exploitation in weighted sum rate computation and related problems

It is known that symmetry can substantially simplify linear programs through dimension reduction [BH09]. While in the symmetry exploiting CHM, we used only the knowledge of the symmetries of $\mathsf{proj}_k(\mathcal{P})$, for computation of weighted sum rate bounds, we can use the knowledge of symmetries of $\mathcal{P}$. Here, we are given an outer bound Γ_{out} on the entropy region, a HMSNC instance A, weights

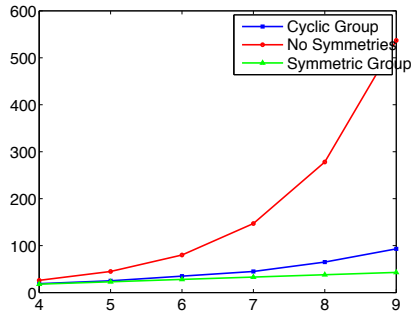

Figure 3.6: Number of LPs solved for projection of a 12-dimensional hypercube versus the dimension of projection, under varying knowledge of symmetry.

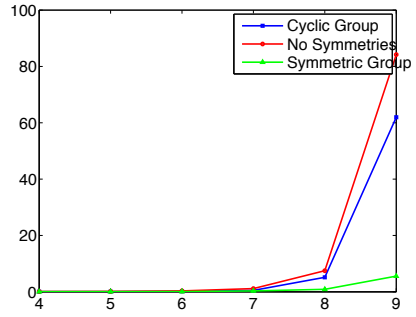

Figure 3.7: Time is seconds required for projection of a 12-dimensional hypercube versus the dimension of projection, under varying knowledge of symmetry.

$\lambda_1, \ldots, \lambda_k$ for each of the k sources and capacities $r_{k+1}, \ldots, r_N$ of the edges. The weighted sum-rate bound associated with Γ_{out}, is the solution to the following linear program:

$$\max_{\mathcal{P}} \sum_{i \in [k]} \lambda_i \omega_i \tag{3.24}$$

where,

$$\mathcal{P} \triangleq \{(\widetilde{\boldsymbol{\omega}}, \widetilde{\mathbf{r}}, \widetilde{\mathbf{h}}) \in \Gamma_{\mathsf{out}} \cap \mathcal{L}' \cap \mathcal{L}'' \mid \widetilde{\mathbf{r}}_i = \mathbf{r}_i, \forall i \in [N] \setminus [k]\} \tag{3.25}$$

A key consideration that enables the dimension reduction is the symmetries of the following vector,

$$\boldsymbol{\delta} = (\lambda_1, \ldots, \lambda_k, r_{k+1}, \ldots, r_N)^T \tag{3.26}$$

under the action of the symmetry group G of $\Gamma_{\mathsf{out}} \cap \mathcal{L}' \cap \mathcal{L}'' \subseteq \mathbb{R}^M$. If G is an ASG of $\mathcal{P}$, let G'be the subgroup of G s.t.

$$G' = \left\{ [\mathbf{b}, \mathbf{A}] \in G \left| \begin{array}{l} [\boldsymbol{\delta}^T \mathbf{0}_{M-N}^T](\mathbf{A} - \mathbb{I}_M) = \mathbf{0}_M, \\ \\ [\boldsymbol{\delta}^T \mathbf{0}_{M-N}^T]\mathbf{b} = 0 \end{array} \right. \right\} \tag{3.27}$$

If G' is the subgroup of G that fixes vector (3.26), the solution of the linear program in equation (3.24) can be obtained by solving the following linear program instead,

$$\max_{\mathsf{Fix}_{G'}(\mathcal{P})} \sum_{i \in [k]} \lambda_i \omega_i \tag{3.28}$$

where $\mathsf{Fix}_{G'}(\mathcal{P})$ is defined as,

$$\mathsf{Fix}_{G'}(\mathcal{P}) = \{ \mathbf{y} \in \mathcal{P} \, | \mathbf{A}\mathbf{y} + \mathbf{b} = \mathbf{y}, \ \forall [\mathbf{b}, \mathbf{A}] \in G' \} \, . \tag{3.29}$$

The dimension of $\mathsf{Fix}_{G'}(\mathcal{P})$ is expected to be much smaller than that of $\mathcal{P}$. For example, if G' is made up of permutation matrices, i.e. it is a subgroup of S_M, the dimension of $\mathsf{Fix}_{G'}(\mathcal{P})$ is equal to number of orbits of G' in set $[M]$. On the same lines as NSG, one can compute symmetry groups of access structures in secret sharing [Pad13], and directed graphs in the context of guessing games [Rii06]. Computation of lower bounds on worst case information ratio in secret sharing and upper bounds on guessing numbers of graphs, using information inequalities, has the same semantics as the computation of weighted sum-rate bounds in network coding, allowing symmetry to be exploited in identical manner. A daunting problem in all these computations is that of the need to solve LPs in exponentially many variables as compared to problem size. The authors believe that the study of higher symmetries, in form of RSG of Γ_N and non-Shannon outer bounds help can alleviate this situation.

3.6 ITCP

ITCP [JJ16] stands for the Information Theoretic Converse Prover. It is a GAP4 [GAP15] package that implements techniques discussed in this work. It uses inbuilt rational arithmetic of GAP (which is itself based on GMP [Gt12]) and exact rational linear programming via an interface to QSopt_ex linear programming solver [DWSD09]. The default outer bound on entropy function region, with respect to which EPCs and other bounds can be computed is the Shannon outer bound Γ_N. There is an inbuilt database of 215 non-Shannon inequalities that are inequivalent under random variable permutations, which includes the inequality of Zhang and Yeung [ZY97] and those of Dougherty,Freiling and Zeger [DFZ11]. The user can optionally specify a subset of these inequalities to be considered in the computation.

