Multiterminal Networks: Rate Regions, Codes, Computations, & Forbidden Minors

Congduan Li

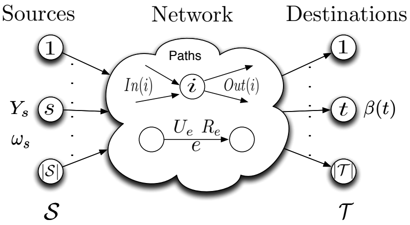
ASPITRG (Dr. John Walsh) & MANL (Dr. Steven Weber)
Drexel University

congduan.li@gmail.com

Nov 25, 2014
INC, CUHK

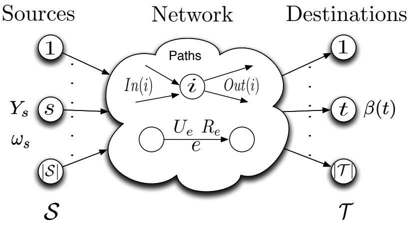
Problems of interest

A multisource multicast network:



- Rate/capacity region: compute
- If Shannon outer (LP) bound tight
- If simple linear codes sufficient

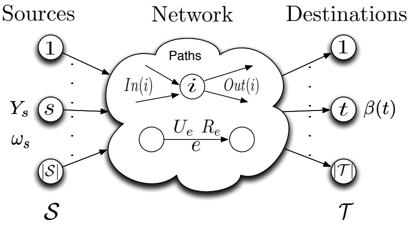
A multisource multicast network:



- Rate/capacity region: compute
- If Shannon outer (LP) bound tight
- If simple linear codes sufficient

Problems of interest

A multisource multicast network:



- Rate/capacity region: compute
- If Shannon outer (LP) bound tight
- If simple linear codes sufficient

Outline

1 Rate Region

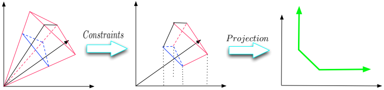
2 Shannon Outer Bounds & Converse

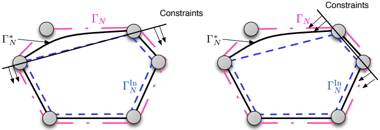
3 Matroid Bounds and Achievability

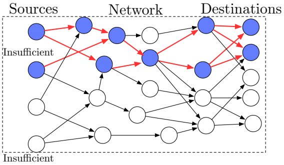
4 Forbidden Embedding Networks

- Motivation
- Operations
- Results

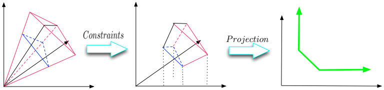
5 Future work



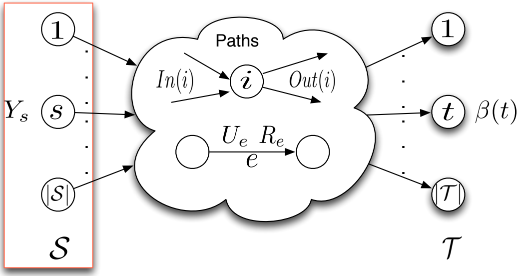




Outline

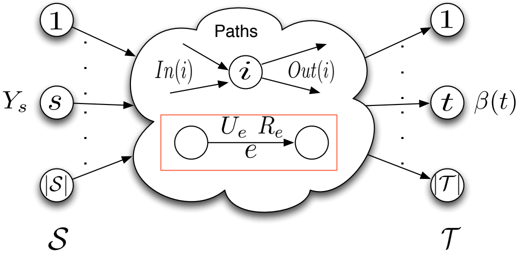


Network Coding: Sources



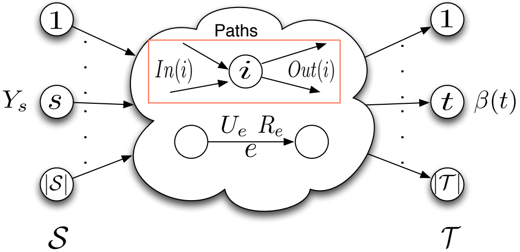
Source Rate	$H(Y_s) \geq \omega_s$
Source Independence	$H(Y_{\mathcal{S}}) = \sum_{s \in \mathcal{S}} H(Y_s)$
Source Encoding	$H(U_{\text{Out}(s)} \| Y_s) = 0$

Network Coding: Edges



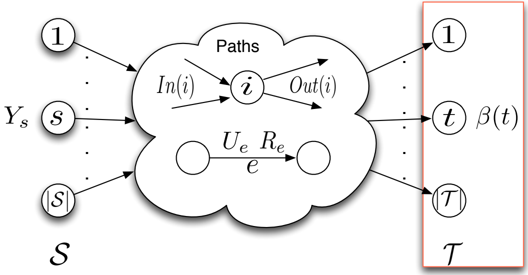
Coding Rate $R_e \geq H(U_e), \; e \in \mathcal{E}$

Network Coding: Nodes



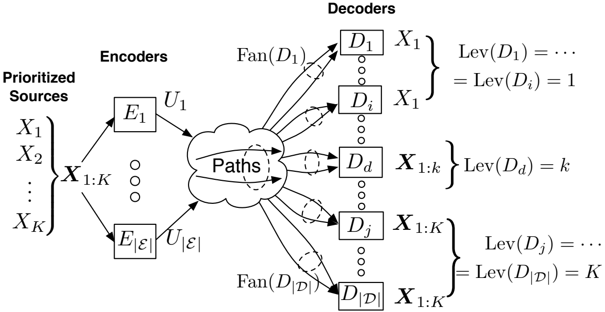
Coding Constraints $H(U_{\mathrm{Out}(i)}|U_{\mathrm{In}(i)}) = 0,\ i \in \mathcal{V} \backslash (\mathcal{S} \cup \mathcal{T})$

Network Coding: Sinks



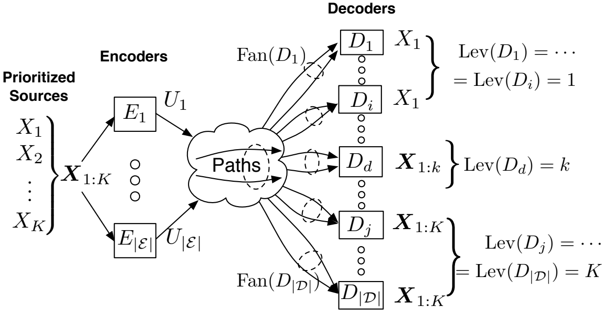
Decoding Constraints $H(Y_{\beta(t)}|U_{\text{In}(t)}) = 0, \ t \in \mathcal{T}$

Example: Multilevel Diversity Coding Systems (MDCS)



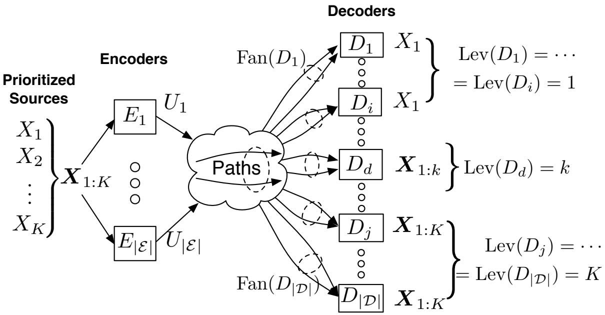
- Prioritized sources, no intermediate nodes
- Decoders are classified to levels: level k, decode $\mathbf{X}_{1:k}$
- Notation: $(k, |\mathcal{E}|)$ MDCS instances

Example: Multilevel Diversity Coding Systems (MDCS)



- Prioritized sources, no intermediate nodes
- Decoders are classified to levels: level k, decode $\mathbf{X}_{1:k}$
- Notation: $(k, |\mathcal{E}|)$ MDCS instances

Example: Multilevel Diversity Coding Systems (MDCS)



- Prioritized sources, no intermediate nodes
- Decoders are classified to levels: level k, decode $\mathbf{X}_{1:k}$
- Notation: $(k, |\mathcal{E}|)$ MDCS instances

Rate region

- Rate region: all possible rate and source entropy vectors satisfying all network constraints.
- Collect the N network random variables and their joint entropies.
- Define Γ_N^*: in $2^N - 1$-dim. space, region of valid entropy vectors.
- Constraints from network A:

$$\mathcal{L}_1 = \{\mathbf{h} \in \Gamma_N^* : h_{Y_\mathcal{S}} = \Sigma_{s \in \mathcal{S}} h_{Y_s}\} \quad (1)$$

$$\mathcal{L}_2 = \{\mathbf{h} \in \Gamma_N^* : h_{X_{\mathrm{Out}(k)}|Y_s} = 0\} \quad (2)$$

$$\mathcal{L}_3 = \{\mathbf{h} \in \Gamma_N^* : h_{X_{\mathrm{Out}(i)}|X_{\mathrm{In}(i)}} = 0\} \quad (3)$$

$$\mathcal{L}_4 = \{(\mathbf{h}^T, \mathbf{R}^T)^T \in \mathbb{R}_+^{2^N - 1 + |\mathcal{E}|} : R_e \geq h_{U_e}, e \in \mathcal{E}\} \quad (4)$$

$$\mathcal{L}_5 = \{\mathbf{h} \in \Gamma_N^* : h_{Y_{\beta(t)}|U_{\mathrm{In}(t)}} = 0\}. \quad (5)$$

- Rate region (cone) in terms of rates and source entropies (derived from [Yan, Yeung, Zhang TranIT 2012]):

$$\mathcal{R}(\mathrm{A}) = \mathrm{proj}_{R_\mathcal{E}, H(Y_\mathcal{S})}(\overline{\mathrm{con}(\Gamma_\mathrm{N}^* \cap \mathcal{L}_{123})} \cap \mathcal{L}_{45}) \quad (6)$$

