On the Complexity of Computing the Shannon Outer Bound to a Network Coding Capacity Region

Yirui Liu and John MacLaren Walsh
Department of Electrical and Computer Engineering
Drexel University
Philadelphia, PA 19104, USA
Email: {yl926, jmw96}@drexel.edu

***Abstract*—A new method is presented, consisting of exclusively simple linear algebra computations, for computing the linear programming Shannon outer bound to the network coding capacity region of a directed hypergraph network. This linear algebraic formulation enables a new upper bound on the worst case complexity of computing the Shannon outer bound to a network coding capacity region to be determined.**

I. Introduction

Determining the capacity regions of networks under network coding form an important class of fundamental problems in information theory and coding. From an applied perspective, these problems are key to determining fundamental limits, and practical designs that approach them, for engineering problems ranging from exact repair distributed information storage [1]–[4], to index coding and coded caching [5]–[8], to delay mitigating codes and rate delay tradeoffs for streaming information [9]–[11]. From a fundamental perspective, formal equivalences between solving all of these problems and determining all fundamental inequalities in information theory and some fundamental subgroup inequalities in group theory, have been established.

A key result [12], enables inner and outer bounds to the network coding capacity region to obtained via the region of entropic vectors. Algorithms [13]–[15], and their practical implementations in software [16]–[18], which can compute complicated network coding capacity regions of small hypergraph networks [19], were created by building upon this result. In particular, the information theoretic converse prover (ITCP) [16] enables the Shannon/Linear programming outer bound to a capacity region to be computed through a process of polyhedral projection, while the information theoretic achievability prover (ITAP) [17] enables the rate region achievable with a class of codes to be determined. When the bounds from these two pieces of software match, as has been shown to be true for all minimal hypergraph networks with a number of source and edges at most 5 [19], the capacity region, its proof, and its efficient codes have been determined. ITCP works via a process of polyhedral projection, and it has been shown that it can exploit powerful notions of problem symmetry to reduce the number of steps and dimensions in the polyhedral projection process [13].

While specific examples have rather thoroughly demonstrated complexity reductions by exploiting symmetry in real world algorithms pragmatically [13], [20] in the form of runtime, as well as the potential of divide-and-conquer style methods for reducing the complexity by breaking apart the problem into pieces [19], [21], [22], theorists have repeatedly expressed interest in overall worst case complexity bounds for computing the LP outer bound, and how this complexity depends on the network structure and various measures of problems size. A common argument made has been that this worst case complexity bound for algorithms negligent of symmetry and hierarchy is necessary to be best quantify the gains that symmetry and hierarchy exploitation enable.

Bearing this in mind, this paper provides a method of computing the LP outer bound selected not for its computational efficiency in a real polyhedral computation implementation, as has been the motivation in [13], [15], but rather selected for its ease of obtaining an upper bound on the number of computations it requires. We have attempted to express the necessary computations in as explicit a manner as possible by referring to linear algebra and avoiding the linear programing steps which are typically encountered in an efficient linear programming process. Utilizing this alternative formulation, we are able to give an upper bound to the worst case complexity of computing the linear programming outer bound.

II. Background

This paper focuses on multi-source multi-sink network coding problems on acyclic networks that have directed hyperedges, which we hereafter refer to as the MSNC-DH problems. *For the sake of brevity, we will use, without repeating some definitions, the same notation as [19],* where a MSNC-DH problem is be represented as a tuple $A = (\mathcal{S}, \mathcal{G}, \mathcal{T}, \mathcal{E}_{\mathcal{S}}, \mathcal{E}_U, \beta(t))$, where $\mathcal{S}, \mathcal{G}$ and $\mathcal{T}$ denotes the set of source nodes, intermediate nodes and sink nodes respectively, $\mathcal{E}_U$ the set of edges outgoing from intermediate nodes (hereafter $\mathcal{E}$) and $\beta(t)$ the demand of the sink node t. We will denote $\mathcal{V} = \mathcal{G} \cup \mathcal{T}$ the set of intermediate and sink nodes, and assume that the network is minimal so that no two of these nodes have the same collections of hyperedges as inputs.

A. Rate region of a MSNC-DH Network

Yan *et al.'s* celebrated paper [12], showed that the capacity region of a network can be expressed implicitly in terms of region of *entropic* vectors Γ_N^* [12] and some network constraints. As proved in Theorem 1 in [12] and later extended in [23] the rate region $\mathcal{R}(A)$ of network A can be expressed as,

$$\mathcal{R}(A) = \text{Proj}_{\boldsymbol{r},\boldsymbol{\omega}}(\overline{\text{con}(\Gamma_N^* \cap \mathcal{L}_{13})} \cap \mathcal{L}_{4'5}) \quad (1)$$