Example 11. This example demonstrates the computation of Network Symmetry Group in ITCP. The network coding instance used here is an IDSC [LWW15] system, that is constructed so that it has a desired symmetry group. As mentioned in §3.3.1, the network symmetry group is the direct product of two groups the form $G_1 \times G_2$ where group G_1 is a group of permutations of source labels while group G_2 is a group of permutations of edge labels. The instance considered here has size 8 while its NSG G has order 20, and is isomorphic to $S_2 \times D_5$, where $D5$ is the dihedral group corresponding to the symmetries of a regular pentagon. Starting from group G and 5 encoders, this instance can be constructed by, 1) adding decoders demanding sources $\{1, 2\}$, having access to a set in the orbit of $\{4, 5\}$ under G, and 2) adding decoders demanding sources $\{3\}$, having access to a set in the orbit of $\{4, 6\}$ under G.

Sample ITCP session for example 11

```
gap> # Define a size 8 IDSC instance
> idsc:=[ [ [ [ 1, 2, 3 ], [ 1, 2, 3, 4, 5, 6, 7, 8 ] ],\
>       [ [ 4, 5 ], [ 1, 2, 4, 5 ] ], [ [ 5, 6 ], [ 1, 2, 5, 6 ] ],\
>       [ [ 6, 7 ], [ 1, 2, 6, 7 ] ], [ [ 7, 8 ], [ 1, 2, 7, 8 ] ],\
>       [ [ 4, 8 ], [ 1, 2, 4, 8 ] ], [ [ 4, 6 ], [ 3, 4, 6 ] ],\
>       [ [ 5, 8 ], [ 3, 5, 8 ] ], [ [ 4, 7 ], [ 3, 4, 7 ] ],\
>       [ [ 5, 7 ], [ 3, 5, 7 ] ], [ [ 6, 8 ], [ 3, 6, 8 ] ] ], 3, 8 ];
gap> G:=NetSymGroup(idsc);
Group([ (5,8)(6,7), (4,5)(6,8), (4,6)(7,8), (1,2) ])
gap> Size(G);
20
```

Example 12. This example shows how EPOBs can be computed using ITCP. The HMSNC instance considered in this example is the Fano network, which is a well-known matroidal network [DFZ07], whose network symmetry group is trivial. The EPC is computed using the Shannon outer bound Γ_7. If we substitute $R_i = 1, \forall i \in [7] \setminus [3]$ in the rate region shown in the sample session, we get the region described by Dougherty, Freiling and Zeger in [DFZ15], which is a 3 dimensional cube.

Sample ITCP session for example 12

```
gap> # define a network instance (in this case, Fano network)
> F:=[ [ [ [ 1, 2 ], [ 1, 2, 4 ] ], [ [ 2, 3 ], [ 2, 3, 5 ] ],\
>       [ [ 4, 5 ], [ 4, 5, 6 ] ], [ [ 3, 4 ], [ 3, 4, 7 ] ],\
>       [ [ 1, 6 ], [ 3, 1, 6 ] ], [ [ 6, 7 ], [ 2, 6, 7 ] ],\
>       [ [ 5, 7 ], [ 1, 5, 7 ] ] ], 3, 7 ];;
gap> rlist:=NCRateRegionOB(F,true,[]);;
gap> Display(rlist[2]);
0 >= -w2
0 >= -w1
0 >= -w3
+R6 >= +w3
+R5 >= +w3
+R7 >= +w1
+R4 >= +w1
+R6 +R7 >= +w2 +w3
+R4 +R6 >= +w2 +w3
+R4 +R5 >= +w2 +w3
+R6 +R7 >= +w1 +w2
+R4 +R6 >= +w1 +w2
+R4 +R5 >= +w1 +w2
```

Example 13. This example also describes computation of EPOBs using ITCP, except for the IDSC instance in example 11, where the network symmetry group is non-trivial. The EPC is computed using the Shannon outer bound Γ_8. In such cases when NSG is non-trivial, ITCP outputs only one inequality per equivalence class under the action of NSG G. There are 18 such equivalence classes for the IDSC instance under consideration, as shown in the sample ITCP session, while there are a total of 94 distinct permuted forms of these inequalities that form the facets of $\mathcal{R}_{\mathsf{out}}$. A total of 119 LPs are solved during this computation, while if on decides to ignore symmetries, 207 LPs must be solved instead. The time required for computation of the EPOBs is 43.91 sec and 67.72 sec, with and without the knowledge of symmetries, respectively.

Example 14. This example considers the computation of upper bound on the sum rate of a size 5 HMSNC instance, with symmetry group of order 2, that contains permutations $\{(1), (3, 4)\}$. The

Sample ITCP session for example 13

```
gap> rlist1:=NCRateRegionOB2(idsc,true,[]);;
gap> Display(rlist1[2]);
0 >= -w2
0 >= -w3
+R4 >= 0
+R4 +R6 >= +w3
+R4 +R5 >= +w1 +w2
+R4 +1/2 R5 +1/2 R8 >= +w1 +w2 +1/2 w3
+1/2 R4 +1/2 R5 +1/2 R6 +1/2 R7 >= +w1 +w2 +1/2 w3
+2/3 R4 +2/3 R5 +1/3 R6 +1/3 R8 >= +w1 +w2 +2/3 w3
+2/3 R4 +1/3 R5 +1/3 R6 +1/3 R7 +1/3 R8 >= +w1 +w2 +2/3 w3
+1/2 R4 +1/2 R5 +1/2 R6 +1/4 R7 +1/4 R8 >= +w1 +w2 +3/4 w3
+R4 +1/2 R5 +1/2 R6 +1/2 R7 >= +w1 +w2 +w3
+R4 +1/2 R5 +1/2 R6 +1/2 R8 >= +w1 +w2 +w3
+R4 +1/3 R5 +1/3 R6 +1/3 R7 +1/3 R8 >= +w1 +w2 +w3
+2/3 R4 +2/3 R5 +1/3 R6 +2/3 R7 +1/3 R8 >= +w1 +w2 +4/3 w3
+R4 +1/2 R5 +1/2 R6 +R7 >= +w1 +w2 +3/2 w3
+R4 +1/2 R5 +1/2 R6 +1/2 R7 +1/2 R8 >= +w1 +w2 +3/2 w3
+2 R4 +R6 +R7 >= +w1 +w2 +2 w3
+R4 +R5 +R6 +R7 >= +w1 +w2 +2 w3
```

upper bound is computed with respect to Γ_5.