Rate region

- Rate region: all possible rate and source entropy vectors satisfying all network constraints.
- Collect the N network random variables and their joint entropies.
- Define Γ_N^*: in $2^N - 1$-dim. space, region of valid entropy vectors.
- Constraints from network A:

$$\mathcal{L}_1 = \{\mathbf{h} \in \Gamma_N^* : h_{Y_\mathcal{S}} = \Sigma_{s\in\mathcal{S}} h_{Y_s}\} \quad (1)$$

$$\mathcal{L}_2 = \{\mathbf{h} \in \Gamma_N^* : h_{X_{\mathrm{Out}(k)}|Y_s} = 0\} \quad (2)$$

$$\mathcal{L}_3 = \{\mathbf{h} \in \Gamma_N^* : h_{X_{\mathrm{Out}(i)}|X_{\mathrm{In}(i)}} = 0\} \quad (3)$$

$$\mathcal{L}_4 = \{(\mathbf{h}^T, \mathbf{R}^T)^T \in \mathbb{R}_+^{2^N-1+|\mathcal{E}|} : R_e \geq h_{U_e}, e \in \mathcal{E}\} \quad (4)$$

$$\mathcal{L}_5 = \{\mathbf{h} \in \Gamma_N^* : h_{Y_{\beta(t)}|U_{\mathrm{In}(t)}} = 0\}. \quad (5)$$

- Rate region (cone) in terms of rates and source entropies (derived from [Yan, Yeung, Zhang TranIT 2012]):

$$\mathcal{R}(\mathrm{A}) = \mathrm{proj}_{R_\mathcal{E}, H(Y_\mathcal{S})}(\overline{\mathrm{con}(\Gamma_\mathrm{N}^* \cap \mathcal{L}_{123})} \cap \mathcal{L}_{45}) \quad (6)$$

Rate region

- Rate region: all possible rate and source entropy vectors satisfying all network constraints.
- Collect the N network random variables and their joint entropies.
- Define Γ_N^*: in $2^N - 1$-dim. space, region of valid entropy vectors.
- Constraints from network A:

$$\mathcal{L}_1 = \{\mathbf{h} \in \Gamma_N^* : h_{Y_\mathcal{S}} = \Sigma_{s \in \mathcal{S}} h_{Y_s}\} \quad (1)$$

$$\mathcal{L}_2 = \{\mathbf{h} \in \Gamma_N^* : h_{X_{\mathrm{Out}(k)}|Y_s} = 0\} \quad (2)$$

$$\mathcal{L}_3 = \{\mathbf{h} \in \Gamma_N^* : h_{X_{\mathrm{Out}(i)}|X_{\mathrm{In}(i)}} = 0\} \quad (3)$$

$$\mathcal{L}_4 = \{(\mathbf{h}^T, \mathbf{R}^T)^T \in \mathbb{R}_+^{2^N-1+|\mathcal{E}|} : R_e \geq h_{U_e}, e \in \mathcal{E}\} \quad (4)$$

$$\mathcal{L}_5 = \{\mathbf{h} \in \Gamma_N^* : h_{Y_{\beta(t)}|U_{\mathrm{In}(t)}} = 0\}. \quad (5)$$

- Rate region (cone) in terms of rates and source entropies (derived from [Yan, Yeung, Zhang TranIT 2012]):

$$\mathcal{R}(\mathrm{A}) = \mathrm{proj}_{R_\mathcal{E}, H(Y_\mathcal{S})}(\overline{\mathrm{con}(\Gamma_\mathrm{N}^* \cap \mathcal{L}_{123})} \cap \mathcal{L}_{45}) \quad (6)$$

Rate region

- Rate region: all possible rate and source entropy vectors satisfying all network constraints.
- Collect the N network random variables and their joint entropies.
- Define Γ_N^*: in $2^N - 1$-dim. space, region of valid entropy vectors.
- Constraints from network A:

$$\begin{aligned}\mathcal{L}_1 &= \{\mathbf{h} \in \Gamma_N^* : h_{Y_\mathcal{S}} = \Sigma_{s\in\mathcal{S}} h_{Y_s}\} & (1)\\ \mathcal{L}_2 &= \{\mathbf{h} \in \Gamma_N^* : h_{X_{\mathrm{Out}(k)}|Y_s} = 0\} & (2)\\ \mathcal{L}_3 &= \{\mathbf{h} \in \Gamma_N^* : h_{X_{\mathrm{Out}(i)}|X_{\mathrm{In}(i)}} = 0\} & (3)\\ \mathcal{L}_4 &= \{(\mathbf{h}^T, \mathbf{R}^T)^T \in \mathbb{R}_+^{2^N-1+|\mathcal{E}|} : R_e \geq h_{U_e}, e \in \mathcal{E}\} & (4)\\ \mathcal{L}_5 &= \{\mathbf{h} \in \Gamma_N^* : h_{Y_{\beta(t)}|U_{\mathrm{In}(t)}} = 0\}. & (5)\end{aligned}$$

- Rate region (cone) in terms of rates and source entropies (derived from [Yan, Yeung, Zhang TranIT 2012]):

$$\mathcal{R}(\mathrm{A}) = \mathrm{proj}_{R_\mathcal{E}, H(Y_\mathcal{S})}(\overline{\mathrm{con}(\Gamma_\mathrm{N}^* \cap \mathcal{L}_{123})} \cap \mathcal{L}_{45}) \quad (6)$$

Rate region

- Rate region: all possible rate and source entropy vectors satisfying all network constraints.
- Collect the N network random variables and their joint entropies.
- Define Γ_N^*: in $2^N - 1$-dim. space, region of valid entropy vectors.
- Constraints from network A:

$$\begin{aligned}
\mathcal{L}_1 &= \{\mathbf{h} \in \Gamma_N^* : h_{Y_\mathcal{S}} = \Sigma_{s\in\mathcal{S}} h_{Y_s}\} && (1)\\
\mathcal{L}_2 &= \{\mathbf{h} \in \Gamma_N^* : h_{X_{\mathrm{Out}(k)}|Y_s} = 0\} && (2)\\
\mathcal{L}_3 &= \{\mathbf{h} \in \Gamma_N^* : h_{X_{\mathrm{Out}(i)}|X_{\mathrm{In}(i)}} = 0\} && (3)\\
\mathcal{L}_4 &= \{(\mathbf{h}^T, \mathbf{R}^T)^T \in \mathbb{R}_+^{2^N-1+|\mathcal{E}|} : R_e \geq h_{U_e}, e \in \mathcal{E}\} && (4)\\
\mathcal{L}_5 &= \{\mathbf{h} \in \Gamma_N^* : h_{Y_{\beta(t)}|U_{\mathrm{In}(t)}} = 0\}. && (5)
\end{aligned}$$

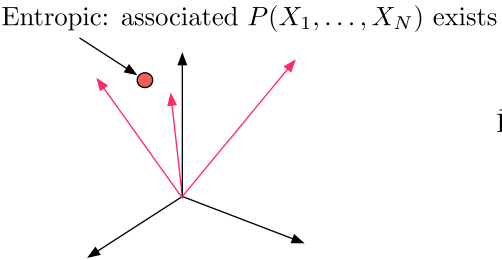
- Rate region (cone) in terms of rates and source entropies (derived from [Yan, Yeung, Zhang TranIT 2012]):

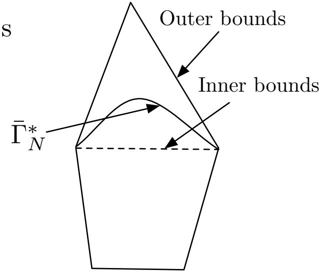
$$\mathcal{R}(\mathrm{A}) = \mathrm{proj}_{R_\mathcal{E}, H(Y_\mathcal{S})}(\overline{\mathrm{con}(\Gamma_\mathrm{N}^* \cap \mathcal{L}_{123})} \cap \mathcal{L}_{45}) \qquad (6)$$

Region of Entropic Vectors

Γ_N^*:

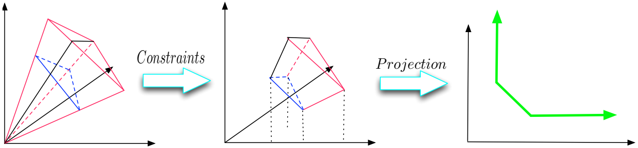
- Not all points in Euclidean space have distributions.
- **h** entropic: exists a joint distribution associated with **h**.
- Region of entropic vector: $\Gamma_N^* = \{\mathbf{h} : \mathbf{h} \text{ is entropic}\}$.
- $\bar{\Gamma}_N^*$ not fully characterized for $N \geq 4$: convex but contains non-polyhedral part

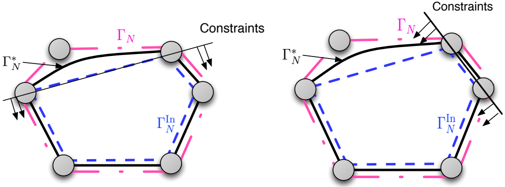




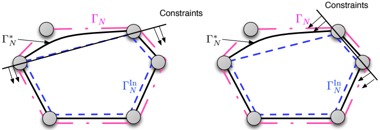
Sandwich Bounds

- $\Gamma_N^* \to \Gamma_N^{Out}$: $\mathcal{R}_{out}(A) = \text{proj}_{R_{\mathcal{E}}, H(Y_{\mathcal{S}})}(\Gamma_N^{Out} \cap \mathcal{L}_{12345})$
- $\Gamma_N^* \to \Gamma_N^{In}$: $\mathcal{R}_{in}(A) = \text{proj}_{R_{\mathcal{E}}, H(Y_{\mathcal{S}})}(\Gamma_N^{In} \cap \mathcal{L}_{12345})$
- It becomes: Initial polyhedra $\to \cap$ constraints $\to$ projections
- $\mathcal{R}(A) = \mathcal{R}_{out}(A) = \mathcal{R}_{in}(A)$, if $\mathcal{R}_{out}(A) = \mathcal{R}_{in}(A)$
- Our work following this idea: Li, *et. al*, Allerton 2012, NetCod 2013, submission TransIT 2014.