where $\text{con}(\mathcal{B})$ is the conic hull of $\mathcal{B}$, and $\text{Proj}_{\boldsymbol{r},\boldsymbol{\omega}}(\mathcal{B})$ is the projection of $\mathcal{B}$ onto the coordinates $[\boldsymbol{r}^T, \boldsymbol{\omega}^T]^T$ where $\boldsymbol{r} = [R_e | e \in \mathcal{E}_U]$ and $\boldsymbol{\omega} = [\omega_s | s \in \mathcal{S}]$. Further, Γ_N^* and $\mathcal{L}_i$, $i = 1, 3, 4', 5$ can be viewed as subsets of $\mathbb{R}^L$, $L = 2^N - 1 + N$, $N = |\mathcal{S}| + |\mathcal{E}|$, with $\mathbf{h} \in \mathbb{R}^{2^N - 1}$ indexed by subsets of N as is usual in entropic vectors, $\boldsymbol{r} \in \mathbb{R}^{|\mathcal{E}_U|}$ playing the role of the capacities of edges, and $\boldsymbol{\omega} \in \mathbb{R}^{|\mathcal{S}|}$ playing the role of source rates, by creating the stacked vector $\mathbf{x} = [\boldsymbol{r}^T, \boldsymbol{\omega}^T, \mathbf{h}^T]^T$. $\mathcal{L}_i$, $i = 1, 3, 4', 5$ are network constraints which can be written as,

$$\mathcal{L}_1 = \{\mathbf{x} \in \mathbb{R}^M |\ h_{\mathcal{S}} = \sum_{s \in \mathcal{S}} h_s\} \quad (2)$$

$$\mathcal{L}_3 = \{\mathbf{x} \in \mathbb{R}^M |\ h_{\text{Out}(i)|\text{In}(i)} = 0,\ \forall i \in \mathcal{G}\} \quad (3)$$

$$\mathcal{L}_{4'} = \{\mathbf{x} \in \mathbb{R}^M |\ h_{U_e} \leq R_e, h_s \geq \omega_s, e \in \mathcal{E}, s \in \mathcal{S}\} \quad (4)$$

$$\mathcal{L}_5 = \{\mathbf{x} \in \mathbb{R}^M |\ h_{Y_{\beta(t)}|\text{In}(t)} = 0,\ \forall t \in \mathcal{T}\} \quad (5)$$

we will use $\mathcal{L}(A) = \mathcal{L}_1 \cap \mathcal{L}_3 \cap \mathcal{L}_{4'} \cap \mathcal{L}_5$ to denote the network constraints of network A.

Equation (1) determines the rate region of a network only implicitly. This is due to the fact that Γ_N^* is unknown and even not polyhedral for $N \geq 4$. Thus the direct calculation of rate region for a network with more than 4 variables is infeasible. However when replacing Γ_N^* with some inner bound Γ_N^{in} or outer bound Γ_N^{out}, we will have a feasible polyhedral projection

$$\mathcal{R}_{out}(A) = \text{Proj}_{\mathbf{r},\mathbf{w}}(\Gamma_N^{out} \cap \mathcal{L}_A) \quad (6)$$

$$\mathcal{R}_{in}(A) = \text{Proj}_{\mathbf{r},\mathbf{w}}(\Gamma_N^{in} \cap \mathcal{L}_A) \quad (7)$$

where $\mathcal{R}_{out}(A)$ and $\mathcal{R}_{in}(A)$ are called the *Linear programming* (LP) outer bound and inner bound of rate region $\mathcal{R}(A)$ respectively. Details of theory of these outer and inner bounds can be found in [24]–[26].

III. Computing LP outer bound algebraically

Equations (6) and (7) show why polyhedra projection is involved in the calculation of the rate region of a network. The problem is that either Γ_N^{out} or Γ_N^{in} has the same dimension as Γ_N^*, which makes the computational cost of direct projection very high for even a reasonable value of N, say $N = 10$. Moreover, the worst case upper bound of the computational complexity of general polyhedra projection remains unknown, and were such a bound even to exist, it would be difficult to couple the structure of the network coding problem to it to obtain insights as to the relationship between characteristics of the network and the computation complexity of the LP bound.

Note that by taking the Shannon outer bound Γ_N [25] of Γ_N^* as an example, we will explain in this section how $\mathcal{R}_o = \text{Proj}_{\mathbf{r},\mathbf{w}}(\Gamma_N \cap \mathcal{L}_A)$ can be computed by our method. However, any LP bound with the form in (6) or (7) can be handled by the method we will describe. To make things clear, we will take the network in Fig.1 as an example and provide the corresponding constraint matrix, etc. right after we finish discussing the abstract equations.