Sample ITCP session for example 14

```
gap> # define a network instance
> N:= [[ [ [ 1 ], [ 1, 3 ] ], [ [ 1 ], [ 1, 4 ] ],[ [ 1, 2, 5 ],\
>       [ 1, 2 ] ],[ [ 1, 2, 3 ], [ 2, 3 ] ],[ [ 2, 4 ], [ 1, 2, 4 ] ],\
>       [ [ 2, 3, 4, 5 ], [ 3, 4, 5 ] ]] , 2, 5 ];;
gap> ub:=NCSumRateUB(N,[1,1,1],[]);;
Original LP dimension...28
LP dimension after considering symmetries...22
gap> ub;
2
```

Example 15. This example shows how worst case information ratio lower bounds can be computed for a specified secret sharing access structure. The access structure considered here has authorized sets $\{\{2,3\},\{3,4\},\{4,5\}\}$, where there are 4 parties labeled $\{2,3,4,5\}$ while the dealer of secret is labeled 1. The lower bound is computed with respect to Γ_N, which is already known in the literature to be $\frac{3}{2}$, which is also achievable using multi-linear scheme (see [Pad13] §2.8).

Example 16. This example shows how upper bounds on the guessing number of a directed graph can be found using ITCP. The graph under consideration is the cycle graph C_5. The upper bound obtained using Γ_5, in this case, is $\frac{5}{2}$ (see eg. [ARS16]).

Sample ITCP session for example 15

```
gap> # define an access structure
> Asets:=[[2,3],[3,4],[4,5]];;
gap> lb:=SSWorstInfoRatioLB(Asets,5,[]);;
Original LP dimension...20
LP dimension after considering symmetries...12
gap> lb;
3/2
```

Sample ITCP session for example 16

```
gap> # define a directed graph (in this case, the cycle graph C5)
> C5:=[ [ 1, 2, 3, 4, 5 ],\
>   rec( 1 := [ 2, 5 ], 2 := [ 1, 3 ], 3 := [ 2, 4 ],\
>       4 := [ 3, 5 ], 5 := [ 4, 1 ] ) ];;
gap> ub:=GGnumberUB(C5,[]);;
Original LP dimension...25
LP dimension after considering symmetries...5
gap> ub;
5/2
```

Chapter 4: Conclusion and Future Directions

This thesis considered the problem of designing computer-assisted theorem provers for achievability and converse proofs in Multi-source Multi-sink Network Coding over Directed Acyclic Hypergraphs (HMSNC). The characterization of HMSNC rate regions in terms of the unknown region of entropic vectors was used as a foundation, and the problem computing achievability and converse proofs was posed as the problem of computing projections of certain inner and outer bounds on the region of entropic vectors.

Chapter 2 defined the existential and enumerative variants of the constrained linear representability problem for polymatroids, and showed that special instances of these problems include the construction of achievability proofs in network coding and secret sharing. An algorithm built from group theoretic techniques for combinatorial generation was developed to solve this problem, and an implementation of this algorithm in the GAP package ITAP accompanies the article. Several experiments with the developed enumerative method demonstrated its utility as well as improvement in runtime over competing methods for solving CLRPq-EX in some problems. As network coding and secret sharing constructions in general necessitate non-linear codes, a key future extension of the work will focus on finding nonlinear achievability constructions when linear ones fail. In this vein, the group theoretic method of combinatorial generation employed here, Leiterspiel, can also be adapted to efficiently generate such nonlinear dependence structures, as demonstrated in [64], [65]. As we approach problem instances larger than those currently within the computational reach of ITAP, there is a great scope for improvement in speed, via the use of Orbiter [Bet13], a dedicated software for combinatrial generation, and parallelization of the algorithms to solve CLRP.

In chapter 3, we considered the problem of computing explicit polyhedral outer bounds (EPOBs) on multi-source network coding rate regions. Algorithms for computing EPOBs that can exploit problem symmetry, were built from known polyhedral computation techniques in the literature. ITCP, a software implementation of the techniques discussed in this work in form of a GAP package,

was provided. It is a first of its kind application that can allows the user to computationally explore outer bounds on the rate regions of network coding instances of small to moderate size, and to verify conjectures up to a certain size of instances. An important future extension the work in chapter 3 is the characterization and exploitation of the group of restricted symmetries of Γ_N, which the authors believe to hold the key to determination of the bounds on network coding rate regions without having to solve linear programs with exponentially many variables in the instance size.

Bibliography

[ACLY00] Rudolf Ahlswede, Ning Cai, S-YR Li, and Raymond W Yeung. Network information flow. *IEEE Transactions on information theory*, 46(4):1204–1216, 2000.

[ACW16] Jayant Apte, Qi Chen, and John MacLaren Walsh. Symmetries in the entropy space, 2016. to appear, IEEE Information Theory Workshop, Cambridge, UK.

[ALW14] Jayant Apte, Congduan Li, and J.M. Walsh. Algorithms for computing network coding rate regions via single element extensions of matroids. In *IEEE Int. Symp. Information Theory (ISIT)*, pages 2306–2310, June 2014.

[Apt14] Jayant Apte. chm-An Implementation of Convex Hull Method, 2014. http://www.ece.drexel.edu/walsh/aspitrg/software.html.

[ARS16] Ross Atkins, Puck Rombach, and Fiona Skerman. Guessing numbers of odd cycles. *CoRR*, abs/1602.03586, 2016.

[AW15a] J. Apte and J. M. Walsh. Symmetry in network coding. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 376–380, June 2015.

[AW15b] Jayant Apte and John MacLaren Walsh. Exploiting symmetry in computing polyhedral bounds on network coding rate regions. In *2015 International Symposium on Network Coding, NetCod 2015, Sydney, Australia, June 22-24, 2015*, pages 76–80, 2015.