Outline

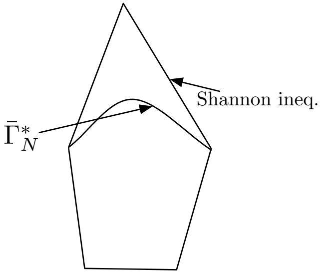


Constraints
Γ_N
Γ_N^*
Γ_N^{In}
Constraints
Γ_N
Γ_N^*
Γ_N^{In}

Shannon Outer Bound

Region determined by Shannon inequalities $H(X_i|\mathcal{N} \setminus X_i) \geq 0$ and $I(X_i, X_j|X_{\mathcal{K}}) \geq 0, i, j \in \mathcal{N}, \mathcal{K} \subseteq \mathcal{N}\setminus\{i, j\}$, equivalent to polymatroid cone:

1. Normalization: $f(\emptyset) = 0$;
2. Monotonicity: if $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{N}$ then $f(\mathcal{A}) \leq f(\mathcal{B})$;
3. Submodularity: if $\mathcal{A}, \mathcal{B} \subseteq \mathcal{N}$, $f(\mathcal{A} \cup \mathcal{B}) + f(\mathcal{A} \cap \mathcal{B}) \leq f(\mathcal{A}) + f(\mathcal{B})$.



Converse Proof

- Recall $\mathcal{R}_{out}(A) = \text{proj}_{R_{\mathcal{E}}, H(Y_{\mathcal{S}})}(\Gamma_N^{Out} \cap \mathcal{L}_{12345})$
- Suppose a polyhedral cone $\mathcal{P}$ $(\Gamma_N^{Out} \cap \mathcal{L}_{12345})$: $\mathbb{A}\mathbf{x} \geq \mathbf{0}$.
- An inequality in the projected cone (rate region), $\mathbf{b}^T\mathbf{x} \geq 0$
- Exists a vector $\boldsymbol{\lambda} \geq \mathbf{0}$ (weighted sum) s. t. $\mathbb{A}^T\boldsymbol{\lambda} = \mathbf{b}$
- This is a converse proof (inspired from Tian ISIT 2013)

$$\overset{\mathbb{A}}{\begin{bmatrix} 1 & 2 & 1 \\ 2 & 1 & 0 \\ -3 & -2 & -1 \end{bmatrix}} \overset{\mathbf{X}}{\begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix}} \geq \overset{\mathbf{0}}{\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}}$$

Projection

$$\overset{\mathbb{A}^T}{\begin{bmatrix} 1 & 2 & -3 \\ 2 & 1 & -2 \\ 1 & 0 & -1 \end{bmatrix}} \overset{\boldsymbol{\lambda}}{\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}} = \overset{\mathbf{b}}{\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}}$$

$$\overset{\mathbf{b}^T}{\begin{bmatrix} 0 & 1 & 0 \end{bmatrix}} \overset{\mathbf{X}}{\begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix}} \geq 0$$

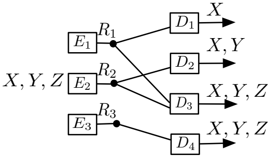
Determine Human Readable Converse Proof

- $\boldsymbol{\lambda}$ should be as sparse as possible
- Optimization

$$\begin{aligned} \underset{\boldsymbol{\lambda}}{\text{minimize}} \quad & \| \boldsymbol{\lambda} \|_0 \\ \text{subject to} \quad & \mathbb{A}^T \boldsymbol{\lambda} = \mathbf{b} \\ & \boldsymbol{\lambda} \geq \mathbf{0}. \end{aligned}$$

- Approximated by L_1-norm: $\| \boldsymbol{\lambda} \|_1$
- We observed: redundant inequalities helpful, the order of inequalities determinable by computer
- Automatic human readable converse (Li, *et. al* submission TransIT 2014): $\Gamma_N^{Out} \cap \mathcal{L}_{12345} + \mathbf{b}^T \mathbf{x} \geq 0 \rightarrow$ LP solver $\rightarrow$ Order determiner $\rightarrow$ Converse proof

Converse Proof Example

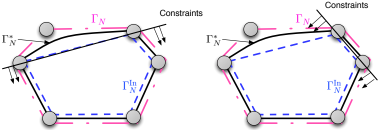


$$\begin{aligned} R_1 &\geq H(X) \\ R_2 &\geq H(X) + H(Y) \\ R_3 &\geq H(X) + H(Y) + H(Z) \\ R_1 + R_2 &\geq 2H(X) + H(Y) + H(Z) \end{aligned}$$

$$\begin{aligned} R_1 + R_2 &\overset{(1,2)}{\geq} H(U_1) + H(U_2) \\ &\overset{(3,4)}{=} H(X, U_1) + H(X, Y, U_2) \\ &\overset{(5)}{\geq} H(X) + H(X, Y, U_1, U_2) \\ &\overset{(6)}{\geq} H(X) + H(U_1, U_2) \\ &\overset{(7)}{\geq} H(X) + H(X, Y, Z) \\ &\overset{(8)}{=} 2H(X) + H(Y) + H(Z) \end{aligned}$$

Order	Coefficients	Inequality or equality
1	1	$R_1 \geq H(U_1)$
2	1	$R_2 \geq H(U_2)$
3	1	$H(X\|U_1) = 0$
4	1	$H(X, Y\|U_2) = 0$
5	1	$I(U_1; YU_2\|X) \geq 0$
6	1	$H(X, Y\|U_1, U_2) = 0$
7	1	$H(X, Y, Z\|U_1, U_2) = 0$
8	1	$H(X, Y, Z) = H(X) + H(Y) + H(Z)$

Outline



Γ_N
Constraints
Γ^*_N
Γ^{In}_N
Γ_N
Constraints
Γ^*_N
Γ^{In}_N

Matroid Definition: Rank Function

Matroid: generalization of linear dependence and independence

Definition (Matroid rank function)

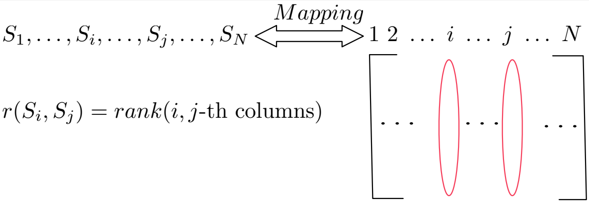
A set function $r : 2^S \to \{0, \ldots, N\}$ is a rank function of a matroid if it obeys the polymatroid axioms and two more conditions:

1. Integrality: $r(A)$ is integer-valued;
2. Cardinality: $0 \leq r(A) \leq |A|$.

Representable Matroids

Definition

A matroid M with ground set S of size $|S| = N$ and rank $r_M = r$ is representable over a field $\mathbb{F}$ if there exists a matrix $\mathbb{A} \in \mathbb{F}^{r \times N}$ such that for each independent set $I \in \mathcal{I}$ the corresponding columns in $\mathbb{A}$, viewed as vectors in $\mathbb{F}^r$, are linearly independent.



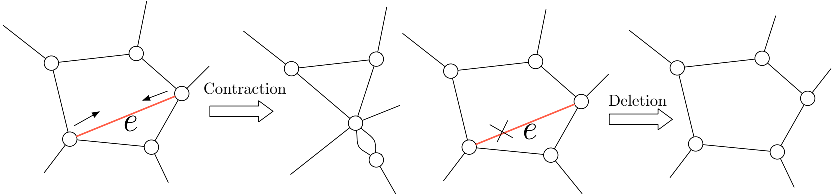
- Representable matroids usually can be characterized by forbidden minors (cannot contain such minors).
- Example: $U_{2,4}$ forbidden for binary (Tutte)

Minors of Matroids

Definition

If M is a matroid on S and $T \subseteq S$, a matroid M' on T is called a *minor* of M if M' is obtained by any combination of deletion ($\backslash$) and contraction ($/$) of M.

Illustration of contraction (conditional) and deletion (unconditional):



Inner Bounds from Representable Matroids

Observation: Conic hull Γ_N^q of representable matroids form inner bounds on (closure of) region of entropic vectors.

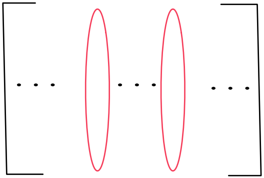
Proof: It suffices to show that a rank of representable matroid is entropic. Suppose the associated representation matrix is $\mathbb{A} \in \mathbb{F}_q^{k \times N}$, from which we can create the random variables. (See also [YeungLiCaiZhang, 2006])

$$(X_1, \ldots, X_N) = \mathbf{u}\mathbb{A}, \quad \mathbf{u} \sim \mathcal{U}(\mathbb{F}_q^k)$$

$$S_1, \ldots, S_i, \ldots, S_j, \ldots, S_N \overset{Mapping}{\Longleftrightarrow} 1\ 2\ \ldots\ i\ \ldots\ j\ \ldots\ N$$

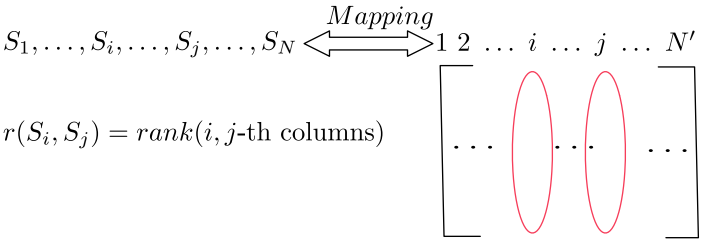
$$r(S_i, S_j) = rank(i, j\text{-th columns})$$

$$h_{i,j} = rank(i, j\text{-th columns}) * h_{\mathbf{u}} = r(S_i, S_j) \log_2 q$$



Generalized Inner Bounds

- $\Gamma^{q}_{N,N'}$: Generalize the mapping to: $S_1, \ldots, S_N \Leftrightarrow N$-partition of $\mathcal{N}', |\mathcal{N}'| > N$.
- Tighter inner bounds on Γ^*_N.