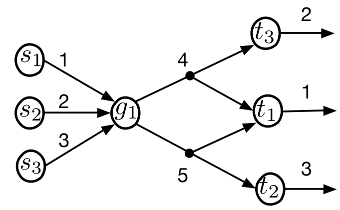

Figure 1: A MSNC network with hyperedges

A. Computing $\Gamma_N \cap \mathcal{L}_A$

We begin by transforming the problem of computing the LP outer bound into one involving new variables. Our first step is to remove the $|\mathcal{V}|$ equality constraints in $\Gamma_N \cap \mathcal{L}_A$ corresponding to the encoding and decoding constraints. Each of these is of the form

$$h_{\text{In}(v)} = h_{\text{In}(v) \cup \text{Out}(v)} \quad (8)$$

for some $v \in \mathcal{V}$. In fact, in the presence of the elemental inequalities in Γ_N, these equalities can implicitly imply multiple other equalities. Indeed, when $|\text{Out}(v)| > 1$, we also have the other implied equalities

$$h_{\text{In}(v)} = h_{\text{In}(v) \cup \mathcal{A}}, \quad \forall \mathcal{A} \subseteq \text{Out}(v), \mathcal{A} \neq \emptyset \quad (9)$$

so that the equality (8) actually implies equality among $2^{|\text{Out}(v)|}$ entropies. Similarly, for the source independence constraint

$$h_{\mathcal{S}} = \sum_{s \in \mathcal{S}} h_s \quad (10)$$

the elemental inequalities imply $2^{|\mathcal{S}|} - 1$ equalities of the form

$$h_{\mathcal{S}'} = \sum_{s \in \mathcal{S}'} h_s, \quad \mathcal{S}' \subseteq \mathcal{S} \quad (11)$$

Stack the constraints (9) and (11) together we will have a system of linear equations that can be represented as,

$$\mathbb{M}\mathbf{h} = \mathbf{0} \quad (12)$$

- *For example, the network constraints of Fig.1 obtained from (9) and (11) are,*

$$\begin{cases} h_{\{1,2,3\}} & = h_{\{1,2,3,4,5\}} \\ h_{\{4\}} & = h_{\{2,4\}} \\ h_{\{4,5\}} & = h_{\{1,4,5\}} \\ h_{\{5\}} & = h_{\{3,5\}} \\ h_{\{1,2,3\}} & = h_{\{1\}} + h_{\{2\}} + h_{\{3\}} \\ h_{\{1,2\}} & = h_{\{1\}} + h_{\{2\}} \\ h_{\{1,3\}} & = h_{\{1\}} + h_{\{3\}} \\ h_{\{2,3\}} & = h_{\{2\}} + h_{\{3\}} \\ h_{\{1,2,3\}} & = h_{\{1,2,3,4\}} \\ h_{\{1,2,3\}} & = h_{\{1,2,4,5\}} \end{cases} \quad (13)$$

The corresponding matrix $\mathbb{M}$ has 10 rows, 36 columns and rank$(\mathbb{M}) = 10$.

By calculating the reduced row echelon form rref($\mathbb{M}$), we can divide $\mathbf{h} = \mathbf{h}_1 \cup \mathbf{h}_r$ with $|\mathbf{h}_1| = \text{rank}(\mathbb{M})$, $|\mathbf{h}_r| = 2^N - 1 - \text{rank}(\mathbb{M})$ such that each subset entropy in $\mathbf{h}_1$ can be written as a linear combination of singleton and/or subset entropies in $\mathbf{h}_r$ as,

$$\mathbf{h}_1 = \mathbb{M}'\mathbf{h}_r \tag{14}$$

- *For example, based on rref($\mathbb{M}$), the entropy vector $\mathbf{h}$ of Fig.1 can be divided as $\mathbf{h} = \mathbf{h}_1 \cup \mathbf{h}_r$ and the corresponding $\mathbb{M}'$ has 10 rows 26 columns.*

After substituting (14) into the remaining elemental inequalities and rate constraints, $\Gamma_N \cap \mathcal{L}_A$ will become a polyhedral cone written in terms of $2N + \binom{N}{2}2^{N-2}$ inequalities (from the elemental inequalities defining Γ_N and the source and edge rate inequalities) and $N + 2^N - 1 - \text{rank}(\mathbb{M})$ variables, possibly with redundant inequalities and implicit equalities.

- *For example, after substituting in* (14)*, $\Gamma_N \cap \mathcal{L}_A$ of network in Fig.1 will become a polyhedral cone that has $2 \times 5 + \binom{5}{2}2^3 = 95$ inequalities and $5 + 2^5 - 1 - 10 = 26$ variables.*

B. Computing the Projection Algebraically

Let us denote by $\mathbb{A}$ the matrix defining the minimal representation of $\Gamma_N \cap \mathcal{L}_A$ as,

$$\mathcal{P}_A = \left\{ [\boldsymbol{\omega}^T, \boldsymbol{r}^T, \boldsymbol{h}_r^T]^T \in \mathbb{R}^d | \mathbb{A}[\boldsymbol{\omega}^T, \boldsymbol{r}^T, \boldsymbol{h}_r^T]^T \geq \mathbf{0} \right\} \tag{15}$$

where implicit equalities and redundant inequalities of this cone are eliminated by 1) eliminating the variables after substituting in (14), then 2) removing any further implicit equalities by eliminating other entropy variables, then 3) removing any redundant remaining inequalities.