[BBF+06] A. Betten, M. Braun, H. Fripertinger, A. Kerber, A. Kohnert, and A. Wassermann. *Error-Correcting Linear Codes: Classification by Isometry and Applications*. Algorithms and Computation in Mathematics. Springer Berlin Heidelberg, 2006.

[BCH73] John E. Blackburn, Henry H. Crapo, and Denis A. Higgs. A catalogue of combinatorial geometries. *Mathematics of Computation*, 27(121):pp. 155–166, 1973.

[BDSSV95] C. Blundo, A. De Santis, D.R. Stinson, and U. Vaccaro. Graph decompositions and secret sharing schemes. *Journal of Cryptology*, 8(1):39–64, 1995.

[Bei11] Amos Beimel. Secret-sharing schemes: A survey. In YeowMeng Chee, Zhenbo Guo, San Ling, Fengjing Shao, Yuansheng Tang, Huaxiong Wang, and Chaoping Xing, editors, *Coding and Cryptology*, volume 6639 of *Lecture Notes in Computer Science*, pages 11–46. Springer Berlin Heidelberg, 2011.

[Bet13] Anton Betten. Orbiter – a program to classify discrete objects, 2013. http://www.math.colostate.edu/ betten/orbiter/orbiter.html.

[BH09] R. Bödi and K. Herr. Symmetries in linear and integer programs. 2009.

[BHJ13] Richard Bödi, Katrin Herr, and Michael Joswig. Algorithms for highly symmetric linear and integer programs. *Mathematical Programming*, 137(1-2):65–90, 02 2013.

[BI93] Michael Bertilsson and Ingemar Ingemarsson. A construction of practical secret sharing schemes using linear block codes. In Jennifer Seberry and Yuliang Zheng, editors, *Advances in Cryptology - AUSCRYPT 1992*, volume 718 of *Lecture Notes in Computer Science*, pages 67–79. Springer Berlin Heidelberg, 1993.

[BL90] J. Benaloh and J. Leichter. Generalized secret sharing and monotone functions. In *Proceedings on Advances in Cryptology*, CRYPTO '88, pages 27–35, New York, NY, USA, 1990. Springer-Verlag New York, Inc.

[Bre96] David Bremner. *Incremental convex hull algorithms are not output sensitive*, pages 26–35. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996.

[BSS09] David Bremner, Mathieu Dutour Sikirić, and Achill Schürmann. Polyhedral representation conversion up to symmetries, 2009.

[CdG12] M. Costantini and W. de Graaf. singular, the gap interface to singular, Version 12.04.28, Apr 2012. GAP package.

[CLO07] David A. Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra, 3/e (Undergraduate Texts in Mathematics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[Cona] Congduan Li, Steven Weber, and John MacLaren Walsh. On Multilevel Diversity Coding Systems. *IEEE Trans. Inform. Theory*. Submitted on July 21, 2014. Reviews received January 6, 2016, revised April 4, 2016.

[Conb] Congduan Li, Steven Weber, and John Walsh. On Multi-source Networks: Enumeration, Rate Region Computation, and Hierarchy. *IEEE Trans. Inform. Theory*. Submitted July 21, 2015.

[Con12] Congduan Li, John MacLaren Walsh, Steven Weber. A computational approach for determining rate regions and codes using entropic vector bounds. In *50th Annual Allerton Conference on Communication, Control and Computing*, October 2012.

[Con13] Congduan Li, Jayant Apte, John MacLaren Walsh, Steven Weber. A new computational approach for determining rate regions and optimal codes for coded networks. In *The 2013 IEEE International Symposium on Network Coding (NetCod 2013)*, June 2013.

[Con15] Congduan Li. *On Multi-source Multi-Sink Hyperedge Networks: Enumeration, Rate Region Computation, and Hierarchy*. PhD thesis, Drexel University, Philadelphia, PA, 2015.

[Cra65] Henry H. Crapo. Single-element extensions of matroids. *J. Res. Natl. Bur. Standards, Sec. B: Math. & Math. Phys.*, 69B(1-2):55–65, 1965.

[Csi13] L. Csirmaz. Information inequalities for four variables. Avaible Online: https://www.eprints.renyi.hu/65/1/benson.pdf, 2013.

[CY02a] Ning Cai and R.W. Yeung. Secure network coding. In *Information Theory, 2002. Proceedings. 2002 IEEE International Symposium on*, pages 323–, 2002.

[CY02b] T. H. Chan and R. W. Yeung. On a relation between information inequalities and group theory. *IEEE Transactions on Information Theory*, 48(7):1992–1995, Jul 2002.

[DFZ05] Randall Dougherty, Christopher Freiling, and Kenneth Zeger. Insufficiency of linear coding in network information flow. *IEEE Transactions on Information Theory*, 51(8):2745–2759, 2005.

[DFZ06] R. Dougherty, C. Freiling, and K. Zeger. Six new non-shannon information inequalities. In *Information Theory, 2006 IEEE International Symposium on*, pages 233–236, July 2006.

[DFZ07] R. Dougherty, C. Freiling, and K. Zeger. Networks, matroids, and non-shannon information inequalities. *Information Theory, IEEE Transactions on*, 53(6):1949–1969, 2007.

[DFZ09] R. Dougherty, C. Freiling, and K. Zeger. Linear rank inequalities on five or more variables. *arXiv cs.IT/0910.0284v3*, 2009.

[DFZ11] R. Dougherty, C. Freiling, and K. Zeger. Non-shannon information inequalities in four random variables. *CoRR*, abs/1104.3602, 2011.

[DFZ15] Randall Dougherty, Chris Freiling, and Kenneth Zeger. Achievable rate regions for network coding. *IEEE Transactions on Information Theory*, 61(5):2488–2509, 2015.