Notion of sufficiency

- Recall: $\mathcal{R}_{in}(\mathrm{A}) = \mathrm{proj}_{R_{\mathcal{E}}, H(Y_{\mathcal{S}})}(\Gamma_N^{In} \cap \mathcal{L}_{12345})$
- Substitute in Γ_N^q: each extreme ray associated with a matroid
- Variable to matrix: one to one mapping, scalar codes, region $\boxed{\mathcal{R}_{s,q}}$
- Substitute in $\Gamma_{N,N'}^q$: each extreme ray associated with a projection of matroids
- Variable to matrix: one to multiple mapping, vector codes, region $\boxed{\mathcal{R}_q}$
- Scalar sufficiency: $\boxed{\mathcal{R}(\mathrm{A}) = \mathcal{R}_{s,q}(\mathrm{A})}$
- General sufficiency: $\boxed{\mathcal{R}(\mathrm{A}) = \mathcal{R}_q(\mathrm{A})}$

Notion of sufficiency

- Recall: $\mathcal{R}_{in}(\mathrm{A}) = \mathrm{proj}_{R_{\mathcal{E}}, H(Y_{\mathcal{S}})}(\Gamma_N^{In} \cap \mathcal{L}_{12345})$
- Substitute in Γ_N^q: each extreme ray associated with a matroid
- Variable to matrix: one to one mapping, scalar codes, region $\boxed{\mathcal{R}_{s,q}}$
- Substitute in $\Gamma_{N,N'}^q$: each extreme ray associated with a projection of matroids
- Variable to matrix: one to multiple mapping, vector codes, region $\boxed{\mathcal{R}_q}$
- Scalar sufficiency: $\boxed{\mathcal{R}(\mathrm{A}) = \mathcal{R}_{s,q}(\mathrm{A})}$
- General sufficiency: $\boxed{\mathcal{R}(\mathrm{A}) = \mathcal{R}_q(\mathrm{A})}$

Notion of sufficiency

- Recall: $\mathcal{R}_{in}(\mathrm{A}) = \mathrm{proj}_{R_{\mathcal{E}}, H(Y_{\mathcal{S}})}(\Gamma_N^{In} \cap \mathcal{L}_{12345})$
- Substitute in Γ_N^q: each extreme ray associated with a matroid
- Variable to matrix: one to one mapping, scalar codes, region $\boxed{\mathcal{R}_{s,q}}$
- Substitute in $\Gamma_{N,N'}^q$: each extreme ray associated with a projection of matroids
- Variable to matrix: one to multiple mapping, vector codes, region $\boxed{\mathcal{R}_q}$
- Scalar sufficiency: $\boxed{\mathcal{R}(\mathrm{A}) = \mathcal{R}_{s,q}(\mathrm{A})}$
- General sufficiency: $\boxed{\mathcal{R}(\mathrm{A}) = \mathcal{R}_q(\mathrm{A})}$

Notion of sufficiency

- Recall: $\mathcal{R}_{in}(\mathrm{A}) = \mathrm{proj}_{R_{\mathcal{E}},H(Y_{\mathcal{S}})}(\Gamma_N^{In} \cap \mathcal{L}_{12345})$
- Substitute in Γ_N^q: each extreme ray associated with a matroid
- Variable to matrix: one to one mapping, scalar codes, region $\boxed{\mathcal{R}_{s,q}}$
- Substitute in $\Gamma_{N,N'}^q$: each extreme ray associated with a projection of matroids
- Variable to matrix: one to multiple mapping, vector codes, region $\boxed{\mathcal{R}_q}$
- Scalar sufficiency: $\boxed{\mathcal{R}(\mathrm{A}) = \mathcal{R}_{s,q}(\mathrm{A})}$
- General sufficiency: $\boxed{\mathcal{R}(\mathrm{A}) = \mathcal{R}_q(\mathrm{A})}$

Notion of sufficiency

- Recall: $\mathcal{R}_{in}(\mathrm{A}) = \mathrm{proj}_{R_{\mathcal{E}},H(Y_{\mathcal{S}})}(\Gamma_N^{In} \cap \mathcal{L}_{12345})$
- Substitute in Γ_N^q: each extreme ray associated with a matroid
- Variable to matrix: one to one mapping, scalar codes, region $\boxed{\mathcal{R}_{s,q}}$
- Substitute in $\Gamma_{N,N'}^q$: each extreme ray associated with a projection of matroids
- Variable to matrix: one to multiple mapping, vector codes, region $\boxed{\mathcal{R}_q}$
- Scalar sufficiency: $\boxed{\mathcal{R}(\mathrm{A}) = \mathcal{R}_{s,q}(\mathrm{A})}$
- General sufficiency: $\boxed{\mathcal{R}(\mathrm{A}) = \mathcal{R}_q(\mathrm{A})}$

Notion of sufficiency

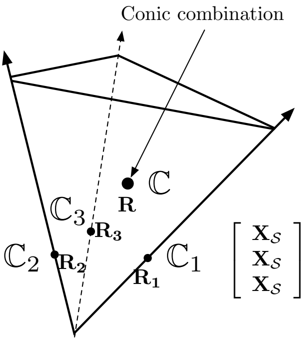
- Recall: $\mathcal{R}_{in}(\text{A}) = \text{proj}_{R_{\mathcal{E}},H(Y_{\mathcal{S}})}(\Gamma_N^{In} \cap \mathcal{L}_{12345})$
- Substitute in Γ_N^q: each extreme ray associated with a matroid
- Variable to matrix: one to one mapping, scalar codes, region $\boxed{\mathcal{R}_{s,q}}$
- Substitute in $\Gamma_{N,N'}^q$: each extreme ray associated with a projection of matroids
- Variable to matrix: one to multiple mapping, vector codes, region $\boxed{\mathcal{R}_q}$
- Scalar sufficiency: $\boxed{\mathcal{R}(\text{A}) = \mathcal{R}_{s,q}(\text{A})}$
- General sufficiency: $\boxed{\mathcal{R}(\text{A}) = \mathcal{R}_q(\text{A})}$

Notion of sufficiency

- Recall: $\mathcal{R}_{in}(\mathsf{A}) = \text{proj}_{R_{\mathcal{E}},H(Y_{\mathcal{S}})}(\Gamma_N^{In} \cap \mathcal{L}_{12345})$
- Substitute in Γ_N^q: each extreme ray associated with a matroid
- Variable to matrix: one to one mapping, scalar codes, region $\boxed{\mathcal{R}_{s,q}}$
- Substitute in $\Gamma_{N,N'}^q$: each extreme ray associated with a projection of matroids
- Variable to matrix: one to multiple mapping, vector codes, region $\boxed{\mathcal{R}_q}$
- Scalar sufficiency: $\boxed{\mathcal{R}(\mathsf{A}) = \mathcal{R}_{s,q}(\mathsf{A})}$
- General sufficiency: $\boxed{\mathcal{R}(\mathsf{A}) = \mathcal{R}_q(\mathsf{A})}$

Representable Matroids Determine Linear Codes

- Basic codes: matrices associated with extreme rays
- A point: conic combination of extreme rays
- Code for the point: block diagonalize the matrices according to conic coefficients (time sharing), reshuffling columns (details in [Li, *et. al* NetCod 2013])



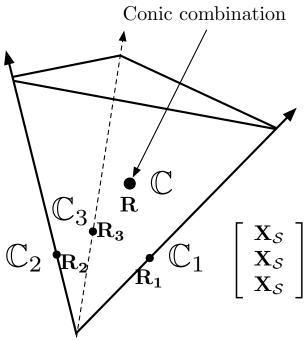
$\mathbb{C}_1, \mathbb{C}_2, \mathbb{C}_3$: basic codes

$$\mathbf{R} = \mathbf{R}_1 + \mathbf{R}_2 + \mathbf{R}_3$$

$$\begin{bmatrix} \mathbf{X}_\mathcal{S} \\ \mathbf{X}_\mathcal{S} \\ \mathbf{X}_\mathcal{S} \end{bmatrix}^T \mathbb{C} = \begin{bmatrix} \mathbf{X}_\mathcal{S} \\ \mathbf{X}_\mathcal{S} \\ \mathbf{X}_\mathcal{S} \end{bmatrix}^T \begin{array}{c} \begin{array}{ccc} \mathbf{U}_\mathcal{E} & \mathbf{U}_\mathcal{E} & \mathbf{U}_\mathcal{E} \end{array} \\ \begin{bmatrix} \mathbb{C}_1 & \mathbb{O} & \mathbb{O} \\ \mathbb{O} & \mathbb{C}_2 & \mathbb{O} \\ \mathbb{O} & \mathbb{O} & \mathbb{C}_3 \end{bmatrix} \end{array}$$

Representable Matroids Determine Linear Codes

- Basic codes: matrices associated with extreme rays
- A point: conic combination of extreme rays
- Code for the point: block diagonalize the matrices according to conic coefficients (time sharing), reshuffling columns (details in [Li, *et. al* NetCod 2013])



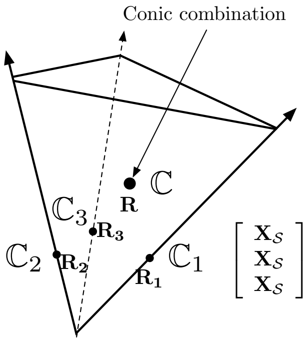
$\mathbb{C}_1, \mathbb{C}_2, \mathbb{C}_3$: basic codes

$$\mathbf{R} = \mathbf{R}_1 + \mathbf{R}_2 + \mathbf{R}_3$$

$$\begin{bmatrix} \mathbf{X}_\mathcal{S} \\ \mathbf{X}_\mathcal{S} \\ \mathbf{X}_\mathcal{S} \end{bmatrix}^T \mathbb{C} = \begin{bmatrix} \mathbf{X}_\mathcal{S} \\ \mathbf{X}_\mathcal{S} \\ \mathbf{X}_\mathcal{S} \end{bmatrix}^T \overset{\quad\mathbf{U}_\mathcal{E} \quad\; \mathbf{U}_\mathcal{E} \quad\; \mathbf{U}_\mathcal{E}}{\begin{bmatrix} \mathbb{C}_1 & \mathbb{O} & \mathbb{O} \\ \mathbb{O} & \mathbb{C}_2 & \mathbb{O} \\ \mathbb{O} & \mathbb{O} & \mathbb{C}_3 \end{bmatrix}}$$