- *For example, after removing all the redundant inequalities and variables in $\Gamma_N \cap \mathcal{L}_A$ the minimal representation $\mathcal{P}_A$ of network in Fig.1 has 30 out of 95 inequalities and 26 out of 26 variables left.*

From (6) and (15) one can see that $\mathcal{R}_o$ can be calculated as,

$$\mathcal{R}_o = \text{Proj}_{\mathbf{r},\mathbf{w}}(\mathcal{P}_A) \tag{16}$$

The remainder of the projection process is to remove of the remaining $2^N - 1 - \text{rank}(\mathbb{M})$ dimensions representing subset entropies, leaving only inequalities among the N variables $\boldsymbol{\omega}, \boldsymbol{r}$. This $m \times d$ matrix $\mathbb{A}$ has $m \leq 2N + \binom{N}{2}2^{N-2}$ rows and $d \leq N + 2^N - 1 - \text{rank}(\mathbb{M})$ columns.

Now, any inequality $\boldsymbol{c}^T[\boldsymbol{\omega}^T, \boldsymbol{r}^T]^T \geq 0$ which holds for all $[\boldsymbol{\omega}^T, \boldsymbol{r}^T]^T \in \mathcal{R}_o = \text{Proj}_{\mathbf{r},\mathbf{w}}(\mathcal{P}_A)$ must be expressible as a conic combination of inequalities defining $\mathcal{P}_A$, and thus for any such $\boldsymbol{c}$, there must exist some $\boldsymbol{\mu} \geq \mathbf{0}$ with,

$$[\boldsymbol{c}^T, \mathbf{0}^T][\boldsymbol{\omega}^T, \boldsymbol{r}^T, \mathbf{0}^T]^T = \boldsymbol{\mu}^T \mathbb{A}[\boldsymbol{\omega}^T, \boldsymbol{r}^T, \boldsymbol{h}_r^T]^T \tag{17}$$

Our next transformation, again aiming to maximize those steps which have a linear algebraic flavor in computing the polyhedral projection for the sake of easily upper bounding the complexity of computing the LP outer bound, is to use this duality to switch to thinking of $\boldsymbol{\mu}$ as the variables instead of $[\boldsymbol{\omega}^T, \boldsymbol{r}^T, \boldsymbol{h}_r^T]^T$. From (17) one can see that any such $\boldsymbol{\mu}$ must yield zero coefficients to multiply the subset entropies in $\boldsymbol{h}_r$. If we denote by $\mathscr{M}$ the indices of columns in $\mathbb{A}$ that correspond to $\mathbf{h}_r$, then these "zero-coefficient" equalities are of the form

$$\boldsymbol{\mu}^T \mathbb{A}_{:,\mathcal{A}} = \mathbf{0}^T \quad \mathcal{A} \in \mathscr{M} \tag{18}$$

which is equivalent as,

$$\mathbb{G}\boldsymbol{\mu} = \mathbf{0} \tag{19}$$

where,

$$\mathbb{G} = [\mathbb{A}_{:,\mathcal{A}}^T | \mathcal{A} \in \mathscr{M}] \tag{20}$$

After calculating the rref($\mathbb{G}$), $\boldsymbol{\mu}$ can be divided into two disjoint subsets $\boldsymbol{\mu}_f$ corresponding to the leading 1s in the rows of rref($\mathbb{G}$), and $\boldsymbol{\mu}_b$ corresponding to the remaining elements of $\boldsymbol{\mu}$, with $|\boldsymbol{\mu}_f| = |\text{rank}(\mathbb{G})|$, $|\boldsymbol{\mu}_b| = m - |\text{rank}(\mathbb{G}^T)|$ such that each $\mu \in \boldsymbol{\mu}_f$ can be written as a linear combination of the variables in $\boldsymbol{\mu}_b$ as,

$$\boldsymbol{\mu}_f = \mathbb{B}\boldsymbol{\mu}_b, \tag{21}$$

together with the requirement that $\boldsymbol{\mu} \geq 0$, we will have the following polyhedral cone,

$$P(\boldsymbol{\mu}_b) = \left\{ \boldsymbol{\mu}_b \middle| \begin{bmatrix} \mathbb{I} \\ \mathbb{B} \end{bmatrix} \in \mathbb{R}^{m \times d'} \begin{bmatrix} \mathbb{I} \\ \mathbb{B} \end{bmatrix} \boldsymbol{\mu}_b \geq 0 \right\} \tag{22}$$

where $\mathbb{I}$ denotes a $d' \times d'$ identity matrix, $d' = m - |\text{rank}(\mathbb{G})|$. Notice that each $\boldsymbol{\mu}_b \in P(\boldsymbol{\mu}_b)$ corresponds to a $\boldsymbol{\mu}$ in (17) and thus corresponds to a inequality $\boldsymbol{c}^T[\boldsymbol{\omega}^T, \boldsymbol{r}^T]^T \geq 0$. So once we find all the extreme rays of $P(\boldsymbol{\mu}_b)$, we will find all the conically independent inequalities of the form $\boldsymbol{c}^T[\boldsymbol{\omega}^T, \boldsymbol{r}^T]^T \geq 0$ that defines $\mathcal{R}_o$.