[DGPS15] Wolfram Decker, Gert-Martin Greuel, Gerhard Pfister, and Hans Schoenemann. Singular 4-0-2 — A computer algebra system for polynomial computations, 2015.

[DJM98] Martenvan Dijk, Wen-Ai Jackson, and Keith M. Martin. A general decomposition construction for incomplete secret sharing schemes. *Designs, Codes and Cryptography*, 15(3):301–321, 1998.

[Dou14] R. Dougherty. Computations of linear rank inequalities on six variables. In *Information Theory (ISIT), 2014 IEEE International Symposium on*, pages 2819–2823, June 2014.

[DWSD09] David Applegate, William Cook, Sanjeeb Dash, and Daniel Espinoza. Qsopt-exact rational lp solver, 2009. http://www.math.uwaterloo.ca/~bico/qsopt/ex/.

[Eri] Eric W. Weisstein. Sequence A055545, Number of matroids on n points. The On-Line Encyclopedia of Integer Sequences. http://oeis.org/A055545.

[F. 07] F. Matúš. Infinitely Many Information Inequalities. In *IEEE International Symposium on Information Theory (ISIT)*, pages 41–44, June 2007.

[FC13] T. Fritz and R. Chaves. Entropic inequalities and marginal problems. *IEEE Transactions on Information Theory*, 59(2):803–817, Feb 2013.

[FP96] K. Fukuda and A. Prodon. Double description method revisited. *Combinatorics and computer science*, pages 91–111, 1996.

[Fuj78] S. Fujishige. Polymatroidal dependence structure of a set of random variables. *Information and Control*, 39:55–72, 1978.

[Fuk04] K. Fukuda. Frequently asked questions in polyhedral computation, 2004. ftp://ftp.ifor.math.ethz.ch/pub/fukuda/reports/polyfaq040618.pdf.

[GAP15] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.7.8*, 2015.

[GGW14] Jim Geelen, Bert Gerards, and Geoff Whittle. Solving rotaś conjecture. *Notices of the AMS*, 61(7):736–743, 2014.

[Gt12] Torbjrn Granlund and the GMP development team. *GNU MP: The GNU Multiple Precision Arithmetic Library*, 5.0.5 edition, 2012. `http://gmplib.org/`.

[HLWY13] M. He, Z. Li, C. Wu, and X. Yin. On uniform matroidal networks. In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 2730–2734, July 2013.

[HMK+06] Tracey Ho, Muriel Médard, Ralf Koetter, David R Karger, Michelle Effros, Jun Shi, and Ben Leong. A random linear network coding approach to multicast. *IEEE Transactions on Information Theory*, 52(10):4413–4430, 2006.

[HRAS00] D. Hammer, A. Romashchenko, and N. Vereshchagin A. Shen. Inequalities for shannon entropy and kolmogorov complexity. *Journal of Computer and System Science*, 60(2):442–464, April 2000.

[HY97] Ka Pun Hau and R. W. Yeung. Multilevel diversity coding with three encoders. In *Information Theory. 1997. Proceedings., 1997 IEEE International Symposium on*, pages 440–, Jun 1997.

[Ing71] A. W. Ingleton. Representation of matroids. *Combin. Math. Appl.*, pages 149–167, 1971.

[JJ15] Jayant Apte and John MacLaren Walsh. Information theoretic achievability prover -a gap4 package, 2015. http://www.ece.drexel.edu/walsh/aspitrg/software.html.

[JJ16] Jayant Apte and John MacLaren Walsh. Information Theoretic Convere Prover - A GAP4 Package, 2016. https://github.com/jayant91089/itcp.

[JSC+05] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. M. G. M. Tolhuizen. Polynomial time algorithms for multicast network code construction. *IEEE Transactions on Information Theory*, 51(6):1973–1982, June 2005.

[Kin11] Ryan Kinser. New inequalities for subspace arrangements. *Journal of Chemical Thermodynamics*, 118:152–161, 2011.

[KM03] R. Koetter and M. Medard. An algebraic approach to network coding. *IEEE/ACM Transactions on Networking*, 11(5):782–795, Oct 2003.

[KS03] Volker Kaibel and Alexander Schwartz. On the complexity of polytope isomorphism problems. *Graphs and Combinatorics*, 19(2):215–230, 2003.

[LAWW13] Congduan Li, J. Apte, J.M. Walsh, and S. Weber. A new computational approach for determining rate regions and optimal codes for coded networks. In *IEEE Int. Symp. Network Coding (NetCod)*, pages 1–6, 2013.

[LL93] C. Lassez and J-L. Lassez. Quantifier elimination for conjunctions of linear constraints via a convex hull algorithm. In Donald, Kapur, and Mundy, editors, *Symbolic and Numerical Computation for Artificial Intelligence*. Academic Press, 1993.

[LL04] April Rasala Lehman and Eric Lehman. Complexity classification of network information flow problems. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '04, pages 142–150, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics.

[LWW12] Congduan Li, J. M. Walsh, and S. Weber. Computational approaches for determining rate regions and codes using entropic vector bounds. In *50th Annual Allerton Conference on Communication, Control and Computing, 2012*, pages 1 –9, Oct 2012.

[LWW15] Congduan Li, Steven Weber, and John MacLaren Walsh. Computer aided proofs for rate regions of independent distributed source coding problems. In *2015 International Symposium on Network Coding (NetCod)*, pages 81–85. IEEE, 2015.

[Mara] Marcel Wild. Sequence A076766, Number of nonisomorphic binary matroids on an n-set., The On-Line Encyclopedia of Integer Sequences. http://oeis.org/A076766.

[Marb] Marcel Wild. Sequence A076892, Number of nonisomorphic ternary matroids on an n-set., The On-Line Encyclopedia of Integer Sequences. http://oeis.org/A076892.

[Max] Max Alekseyev. Sequence A256157, Number of 2-polymatroids on n unlabeled points.., The On-Line Encyclopedia of Integer Sequences. http://oeis.org/A256157.