Representable Matroids Determine Linear Codes

- Basic codes: matrices associated with extreme rays
- A point: conic combination of extreme rays
- Code for the point: block diagonalize the matrices according to conic coefficients (time sharing), reshuffling columns (details in [Li, *et. al* NetCod 2013])



$\mathbb{C}_1, \mathbb{C}_2, \mathbb{C}_3$: basic codes

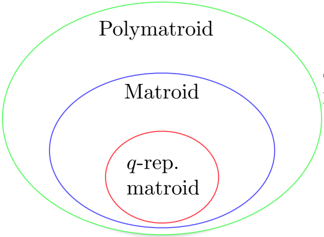
$\mathbf{R} = \mathbf{R}_1 + \mathbf{R}_2 + \mathbf{R}_3$

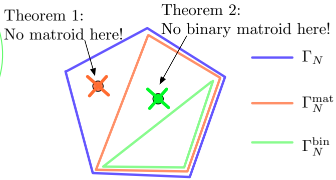
$$\begin{bmatrix} \mathbf{X}_\mathcal{S} \\ \mathbf{X}_\mathcal{S} \\ \mathbf{X}_\mathcal{S} \end{bmatrix}^T \mathbb{C} = \begin{bmatrix} \mathbf{X}_\mathcal{S} \\ \mathbf{X}_\mathcal{S} \\ \mathbf{X}_\mathcal{S} \end{bmatrix}^T \begin{array}{c} \begin{array}{ccc} \mathbf{U}_\mathcal{E} & \mathbf{U}_\mathcal{E} & \mathbf{U}_\mathcal{E} \end{array} \\ \begin{bmatrix} \mathbb{C}_1 & \mathbb{O} & \mathbb{O} \\ \mathbb{O} & \mathbb{C}_2 & \mathbb{O} \\ \mathbb{O} & \mathbb{O} & \mathbb{C}_3 \end{bmatrix} \end{array}$$

Matroid extremality

- Suffice to work on rep. matroids that are extreme rays (complexity reduction, Li, *et. al* NetCod 2013, further work Apte, *et. al* ISIT 2014)
- Extremal ranks relationship: strict containment [Li, *et. al* Allerton 2013]

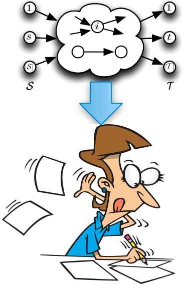
Extreme rays of different cones



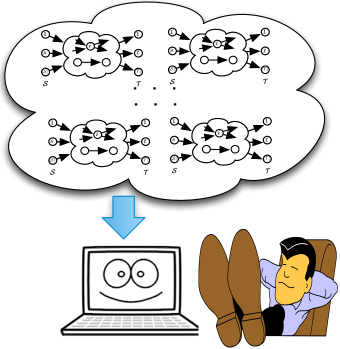


Revolution

- Conventional: 1 (special) network, info. ineq., manually, 1 paper
- Computational: $10^3, 10^6$ (arbitrary) networks, computer, # of papers?
- Improvement: easy to handle small networks, further work on symmetries, *inheritance*



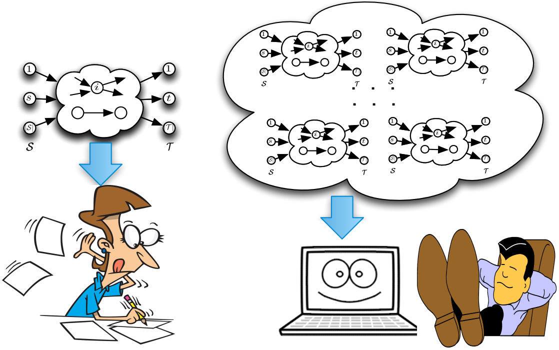
Conventional way



Computational way

Revolution

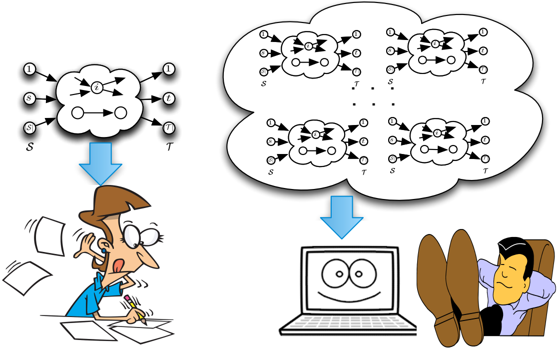
- Conventional: 1 (special) network, info. ineq., manually, 1 paper
- Computational: $10^3, 10^6$ (arbitrary) networks, computer, # of papers?
- Improvement: easy to handle small networks, further work on symmetries, *inheritance*



Conventional way　　Computational way

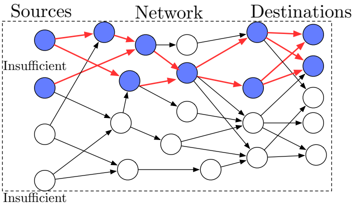
Revolution

- Conventional: 1 (special) network, info. ineq., manually, 1 paper
- Computational: $10^3, 10^6$ (arbitrary) networks, computer, # of papers?
- Improvement: easy to handle small networks, further work on symmetries, *inheritance*



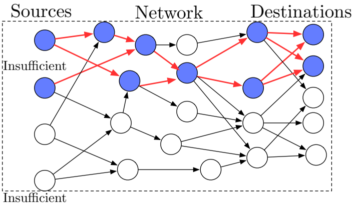
Conventional way　　Computational way

Outline



Sources
Network
Destinations
Insufficient
Insufficient

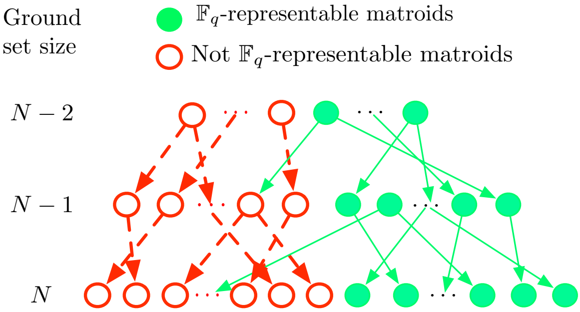
Outline



Sources
Network
Destinations
Insufficient
Insufficient

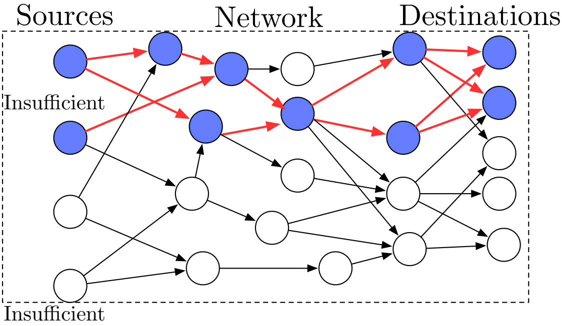
Motivated from matroid theory

- Inheritance property regarding insufficiency of class of codes
- Inspired from matroid theory: forbidden minor for linear representability, e.g., $U_{2,4}$ is the forbidden minor for binary matroids

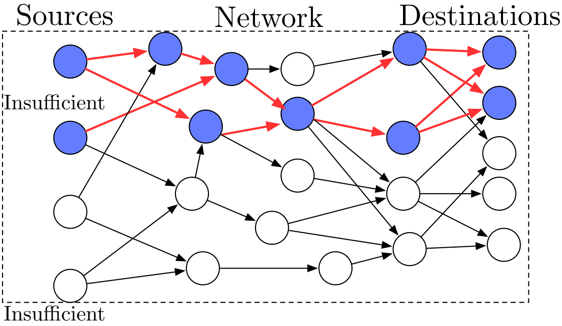


Similar characterization for networks?

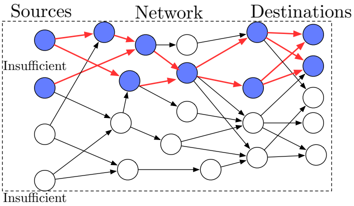
- If networks have similar characterization? Possible list of forbidden embedded networks for sufficiency of linear codes over a field.
- Network operations to obtain such embedded networks preserving insufficiency, & region relationships
- Three operations [Li, *et. al*, Allerton 2014]



- If networks have similar characterization? Possible list of forbidden embedded networks for sufficiency of linear codes over a field.
- Network operations to obtain such embedded networks preserving insufficiency, & region relationships
- Three operations [Li, *et. al*, Allerton 2014]

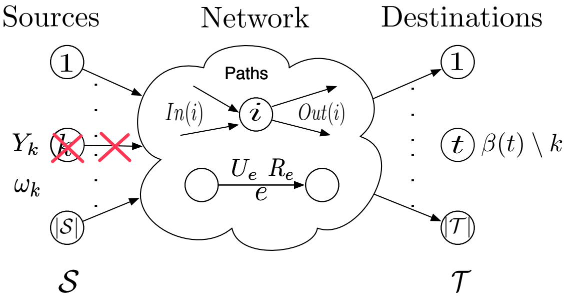


Outline



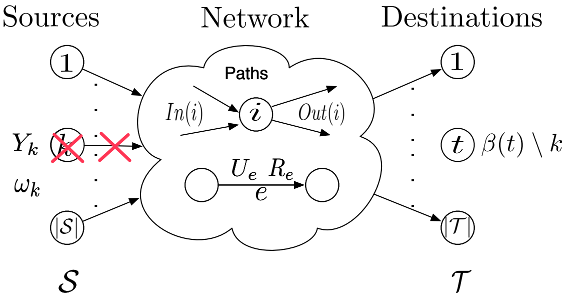
Sources
Network
Destinations
Insufficient
Insufficient

- $A' = A \setminus k$
- Source Y_k deleted, source k stops sending information to the network, $H(Y_k) = 0$
- Sinks requiring Y_k will no longer demand it.