- *For example for the network in Fig.1, the vector $\boldsymbol{\mu}$ is split into $[\boldsymbol{\mu}_f, \boldsymbol{\mu}_b]$ with $|\boldsymbol{\mu}_f| = 21$, $|\boldsymbol{\mu}_b| = 14$ and the corresponding $\mathbb{B}$ matrix is shown in Fig.2. Consequently, cone $\mathcal{P}(\mu_b)$ has 35 rows and 14 columns.*

$$\begin{bmatrix}
1, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & -1, & 0, & 0, & 0, & 1, & 0 \\
0, & 0, & 1, & 1, & 1, & 0, & 0, & 0, & 0, & 0, & -1, & 0, & 0, & 1 \\
1, & 1, & 0, & 0, & -1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\
0, & 0, & 0, & 0, & 1, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\
0, & 1, & 0, & 0, & -1, & 0, & -1, & 0, & -1, & 0, & 0, & 0, & 1, & 1 \\
0, & 1, & 0, & 1, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\
0, & -1, & 0, & 0, & 1, & -1, & 0, & 0, & 0, & 0, & -1, & 0, & 1, & 1 \\
0, & -1, & 1, & 0, & 1, & -1, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\
0, & -1, & 0, & 0, & 1, & -1, & 0, & 0, & 0, & 0, & 0, & 0, & 1, & 0 \\
0, & 1, & 0, & 0, & -1, & 1, & -1, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\
0, & 1, & 0, & 0, & -1, & 0, & -1, & 0, & 0, & 0, & 0, & 0, & 0, & 1 \\
0, & -1, & 0, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\
1, & 0, & 0, & 0, & 0, & -1, & 0, & 0, & 0, & 0, & 0, & 0, & 1, & 0 \\
0, & 0, & 0, & 0, & 1, & 0, & 0, & 0, & -1, & 0, & -1, & 0, & 1, & 1 \\
0, & 0, & 1, & 0, & 0, & -1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 1 \\
0, & 1, & 0, & 0, & 0, & 1, & 0, & 0, & 0, & 0, & 1, & 0, & -1, & -1 \\
0, & 0, & 0, & 1, & 0, & 1, & 0, & 0, & 1, & 0, & 0, & 0, & -1, & -1 \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -1, & 0, & 0, & 0, & 1, & 0 \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -1, & 0, & 0, & 0, & 1, & 1 \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -1, & 0, & 1, & 1 \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & -1, & 0, & 0, & 1
\end{bmatrix}$$

Figure 2: $\mathbb{B}$ matrix for network in Fig.1

IV. Complexity of computing $\mathcal{R}_o$ from $P(\boldsymbol{\mu}_b)$

The problem of finding all extreme rays of $P(\boldsymbol{\mu_b})$ left in the previous subsection can be solved by a classical incremental algorithm called *double description* (DD) method [27]. By utilizing the structure of $[\mathbb{I}^T, \mathbb{B}^T]^T$ and introducing an intermediate matrix $Indi_K$, we derive the worst case computational complexity bound of computing $\mathcal{R}_o$ from $P(\boldsymbol{\mu}_b)$.

Again for brevity, we borrow the notation and definitions from [27], where $(\mathbf{E}, \mathbf{R})$ denotes the DD pair of a polyhedral cone $P(\mathbf{E})$. Moreover Proposition 7(c) and Lemma 8 [27] are used to write the pseudocode of DD method for complexity analysis.

The existence of an identity matrix in $[\mathbb{I}^T, \mathbb{B}^T]^T \in \mathbb{R}^{m\times d'}$ is one of the reason why lower complexity bound can be achieved than if a normal matrix $\mathbf{E} \in \mathbb{R}^{m\times d'}$ is input into the DD method. This is because 1) DD method involves computation of a vector multiply by the input matrix in several for loops, and 2) identity matrix multiply a vector can be done in constant time.

The computational complexity of DD method can be further decreased if we trade in space complexity for computational complexity. To do that, we introduce an intermediate matrix $Inid_K$ in the DD method iteration step as,

$$Indi_K = \{\mathbf{v}_j = \mathbf{E}\mathbf{r}_j | \forall \mathbf{r}_j \in \mathbf{R}_K\} \tag{23}$$

where $\mathbf{v}_j, \mathbf{r}_j$ denotes $j-th$ column in $Indi_K$ and $\mathbf{R}_K$ respectively. Noet that for a column $\mathbf{r}_j \in \mathbf{R}_K$, $Z(\mathbf{r}_j)$ is defined as,

$$Z(\mathbf{r}_j) = \{i \in \{1,...,m\} | \mathbf{E}_i\mathbf{r} = 0\} \tag{24}$$

One can see that $Z(\mathbf{r}_j)$ is equivalent as finding the indices of zero element of corresponding $\mathbf{v}_j \in Indi_K$.

The DD method is known to be *output sensitive*, which means that the algorithms' performance is guaranteed in terms of the output size as well as the input size [28]. If we use the same assumption as [27] that the number of rays generated in the intermediate step remains in $\mathcal{O}(n)$, then for the input matrix $\mathbf{E} = [\mathbb{I}, \mathbf{B}^T]^T \in \mathbb{R}^{m\times d'}$, the DD method iteration step together with the complexity of each line of pseudocode can be written as Algorithm1.