[MMIB12] Yoshitake Matsumoto, Sonoko Moriyama, Hiroshi Imai, and David Bremner. Matroid enumeration for incidence geometry. *Discrete Comput. Geom.*, 47(1):17–43, January 2012.

[MR08] Dillon Mayhew and Gordon F. Royle. Matroids with nine elements. *Journal of Combinatorial Theory, Series B*, 98(2):415 – 431, 2008.

[MR14] Vijayvaradharaj T. Muralidharan and B. S. Rajan. Linear network coding, linear index coding and representable discrete polymatroids, 2014. Available Online: http://arxiv.org/abs/1306.1157.

[MTD08] S. Mohajer, C. Tian, and S. Diggavi. Asymmetric gaussian multiple descriptions and asymmetric multilevel diversity coding. In *2008 IEEE International Symposium on Information Theory*, pages 1992–1996, July 2008.

[Oxl11] J. G. Oxley. *Matroid Theory*. Oxford University, 2011.

[Pad13] C. Padró. *Lecture Notes in Secret Sharing*. Central European University, 2013, 2013. Available: http://www-ma4.upc.edu/ cpadro/arc02v03.pdf.

[Pen14] R.A. Pendavingh. On the evaluation at $(-i, i)$ of the tutte polynomial of a binary matroid. *Journal of Algebraic Combinatorics*, 39(1):141–152, 2014.

[PR97] E. Petrank and R.M. Roth. Is code equivalence easy to decide? *Information Theory, IEEE Transactions on*, 43(5):1602–1604, Sep 1997.

[PV+13] Rudi Pendavingh, Stefan Van Zwam, et al. *Sage Matroid Package, included in Sage Mathematics Software 5.11*. The Sage Matroid Development Team, 2013. http://www.sagemath.org.

[Rad57] R. Rado. Note on independence functions. *Proceedings of the London Mathematical Society*, s3-7(1):300–320, 1957.

[Reh] Rehn, T., Schürmann A. *SymPol Manual*. 0.1 edition.

[Reh10] T. Rehn. Polyhedral description conversion up to symmetries. Master's thesis, Universität Magdeburg, November 2010.

[RES08] R. Pulikkoonattu, E. Perron, and S. Diggavi. X-information theoretic inequality prover(xitip), 2008. http://xitip.epfl.ch/.

[Rii06] S. Riis. Information flows, graphs and their guessing numbers. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, 2006 4th International Symposium on*, pages 1–9, April 2006.

[Roc70] RT Rockefellar. *Convex Analysis.* Princeton Univ. Press, 1970.

[Rot71] Gian-Carlo Rota. Combinatorial theory, old and new. In *Actes du Congrès International des Mathématiciens (Nice, 1970), Tome 3*, pages 229–233. Gauthier-Villars, Paris, 1971.

[RYH97] J. R. Roche, R. W. Yeung, and Ka Pun Hau. Symmetrical multilevel diversity coding. *IEEE Transactions on Information Theory*, 43(3):1059–1064, May 1997.

[Sav14] Thomas J. Savitsky. Enumeration of 2-polymatroids on up to seven elements, 2014. Available Online: http://arxiv.org/abs/1401.8006.

[Sch90] Bernd Schmalz. Verwendung von Untergruppenleitern zur Bestimmung von Doppelnebenklassen. (Use of subgroup ladders for the determination of double cosets). *Bayreuther Math. Schr.*, 31:109–143, 1990.

[Ser03] Akos Seress. *Permutation Group Algorithms*. Cambridge University Press, 2003. Cambridge Books Online.

[Sha79] A. Shamir. How to share a secret. *Commu. Assoc. Comput.*, 22:612–613, 1979.

[ST10] Abhay T. Subramanian and Andrew Thangaraj. Path gain algebraic formulation for the scalar linear network coding problem. *IEEE Transactions on Information Theory*, 56(9):4520–4531, 2010.

[Sti94] Douglas R. Stinson. Decomposition constructions for secret-sharing schemes. *IEEE Transactions on Information Theory*, 40(1):118–125, 1994.

[TGC11] S. Thakor, A. Grant, and T. Chan. On complexity reduction of the lp bound computation and related problems. In *2011 International Symposium on Networking Coding*, pages 1–6, July 2011.

[Tia14] Chao Tian. Characterizing the rate region of the (4, 3, 3) exact-repair regenerating codes. *IEEE Journal on Selected Areas in Communications*, 32(5):967–975, 2014.

[vD97] Marten van Dijk. A linear construction of secret sharing schemes. *Designs, Codes and Cryptography*, 12(2):161–201, 1997.

[vDKST06] Marten van Dijk, Tom Kevenaar, Geert-Jan Schrijen, and Pim Tuyls. Improved constructions of secret sharing schemes by applying (λ, ω)-decompositions. *Inf. Process. Lett.*, 99(4):154–157, August 2006.

[W. 08] W. Xu, J. Wang, J. Sun. A projection method for derivation of non-Shannon-type information inequalities. In *IEEE International Symposium on Information Theory (ISIT)*, pages 2116 – 2120, 2008.

[Wil94] Marcel Wild. Enumeration of binary and ternary matroids and other applications of the brylawski-lucas theorem. *Preprint Nr. 1693, Tech. Hochschule Darmstadt*, 1994.

[Yeu97] R. W. Yeung. A framework for linear information inequalities. *IEEE Transactions on Information Theory*, 43(6):1924–1934, November 1997.

[Yeu08] R. W. Yeung. *Information Theory and Network Coding*. Springer, 2008.

[YR08] Ying-On Yan and Raymond Yeung. Itip-information theoretic inequality prover, 2008. http://user-www.ie.cuhk.edu.hk/ ITIP/.

[YYZ12] X. Yan, R.W. Yeung, and Zhen Zhang. An implicit characterization of the achievable rate region for acyclic multisource multisink network coding. *IEEE Trans. on Inform. Theory*, 58(9):5625–5639, 2012.

[Z. 98] Z. Zhang and R W. Yeung. On Characterization of Entropy Function via Information Inequalities. *IEEE Transactions on Information Theory*, 44(4), July 1998.