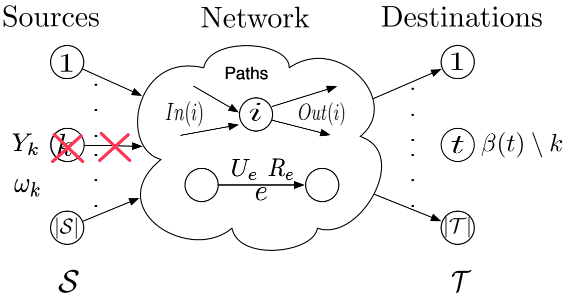
Source deletion

- $\mathsf{A}' = \mathsf{A} \setminus k$
- Source Y_k deleted, source k stops sending information to the network, $H(Y_k) = 0$
- Sinks requiring Y_k will no longer demand it.

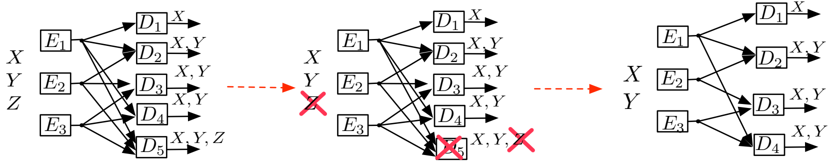


Source deletion

- $A' = A \setminus k$
- Source Y_k deleted, source k stops sending information to the network, $H(Y_k) = 0$
- Sinks requiring Y_k will no longer demand it.



Example: MDCS source deletion



i) A (3, 3) MDCS *ii*) Delete/contract Z *iii*) After deletion of Z

- Decoder D_5 no longer requires Z
- Redundant decoder is deleted

Source deletion: $\mathsf{A}' = \mathsf{A} \setminus k$

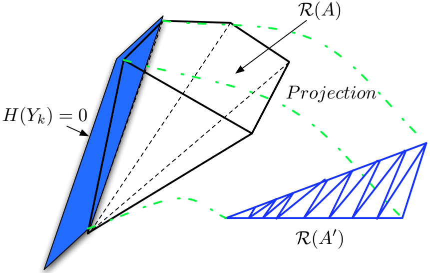
$$\mathcal{R}(\mathsf{A}') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}} \left(\{ \mathbf{R} \in \mathcal{R}(\mathsf{A}) \,|\, H(Y_k) = 0 \} \right), \tag{7}$$

$$\mathcal{R}_q(\mathsf{A}') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}} \left(\{ \mathbf{R} \in \mathcal{R}_q(\mathsf{A}) \,|\, H(Y_k) = 0 \} \right), \tag{8}$$

$$\mathcal{R}_{s,q}(\mathsf{A}') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}} \left(\{ \mathbf{R} \in \mathcal{R}_{s,q}(\mathsf{A}) \,|\, H(Y_k) = 0 \} \right). \tag{9}$$

- Preservation: $\mathcal{R}_q(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_q(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- Scalar case: $\mathcal{R}_{s,q}(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_{s,q}(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$

- $\mathbf{r}' \in \mathcal{R}(\mathsf{A}') \leftrightarrow \mathbf{r} \in \mathcal{R}(\mathsf{A})$ with $H(Y_k) = 0$
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with sending nothing on source Y_k. All-zero is valid $\mathbb{F}_q$ code, can reverse.



Source deletion: $A' = A \setminus k$

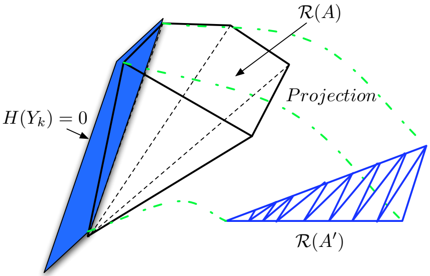
$$\mathcal{R}(A') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}} \left(\{ \mathbf{R} \in \mathcal{R}(A) \,|\, H(Y_k) = 0 \} \right), \tag{7}$$

$$\mathcal{R}_q(A') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}} \left(\{ \mathbf{R} \in \mathcal{R}_q(A) \,|\, H(Y_k) = 0 \} \right), \tag{8}$$

$$\mathcal{R}_{s,q}(A') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}} \left(\{ \mathbf{R} \in \mathcal{R}_{s,q}(A) \,|\, H(Y_k) = 0 \} \right). \tag{9}$$

- Preservation: $\mathcal{R}_q(A) = \mathcal{R}(A) \Rightarrow \mathcal{R}_q(A') = \mathcal{R}(A')$
- Scalar case: $\mathcal{R}_{s,q}(A) = \mathcal{R}(A) \Rightarrow \mathcal{R}_{s,q}(A') = \mathcal{R}(A')$

- $\mathbf{r}' \in \mathcal{R}(A') \leftrightarrow \mathbf{r} \in \mathcal{R}(A)$ with $H(Y_k) = 0$
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with sending nothing on source Y_k. All-zero is valid $\mathbb{F}_q$ code, can reverse.



Source deletion: $\mathsf{A}' = \mathsf{A} \setminus k$

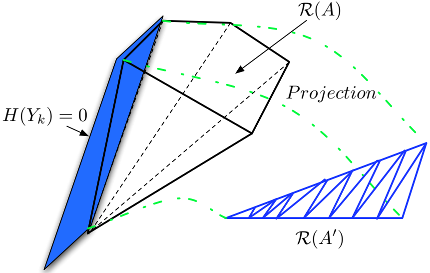
$$\mathcal{R}(\mathsf{A}') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}}\left(\{\mathbf{R} \in \mathcal{R}(\mathsf{A}) \,|\, H(Y_k) = 0\}\right), \tag{7}$$

$$\mathcal{R}_q(\mathsf{A}') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}}\left(\{\mathbf{R} \in \mathcal{R}_q(\mathsf{A}) \,|\, H(Y_k) = 0\}\right), \tag{8}$$

$$\mathcal{R}_{s,q}(\mathsf{A}') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}}\left(\{\mathbf{R} \in \mathcal{R}_{s,q}(\mathsf{A}) \,|\, H(Y_k) = 0\}\right). \tag{9}$$

- Preservation: $\mathcal{R}_q(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_q(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- Scalar case: $\mathcal{R}_{s,q}(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_{s,q}(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$

- $\mathbf{r}' \in \mathcal{R}(\mathsf{A}') \leftrightarrow \mathbf{r} \in \mathcal{R}(\mathsf{A})$ with $H(Y_k) = 0$
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with sending nothing on source Y_k. All-zero is valid $\mathbb{F}_q$ code, can reverse.



Source deletion: $\mathsf{A}' = \mathsf{A} \setminus k$

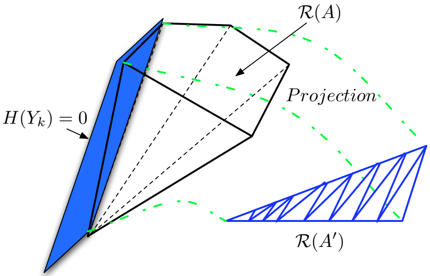
$$\mathcal{R}(\mathsf{A}') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}} \left(\{ \mathbf{R} \in \mathcal{R}(\mathsf{A}) \,|\, H(Y_k) = 0 \} \right), \tag{7}$$

$$\mathcal{R}_q(\mathsf{A}') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}} \left(\{ \mathbf{R} \in \mathcal{R}_q(\mathsf{A}) \,|\, H(Y_k) = 0 \} \right), \tag{8}$$

$$\mathcal{R}_{s,q}(\mathsf{A}') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}} \left(\{ \mathbf{R} \in \mathcal{R}_{s,q}(\mathsf{A}) \,|\, H(Y_k) = 0 \} \right). \tag{9}$$

- Preservation: $\mathcal{R}_q(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_q(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- Scalar case: $\mathcal{R}_{s,q}(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_{s,q}(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$

- $\mathbf{r}' \in \mathcal{R}(\mathsf{A}') \leftrightarrow \mathbf{r} \in \mathcal{R}(\mathsf{A})$ with $H(Y_k) = 0$
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with sending nothing on source Y_k. All-zero is valid $\mathbb{F}_q$ code, can reverse.



Source deletion: $\mathsf{A}' = \mathsf{A} \setminus k$

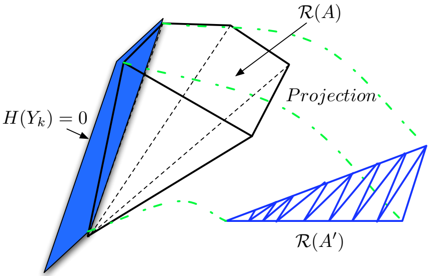
$$\mathcal{R}(\mathsf{A}') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}} \left(\{ \mathbf{R} \in \mathcal{R}(\mathsf{A}) \,|\, H(Y_k) = 0 \} \right), \quad (7)$$

$$\mathcal{R}_q(\mathsf{A}') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}} \left(\{ \mathbf{R} \in \mathcal{R}_q(\mathsf{A}) \,|\, H(Y_k) = 0 \} \right), \quad (8)$$

$$\mathcal{R}_{s,q}(\mathsf{A}') = \mathrm{Proj}_{\mathbf{Y}_{\setminus k}, \mathbf{R}_{\mathcal{E}'}} \left(\{ \mathbf{R} \in \mathcal{R}_{s,q}(\mathsf{A}) \,|\, H(Y_k) = 0 \} \right). \quad (9)$$

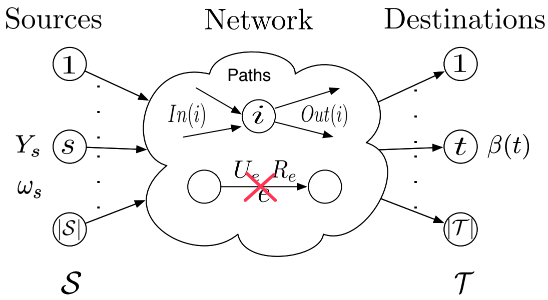
- Preservation: $\mathcal{R}_q(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_q(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- Scalar case: $\mathcal{R}_{s,q}(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_{s,q}(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$

- $\mathbf{r}' \in \mathcal{R}(\mathsf{A}') \leftrightarrow \mathbf{r} \in \mathcal{R}(\mathsf{A})$ with $H(Y_k) = 0$
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with sending nothing on source Y_k. All-zero is valid $\mathbb{F}_q$ code, can reverse.