If we assume that the input size m is less than the output size n, then the complexity of DD method iteration step is

$$\begin{aligned}&\mathcal{O}(n) + \mathcal{O}(n^2)\mathcal{O}(n)(3\mathcal{O}(d')) + \mathcal{O}(n)\mathcal{O}(md' - d'^2)\\ =&\mathcal{O}(n) + \mathcal{O}(n^3 d') + \mathcal{O}(nmd' - nd'^2)\\ =&\mathcal{O}(n^3 d')\end{aligned} \tag{25}$$

Given that we have $[\mathbb{I}, \mathbf{B}^T]^T$ as the input, we can always choose $(\mathbb{I}, \mathbb{I})$ as the initial DD pair. This is to say we need to insert all the rows of $\mathbf{B}$ step by step, which gives us $m - d'$ calls of DD method iteration steps. So the overall complexity of DD method when $[\mathbb{I}, \mathbf{B}^T]^T$ is the input is,

$$(m - d')\mathcal{O}(n^3 d') = \mathcal{O}(mn^3 d' - n^3 d'^2) \tag{26}$$

For each extreme ray $\boldsymbol{\mu}_b$ in $P(\boldsymbol{\mu}_b)$ to get the corresponding inequality $\boldsymbol{c}^T[\boldsymbol{\omega}^T, \boldsymbol{r}^T]^T \geq 0$, we need to first calculate $\boldsymbol{\mu}_f = \mathbb{B}\boldsymbol{\mu}_b$ and then tacking them together to get $\boldsymbol{\mu} = [\boldsymbol{\mu}_f^T, \boldsymbol{\mu}_b^T]^T$ and finally calculate $\boldsymbol{c}^T = \boldsymbol{\mu}^T\mathbb{A}$. As we had assumed n is the number of extreme rays, the computational complexity of these steps that involve two different matrix vector multiplication can be bounded by

$$\mathcal{O}(n(m - d')d') + \mathcal{O}(nmd) = \mathcal{O}(ndm) \tag{27}$$

So the overall complexity of computing $\mathcal{R}_o$ from $P(\boldsymbol{\mu}_b)$ is upper bounded by,

$$\mathcal{O}(ndm) + \mathcal{O}(mn^3 d' - n^3 d'^2) \tag{28}$$

In summary, we have proven the following theorem.

Theorem 1. *Assuming as in [27] that the number of rays generated in at each intermediate step of DD remains in $\mathcal{O}(n)$, the complexity of computing the LP outer bound $\mathcal{R}_o$ to a MSNC-DH problem is upper bounded by*

$$\mathcal{O}(ndm) + \mathcal{O}(mn^3 d' - n^3 d'^2) \tag{29}$$

where $m \leq N + 2^N - 1 - rank(\mathbb{M})$, $d' \leq N + 2^N - 1 - rank(\mathbb{M}) - rank(\mathbb{G})$, $\mathbb{M}$ and $\mathbb{G}$ are matrices built from the network constraints according to equations (9,11) and (20) respectively, and n is the number of inequalities defining the resulting LP outer bound.

Algorithm 1 DD method iteration step

Input: $\mathbf{E}, (\mathbf{E}_K, \mathbf{R}_K), Indi_K, J, i$
Output: $(\mathbf{E}_{K+1}, \mathbf{R}_{K+1}), Indi_{K+1}$
Initialize: $Adj = \emptyset$, $J^+ = \emptyset$, $J^- = \emptyset$, $J^0 = \emptyset$, $\mathbf{R}_{K+i} = \emptyset$, $Indi_{K+1} = \emptyset$

procedure DD UPDATES
  for $j = 1, ..., |J|$ **do** ▷ $\mathcal{O}(n)$
    if $Indi(i,j) > 0$ **then** ▷ $\mathcal{O}(1)$
      J^+.Append(j) ▷ $\mathcal{O}(1)$
      $\mathbf{R}_{K+1}$.Append$(\mathbf{r}_j)$ ▷ $\mathcal{O}(1)$
      $Indi_{K+1}$.Append$(\mathbf{v}_j)$ ▷ $\mathcal{O}(1)$
    else if $Indi(i,j) == 0$ **then** ▷ $\mathcal{O}(1)$
      J^0.Append(j) ▷ $\mathcal{O}(1)$
      $\mathbf{R}_{K+1}$.Append$(\mathbf{r}_j)$ ▷ $\mathcal{O}(1)$
      $Indi_{K+1}$.Append$(\mathbf{v}_j)$ ▷ $\mathcal{O}(1)$
    else ▷ $\mathcal{O}(1)$
      J^-.Append(j) ▷ $\mathcal{O}(1)$
  for each $(j, j') \in (J^+, J^-)$ **do** ▷ $\mathcal{O}(n^2)$
    for each $j'' \in J$ **do** ▷ $\mathcal{O}(n)$
      if $Z(\mathbf{v}_j) \cap Z(\mathbf{v}_{j'}) \subset Z(\mathbf{v}_{j''})$ **then** ▷ $\mathcal{O}(d')$
        if $\mathbf{r}_{j''} \simeq \mathbf{r}_j$ or $\mathbf{r}_{j''} \simeq \mathbf{r}_{j'}$ **then** ▷ $\mathcal{O}(d')$
          Adj.Append$((j, j'))$ ▷ $\mathcal{O}(1)$
          $\mathbf{r}_{jj'} \leftarrow (\mathbf{E}_i\mathbf{r}_j)\mathbf{r}_{j'} - (\mathbf{E}_i\mathbf{r}_{j'})\mathbf{r}_j$ ▷ $\mathcal{O}(d')$
          $\mathbf{R}_{K+1}$.Append$(\mathbf{r}_{jj'})$ ▷ $\mathcal{O}(1)$
  for each $\mathbf{r}_{jj'} \in \mathbf{R}_{K+1} \setminus \mathbf{R}_K$ **do** ▷ $\mathcal{O}(n)$
    $\mathbf{v}_{jj'} \leftarrow \mathbf{E}\mathbf{r}_{jj'}$ ▷ $\mathcal{O}(md' - d'^2)$
    $Indi_{K+1}$.Append$(\mathbf{v}_{jj'})$ ▷ $\mathcal{O}(1)$