[Zie95] G.M. Ziegler. *Lectures on Polytopes*. Springer-Verlag, 1995.

[ZY97] Z. Zhang and R. W. Yeung. A non-shannon-type conditional inequality of information quantities. *IEEE Transactions on Information Theory*, 43(6):1982–1986, Nov 1997.

Appendix A: Transformation of a HMSNC instance into a NCDAMG instance

We specify a NCDAMG instance A' as per the terminology in [ST10], by a tuple $(\mathscr{V}', \mathscr{E}', \mathscr{S}', \mathscr{T}', g)$ where $(\mathscr{V}', \mathscr{E}')$ is a directed acyclic multigraph, $\mathscr{S}', \mathscr{T}' \subseteq V'$ are sets of source and sink nodes respectively, and g is the demand function assigning exactly one member of $\mathscr{S}$ to each $t \in \mathscr{T}$. Each edge $e \in \mathscr{E}'$ is a triplet (v_1, v_2, c) where $v_1, v_2 \in \mathscr{V}'$ and c is the edge label or color. Algorithm 4 accepts a HMSNC instance in form of the associated constraints $\mathcal{L}_i, i \in [3]$, number of source k, number of random variables N and a rate vector $(r_1, \ldots, r_N)$. At the begining, all members of A' are empty. Algorithm 4 populates various member of A' and also maintains a function $\mathsf{msg2node} : \mathscr{X} \rightarrow \mathscr{V}'$ where $\mathscr{X} \subseteq [N]$. For message labels i not in $\mathscr{X}$ at any point, we say that $\mathsf{msg2node}(i)$ is not defined. For each constraint l in $\mathcal{L}_2 \cup \mathcal{L}_3$, we use functions $\mathsf{imsg}(l)$ and $\mathsf{omsg}(l)$ to refer to the input and output message labels involved in constraint l. At the begining $\mathsf{msg2node}(i)$ is undefined for each $i \in [N]$. First for each source message $i \in [k]$, r_i nodes are added to sets $\mathscr{V}'$ and $\mathscr{S}'$ (lines 2-4, fig. A.1). These are the source nodes of NCDAMG instance A'. Second, it goes through constraints $\mathcal{L}_2$, considering at each iteration of the while loop at lines 5-9, a constraint l s.t. $\mathsf{msg2node}(i)$ is defined for each $i \in \mathsf{imsg(l)}$. The new nodes and edges, added to $\mathscr{V}'$ and $\mathscr{E}'$ using procedure convertL2, are summarized as a gadget in figure A.1. Finally, for each decoding constraint $l \in \mathcal{L}_3$, if $|\mathsf{omsg}(l)| = 1$ a single deoder node is added whereas if $|\mathsf{omsg}(l)| > 1$ multiple decoder nodes are added to $\mathcal{V}'$, in while loop at lines 10-14 using procedure convertL3. The demand function g for these decoder nodes is also set during the same procedure. Throughout algorithm 4, the number of parallel edges added between nodes depends on the rate vector. Thus, given a HMSNC instance A and a rate vector $(r_1, \ldots, r_N)$, algorithm 4 constructs a NCDAMG instance A' such that the following holds:

Lemma 8. *Rate vector $(r_1, \ldots, r_N)$ is achievable in A with vector linear network codes if A' is scalar linear solvable.*

Proof: If A' is scalar linear solvable, there exists a representable matroid M associated with the

scalar linear solution. In the matrix representation of M, each column is associated with an edge in A'. We can create a N-subspace arrangement $\{V_1, \ldots, V_N\}$ from this matroid s.t. $V_i, i \in [N]$ is the span of columns associated with edge incoming to $\mathsf{msg2node}(i)$. ■

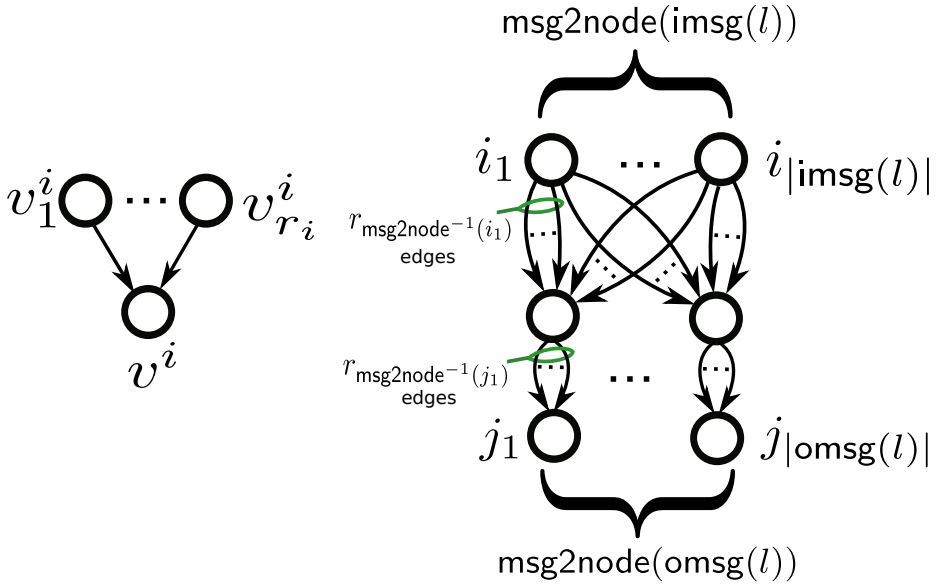

Figure A.1: (left) Gadget used by algorithm 4 for source messages and, (right) gadget used for intermmediate constraints $l \in \mathcal{L}_2$