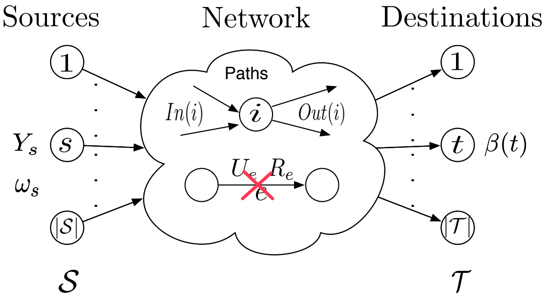
Edge deletion

- $A' = A \setminus e$
- Edge U_e deleted, nothing on U_e, $R_e = H(U_e) = 0$.

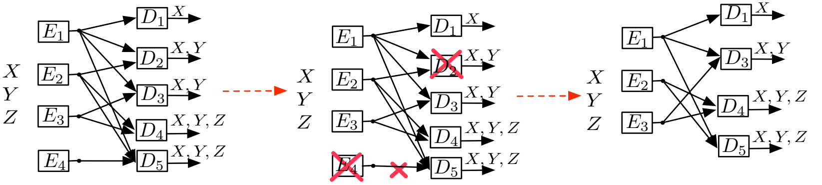


Edge deletion

- $\mathsf{A}' = \mathsf{A} \setminus e$
- Edge U_e deleted, nothing on U_e, $R_e = H(U_e) = 0$.



Example: MDCS encoder deletion



i) A (3, 4) MDCS   *ii*) Delete encoder E_4 *iii*) After deletion of E_4

- Decoders connected to E_4 keep same decoding ability without E_4
- Redundant decoders deleted

Edge deletion: $\mathsf{A}' = \mathsf{A} \setminus Y_k$

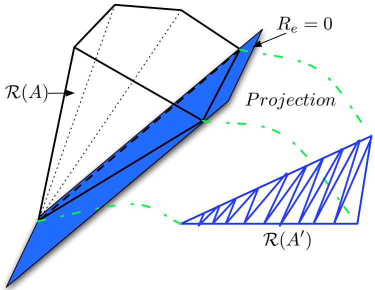
$$\mathcal{R}(\mathsf{A}') = \mathrm{Proj}_{H(Y_{\mathcal{S}}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_q(\mathsf{A}) | R_e = 0\}), \quad (10)$$

$$\mathcal{R}_q(\mathsf{A}') = \mathrm{Proj}_{H(Y_{\mathcal{S}}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_q(\mathsf{A}) | R_e = 0\}), \quad (11)$$

$$\mathcal{R}_{s,q}(\mathsf{A}') = \mathrm{Proj}_{H(Y_{\mathcal{S}}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_{s,q}(\mathsf{A}) | R_e = 0\}). \quad (12)$$

- Preservation: $\mathcal{R}_q(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_q(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- Scalar case: $\mathcal{R}_{s,q}(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_{s,q}(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$

- $\mathbf{r}' \in \mathcal{R}(\mathsf{A}') \leftrightarrow \mathbf{r} \in \mathcal{R}(\mathsf{A})$ with $R_e = H(U_e) = 0$
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with deleting column(s) for U_e, i.e., sending nothing on edge e. All-zero is valid $\mathbb{F}_q$ code, can reverse.



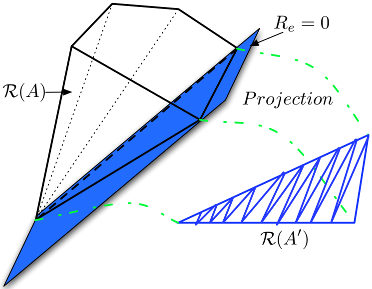
Edge deletion: $\mathsf{A}' = \mathsf{A} \setminus Y_k$

$$\mathcal{R}(\mathsf{A}') = \mathrm{Proj}_{H(Y_\mathcal{S}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_q(\mathsf{A}) | R_e = 0\}), \tag{10}$$

$$\mathcal{R}_q(\mathsf{A}') = \mathrm{Proj}_{H(Y_\mathcal{S}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_q(\mathsf{A}) | R_e = 0\}), \tag{11}$$

$$\mathcal{R}_{s,q}(\mathsf{A}') = \mathrm{Proj}_{H(Y_\mathcal{S}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_{s,q}(\mathsf{A}) | R_e = 0\}). \tag{12}$$

- Preservation: $\mathcal{R}_q(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_q(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- Scalar case: $\mathcal{R}_{s,q}(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_{s,q}(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- $\mathbf{r}' \in \mathcal{R}(\mathsf{A}') \leftrightarrow \mathbf{r} \in \mathcal{R}(\mathsf{A})$ with $R_e = H(U_e) = 0$
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with deleting column(s) for U_e, i.e., sending nothing on edge e. All-zero is valid $\mathbb{F}_q$ code, can reverse.



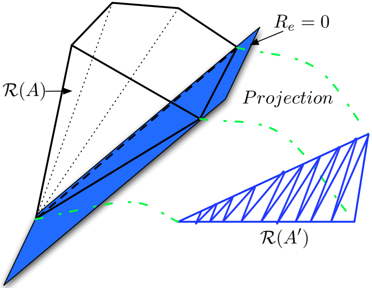
Edge deletion: $\mathsf{A}' = \mathsf{A} \setminus Y_k$

$$\mathcal{R}(\mathsf{A}') = \mathrm{Proj}_{H(Y_\mathcal{S}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_q(\mathsf{A}) | R_e = 0\}), \quad (10)$$

$$\mathcal{R}_q(\mathsf{A}') = \mathrm{Proj}_{H(Y_\mathcal{S}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_q(\mathsf{A}) | R_e = 0\}), \quad (11)$$

$$\mathcal{R}_{s,q}(\mathsf{A}') = \mathrm{Proj}_{H(Y_\mathcal{S}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_{s,q}(\mathsf{A}) | R_e = 0\}). \quad (12)$$

- Preservation: $\mathcal{R}_q(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_q(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- Scalar case: $\mathcal{R}_{s,q}(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_{s,q}(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- $\mathbf{r}' \in \mathcal{R}(\mathsf{A}') \leftrightarrow \mathbf{r} \in \mathcal{R}(\mathsf{A})$ with $R_e = H(U_e) = 0$
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with deleting column(s) for U_e, i.e., sending nothing on edge e. All-zero is valid $\mathbb{F}_q$ code, can reverse.



Edge deletion: $\mathsf{A}' = \mathsf{A} \setminus Y_k$

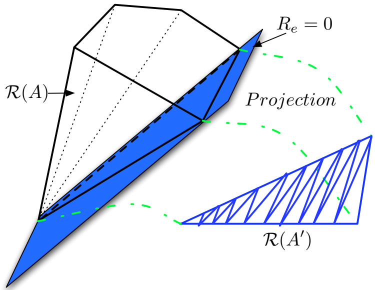
$$\mathcal{R}(\mathsf{A}') = \mathrm{Proj}_{H(Y_{\mathcal{S}}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_q(\mathsf{A}) | R_e = 0\}), \tag{10}$$

$$\mathcal{R}_q(\mathsf{A}') = \mathrm{Proj}_{H(Y_{\mathcal{S}}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_q(\mathsf{A}) | R_e = 0\}), \tag{11}$$

$$\mathcal{R}_{s,q}(\mathsf{A}') = \mathrm{Proj}_{H(Y_{\mathcal{S}}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_{s,q}(\mathsf{A}) | R_e = 0\}). \tag{12}$$

- Preservation: $\mathcal{R}_q(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_q(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- Scalar case: $\mathcal{R}_{s,q}(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_{s,q}(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$

- $\mathbf{r}' \in \mathcal{R}(\mathsf{A}') \leftrightarrow \mathbf{r} \in \mathcal{R}(\mathsf{A})$ with $R_e = H(U_e) = 0$
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with deleting column(s) for U_e, i.e., sending nothing on edge e. All-zero is valid $\mathbb{F}_q$ code, can reverse.