■ *For example for the network in Fig.1, when using the DD method to compute the extreme rays of $P(\boldsymbol{\mu}_b)$ 22 extreme* *rays are found. Those extreme rays are listed in Fig.3 where each row of the matrix denotes an extreme ray of $P(\boldsymbol{\mu_b})$. These extreme rays are then used to calculate, through our aforementioned steps, the inequalities that defines $\mathcal{R}_o$, which are,*

$$\begin{cases} w_1 & \geq 0 \\ w_2 & \geq 0 \\ w_3 & \geq 0 \\ R_5 & \geq w_3 \\ R_4 & \geq w_2 \\ R_4 + R_5 & \geq w_1 + w_2 + w_3 \end{cases} \tag{30}$$

$$\begin{bmatrix}
0, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\
0, & 0, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & 0 \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 1, & 0, & 0, & 0, & 0 \\
0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 1, & 0, & 0 \\
0, & 0, & 1, & 0, & 0, & 1, & 0, & 0, & 1, & 0, & 1, & 0, & 1, & 1 \\
0, & 1, & 0, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0 \\
1, & 0, & 0, & 0, & 1, & 1, & 0, & 0, & 1, & 0, & 1, & 0, & 1, & 1 \\
0, & 1, & 0, & 1, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 1, & 0 \\
1, & 0, & 0, & 1, & 1, & 1, & 0, & 0, & 0, & 0, & 1, & 0, & 1, & 1 \\
0, & 1, & 0, & 0, & 1, & 0, & 0, & 0, & 1, & 0, & 0, & 0, & 1, & 0 \\
0, & 1, & 0, & 1, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 1 \\
1, & 0, & 0, & 0, & 1, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 0, & 1 \\
0, & 1, & 0, & 1, & 1, & 1, & 1, & 0, & 0, & 0, & 0, & 0, & 1, & 1 \\
0, & 1, & 0, & 0, & 1, & 1, & 1, & 0, & 1, & 0, & 0, & 0, & 1, & 1 \\
0, & 0, & 1, & 0, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 1, & 0 \\
0, & 0, & 0, & 0, & 0, & 1, & 1, & 0, & 1, & 0, & 1, & 0, & 1, & 1 \\
0, & 0, & 0, & 1, & 0, & 0, & 0, & 0, & 0, & 0, & 1, & 0, & 0, & 1 \\
1, & 0, & 0, & 0, & 1, & 1, & 0, & 0, & 0, & 0, & 1, & 0, & 0, & 1 \\
0, & 0, & 0, & 1, & 0, & 1, & 1, & 0, & 0, & 0, & 1, & 0, & 1, & 1 \\
0, & 0, & 1, & 0, & 0, & 1, & 0, & 0, & 1, & 0, & 0, & 0, & 1, & 0 \\
0, & 1, & 1, & 0, & 1, & 1, & 0, & 0, & 1, & 0, & 0, & 0, & 1, & 1
\end{bmatrix}$$

Figure 3: Extreme rays found for the network in Fig.1

V. Conclusion

A linear algebraic method for computing the LP outer bound to a network coding capacity region amenable to complexity analysis was presented. Using this method, the complexity of computing the LP outer bound was upper bounded. Future work will compare this upper bound to the complexity of methods which exploit symmetry and hierarchy in the network in order to quantify the reductions in complexity enabled by these aspects.

References

[1] C. Tian, "Rate region of the (4, 3, 3) exact-repair regenerating codes," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 1426–1430, July 2013.

[2] C. Tian, "Characterizing the rate region of the (4,3,3) exact-repair regenerating codes," *IEEE Journal on Selected Areas in Communications*, vol. 32, pp. 967–975, May 2014.

[3] A. Dimakis, P. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Transactions on Information Theory*, vol. 56, pp. 4539–4551, Sept 2010.

[4] M. Sathiamoorthy, M. Asteris, D. S. Papailiopoulos, A. Dimakis, R. Vadali, S. Chen, and D. Borthakur, "Xoring elephants: Novel erasure codes for big data," *Proceedings of the VLDB Endowment*, vol. 6, pp. 325–336, Mar 2013.