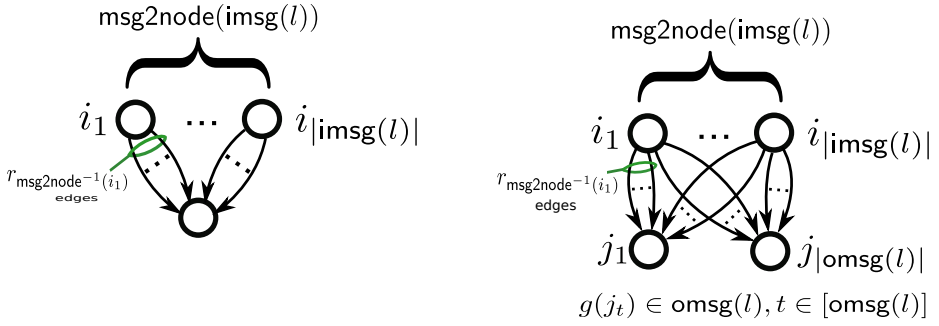

Figure A.2: Gadgets used by algorithm 4 for decoder constraints. (left) the case $|\mathsf{omsg}(l)| = 1$ and (right) the case $|\mathsf{omsg}(l)| > 1$

Input: Sets $\mathcal{L}_1, \mathcal{L}_2$ and $\mathcal{L}_3$ associated with HMSNC instance A, no. of sources k, no. of variables N, rate vector $\mathbf{r} = (r_1, \ldots, r_N)$
Output: A NCDAMG instance $\mathsf{A}' = (\mathscr{V}', \mathscr{E}', \mathscr{S}', \mathscr{T}', g)$

$\mathscr{V}' \leftarrow \emptyset$, $\mathscr{E}' \leftarrow \emptyset$, $g \leftarrow$ empty function, $\mathsf{msg2node} \leftarrow$ empty function, $\mathcal{L} \leftarrow \emptyset$
foreach $i \in [k]$ **do**
  $\mathscr{V}' \leftarrow \mathscr{V}' \cup \{v_1^i, \ldots, v_{r_i}^i\}$, $\mathscr{S}' \leftarrow \mathscr{S}' \cup \{v_1^i \ldots, v_{r_i}^i\}$, $\mathscr{V}' \leftarrow \mathscr{V}' \cup v^i$, $\mathsf{msg2node}(i) \leftarrow v^i$
end
while $\mathcal{L}_2 \not\subseteq \mathcal{L}$ **do**
  $l \leftarrow$ constraint in $\mathcal{L}_2$ not in $\mathcal{L}$ with $\mathsf{msg2node}(i)$ defined for each $i \in \mathsf{imsg}(l)$
  $(\mathsf{msg2node}, \mathscr{V}', \mathscr{E}') \leftarrow convertL2(l, \mathsf{msg2node}, \mathscr{V}', \mathscr{E}', \mathbf{r})$
  $\mathcal{L} \leftarrow \mathcal{L} \cup \{l\}$
end
while $\mathcal{L}_3 \not\subseteq \mathcal{L}$ **do**
  $l \leftarrow$ constraint in $\mathcal{L}_3$ not in $\mathcal{L}$
  $(\mathsf{msg2node}, \mathscr{V}', \mathscr{E}', \mathscr{T}', g) \leftarrow convertL3(l, \mathsf{msg2node}, \mathscr{V}', \mathscr{E}', \mathscr{T}', g, , \mathbf{r})$
  $\mathcal{L} \leftarrow \mathcal{L} \cup \{l\}$
end
return $(\mathscr{V}', \mathscr{E}', \mathscr{S}', \mathscr{T}', g)$

Algorithm 4: Algorithm to transform a HMSNC instance to a NCDAMG instance

foreach $o \in \mathit{omsg}(l)$ **do**
  $\mathscr{V}' \leftarrow \mathscr{V}' \cup \{v_1^o\}$, $\mathscr{V}' \leftarrow \mathscr{V}' \cup \{v_2^o\}$, $\mathsf{msg2node}(o) \leftarrow v_2^o$
  foreach $i \in \mathit{imsg}(l)$ **do**
    $\mathscr{E}' \leftarrow \mathscr{E}' \cup \{(\mathsf{msg2node}(i), v_1^o, c_1), \ldots, (\mathsf{msg2node}(i), v_1^o, c_{r_i})\}$
  end
  $\mathscr{E}' \leftarrow \mathscr{E}' \cup \{(v_1^o, v_2^o, c_1), \ldots, (v_1^o, v_2^o, c_{r_o})\}$
end
return $\mathit{msg2node}, \mathscr{V}', \mathscr{E}'$

Procedure convertL2$(l, \mathsf{msg2node}, \mathscr{V}', \mathscr{E}', , \mathbf{r})$

if $|\mathit{omsg}(l)|$ *is 1* **then**
  $\mathscr{V}' \leftarrow \mathscr{V}' \cup \{v^l\}$
  foreach $i \in \mathit{imsg}(l)$ **do**
    $\mathscr{E}' \leftarrow \mathscr{E}' \cup \{(\mathsf{msg2node}(i), v^l, c_1), \ldots, (\mathsf{msg2node}(i), v^l, c_{r_i})\}$
    $g(v^l) \leftarrow \mathsf{omsg}(l)$
  end
  $\mathcal{T}' \leftarrow \mathscr{T}' \cup \{v^l\}$
end
else
  foreach $o \in \mathit{omsg}(l)$ **do**
    $\mathscr{V}' \leftarrow \mathscr{V}' \cup \{v_o^l\}$
    foreach $i \in \mathit{imsg}(l)$ **do**
      $\mathscr{E}' \leftarrow \mathscr{E}' \cup \{(\mathsf{msg2node}(i), v_o^l, c_1), \ldots, (\mathsf{msg2node}(i), v_o^l, c_{r_i})\}$
      $g(v_o^l) \leftarrow \{o\}$
    end
    $\mathscr{T}' \leftarrow \mathscr{T}' \cup \{v_o^l\}$
  end
end
return $\mathit{msg2node}, \mathscr{V}', \mathscr{E}', \mathscr{T}, g$

Procedure convertL3$(l, \mathsf{msg2node}, \mathscr{V}', \mathscr{E}', \mathscr{T}, g, \mathbf{r})$