[5] R. Koetter, M. Effros, and M. Medard, "A theory of network equivalence part ii: Multiterminal channels," *Information Theory, IEEE Transactions on*, vol. 60, pp. 3709–3732, July 2014.

[6] M. Effros, S. El Rouayheb, and M. Langberg, "An equivalence between network coding and index coding," *Information Theory, IEEE Transactions on*, vol. 61, pp. 2478–2487, May 2015.

[7] H. Sun and S. A. Jafar, "Index coding capacity: How far can one go with only shannon inequalities?," *CoRR*, vol. abs/1303.7000, 2013.

[8] Chao Tian, "Symmetry, Outer Bounds, and Code Constructions: A Computer-Aided Investigation on the Fundamental Limits of Caching." https://arxiv.org/abs/1611.00024.

[9] Steven Weber, Congduan Li, John MacLaren Walsh, "Rate Region for A Class of Delay Mitigating Codes and P2P Networks," in *46th Annual Conference on Information Sciences and Systems*, Mar. 2012. (Invited Paper).

[10] J. M. Walsh, S. Weber, and C. wa Maina, "Optimal Rate Delay Tradeoffs and Delay Mitigating Codes for Multipath Routed and Network Coded Networks," vol. 55, pp. 5491–5510, Dec. 2009.

[11] J. M. Walsh, S. P. Weber, J. C. de Oliveira, A. Eryilmaz, M. Médard, "Trading Rate for Delay at the Application and Transport Layers (Guest Editorial)," vol. 29, pp. 913–915, May 2011.

[12] X. Yan, R. W. Yeung, and Z. Zhang, "An implicit characterization of the achievable rate region for acyclic multisource multisink network coding," *IEEE Transactions on Information Theory*, vol. 58, pp. 5625–5639, Sept 2012.

[13] Jayant Apte and John MacLaren Walsh, "Explicit Polyhedral Bounds on Network Coding Rate Regions via Entropy Function Region: Algorithms, Symmetry, and Computation," 2016. Submitted July 22, 2016, revised July 6, 2017.

[14] Jayant Apte and John MacLaren Walsh, "Constrained Linear Representability of Polymatroids and Algorithms for Computing Achievability Proofs in Network Coding," Submitted May 15, 2016.

[15] Jayant Apte, *Algorithms for Computing Limits on Information Flow and Storage in Networks*. PhD thesis, Drexel University, Philadelphia, PA, 2016.

[16] Jayant Apte and John MacLaren Walsh, "Information Theoretic Converse Prover."

[17] Jayant Apte and John MacLaren Walsh, "Information Theoretic Achievability Prover."

[18] Congduan Li and John MacLaren Walsh, "Network Enumeration and Hierarchy."

[19] C. Li, S. Weber, and J. M. Walsh, "On multi-source networks: Enumeration, rate region computation, and hierarchy," *IEEE Transactions on Information Theory*, vol. 63, pp. 7283–7303, Nov 2017.

[20] J. Apte and J. M. Walsh, "Exploiting symmetry in computing polyhedral bounds on network coding rate regions," in *IEEE International Symposium on Network Coding (NetCod)*, Jun 2015.

[21] Congduan Li, Steven Weber, John MacLaren Walsh, "Network Embedding Operations Preserving the Insufficiency of Linear Network Codes," in *52nd Annual Allerton Conference on Communication, Control, and Computing*, Oct. 2014.

[22] C. Li and J. M. Walsh, "Network combination operations preserving the sufficiency of linear network codes," in *2015 IEEE Information Theory Workshop (ITW2015)*, (Jeju Island, Korea), Oct. 2015.

[23] C. Li, S. Weber, and J. M. Walsh, "Network combination operations preserving the sufficiency of linear network codes," in *2015 IEEE Information Theory Workshop - Fall (ITW)*, pp. 44–48, Oct 2015.

[24] J. Apte, C. Li, and J. MacLaren Walsh, "Algorithms for computing network coding rate regions via single element extensions of matroids," pp. 2306–2310, 06 2014.

[25] C. Li, J. Apte, J. M. Walsh, and S. Weber, "A new computational approach for determining rate regions and optimal codes for coded networks," in *2013 International Symposium on Network Coding (NetCod)*, pp. 1–6, June 2013.

[26] C. Li, S. Weber, and J. M. Walsh, "Multilevel diversity coding systems: Rate regions, codes, computation, amp; forbidden minors," *IEEE Transactions on Information Theory*, vol. 63, pp. 230–251, Jan 2017.

[27] K. Fukuda and A. Prodon, "Double description method revisited," in *Combinatorics and Computer Science* (M. Deza, R. Euler, and I. Manoussakis, eds.), (Berlin, Heidelberg), pp. 91–111, Springer Berlin Heidelberg, 1996.

[28] D. Bremner, "Incremental convex hull algorithms are not output sensitive," *Discrete & Computational Geometry*, vol. 21, pp. 57–68, Jan 1999.