Edge deletion: $A' = A \setminus Y_k$

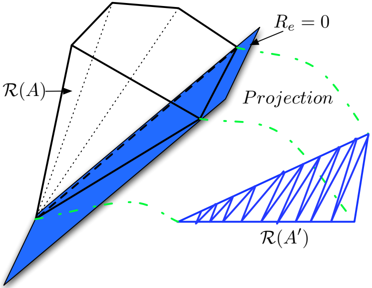
$$\mathcal{R}(A') = \mathrm{Proj}_{H(Y_\mathcal{S}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_q(A) | R_e = 0\}), \quad (10)$$

$$\mathcal{R}_q(A') = \mathrm{Proj}_{H(Y_\mathcal{S}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_q(A) | R_e = 0\}), \quad (11)$$

$$\mathcal{R}_{s,q}(A') = \mathrm{Proj}_{H(Y_\mathcal{S}),\mathbf{R}_{\mathcal{E}'}}(\{\mathbf{R} \in \mathcal{R}_{s,q}(A) | R_e = 0\}). \quad (12)$$

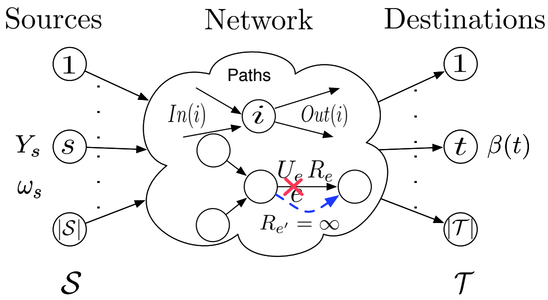
- Preservation: $\mathcal{R}_q(A) = \mathcal{R}(A) \Rightarrow \mathcal{R}_q(A') = \mathcal{R}(A')$
- Scalar case: $\mathcal{R}_{s,q}(A) = \mathcal{R}(A) \Rightarrow \mathcal{R}_{s,q}(A') = \mathcal{R}(A')$

- $\mathbf{r}' \in \mathcal{R}(A') \leftrightarrow \mathbf{r} \in \mathcal{R}(A)$ with $R_e = H(U_e) = 0$
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with deleting column(s) for U_e, i.e., sending nothing on edge e. All-zero is valid $\mathbb{F}_q$ code, can reverse.



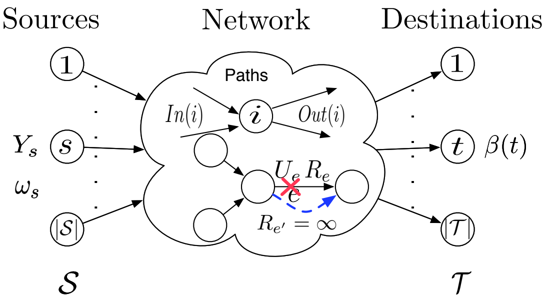
Edge contraction

- $A' = A/e$
- Edge e contracted, input to tail of e available for head of e, $R_e = \infty$, $H(U_e)$ free.

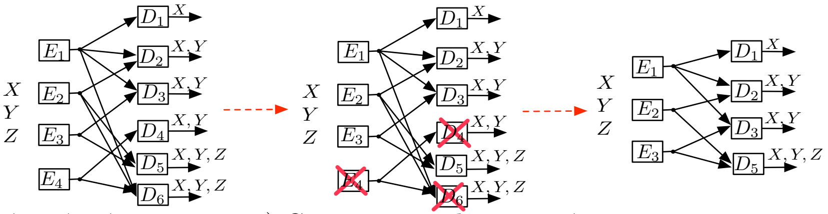


Edge contraction

- $A' = A/e$
- Edge e contracted, input to tail of e available for head of e, $R_e = \infty$, $H(U_e)$ free.



Example: MDCS encoder contraction



$i)$ A $(3, 4)$ MDCS $ii)$ Contract encoder E_4 $iii)$ After contraction of E_4

- Decoders connected to E_4 directly access to all sources
- Requirements trivially satisfied, deleted

Edge contraction: $\mathsf{A}' = \mathsf{A}/e$

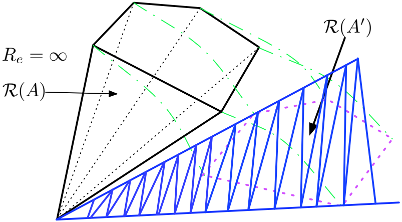
$$\mathcal{R}(\mathsf{A}') = \mathrm{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}(\mathsf{A}), \quad (13)$$

$$\mathcal{R}_q(\mathsf{A}') = \mathrm{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}_q(\mathsf{A}), \quad (14)$$

$$\mathcal{R}_{s,q}(\mathsf{A}') \supseteq \mathrm{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}_{s,q}(\mathsf{A}), \quad (15)$$

- Preservation: $\mathcal{R}_q(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_q(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- Scalar case: $\mathcal{R}_{s,q}(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_{s,q}(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$

- $\mathbf{r}' \in \mathcal{R}(\mathsf{A}') \leftrightarrow \mathbf{r} \in \mathcal{R}(\mathsf{A})$ with some $H(U_e)$ (not of interest in A')
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with deleting column(s) for U_e, sending possibly everything available on e, possibly violating scalar code.



Edge contraction: $\mathsf{A}' = \mathsf{A}/e$

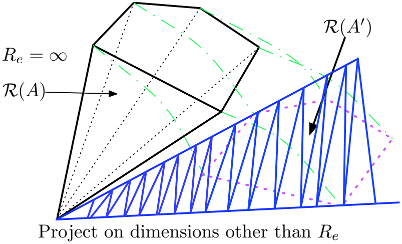
$$\mathcal{R}(\mathsf{A}') = \mathrm{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}(\mathsf{A}), \quad (13)$$

$$\mathcal{R}_q(\mathsf{A}') = \mathrm{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}_q(\mathsf{A}), \quad (14)$$

$$\mathcal{R}_{s,q}(\mathsf{A}') \supseteq \mathrm{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}_{s,q}(\mathsf{A}), \quad (15)$$

- Preservation: $\mathcal{R}_q(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_q(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- Scalar case: $\mathcal{R}_{s,q}(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_{s,q}(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$

- $\mathbf{r}' \in \mathcal{R}(\mathsf{A}') \leftrightarrow \mathbf{r} \in \mathcal{R}(\mathsf{A})$ with some $H(U_e)$ (not of interest in A')
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with deleting column(s) for U_e, sending possibly everything available on e, possibly violating scalar code.



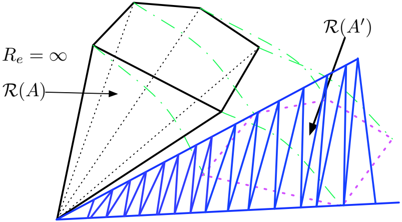
Edge contraction: $\mathsf{A}' = \mathsf{A}/e$

$$\mathcal{R}(\mathsf{A}') = \mathrm{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}(\mathsf{A}), \quad (13)$$

$$\mathcal{R}_q(\mathsf{A}') = \mathrm{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}_q(\mathsf{A}), \quad (14)$$

$$\mathcal{R}_{s,q}(\mathsf{A}') \supseteq \mathrm{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}_{s,q}(\mathsf{A}), \quad (15)$$

- Preservation: $\mathcal{R}_q(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_q(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- Scalar case: $\mathcal{R}_{s,q}(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_{s,q}(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- $\mathbf{r}' \in \mathcal{R}(\mathsf{A}') \leftrightarrow \mathbf{r} \in \mathcal{R}(\mathsf{A})$ with some $H(U_e)$ (not of interest in A')
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with deleting column(s) for U_e, sending possibly everything available on e, possibly violating scalar code.



Edge contraction: $\mathsf{A}' = \mathsf{A}/e$

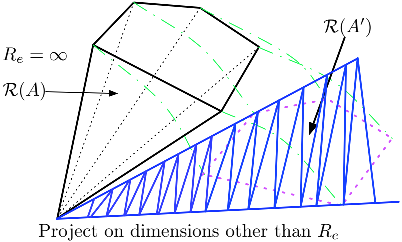
$$\mathcal{R}(\mathsf{A}') = \mathrm{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}(\mathsf{A}), \quad (13)$$

$$\mathcal{R}_q(\mathsf{A}') = \mathrm{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}_q(\mathsf{A}), \quad (14)$$

$$\mathcal{R}_{s,q}(\mathsf{A}') \supseteq \mathrm{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}_{s,q}(\mathsf{A}), \quad (15)$$

- Preservation: $\mathcal{R}_q(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_q(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$
- Scalar case: $\mathcal{R}_{s,q}(\mathsf{A}) = \mathcal{R}(\mathsf{A}) \Rightarrow \mathcal{R}_{s,q}(\mathsf{A}') = \mathcal{R}(\mathsf{A}')$

- $\mathbf{r}' \in \mathcal{R}(\mathsf{A}') \leftrightarrow \mathbf{r} \in \mathcal{R}(\mathsf{A})$ with some $H(U_e)$ (not of interest in A')
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with deleting column(s) for U_e, sending possibly everything available on e, possibly violating scalar code.



Edge contraction: $A' = A/e$

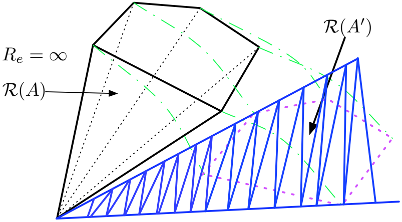
$$\mathcal{R}(A') = \text{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}(A), \quad (13)$$

$$\mathcal{R}_q(A') = \text{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}_q(A), \quad (14)$$

$$\mathcal{R}_{s,q}(A') \supseteq \text{Proj}_{H(Y_k),k\in\mathcal{S},\mathbf{R}_{\mathcal{E}'}}\mathcal{R}_{s,q}(A), \quad (15)$$

- Preservation: $\mathcal{R}_q(A) = \mathcal{R}(A) \Rightarrow \mathcal{R}_q(A') = \mathcal{R}(A')$
- Scalar case: $\mathcal{R}_{s,q}(A) = \mathcal{R}(A) \Rightarrow \mathcal{R}_{s,q}(A') = \mathcal{R}(A')$

- $\mathbf{r}' \in \mathcal{R}(A') \leftrightarrow \mathbf{r} \in \mathcal{R}(A)$ with some $H(U_e)$ (not of interest in A')
- The code to achieve $\mathbf{r}'$ is the code to achieve $\mathbf{r}$ with deleting column(s) for U_e, sending possibly everything available on e, possibly violating scalar code.



Outline

1 Rate Region

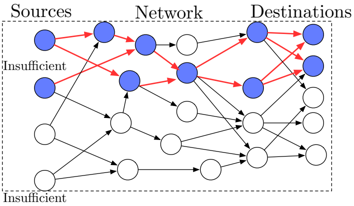
2 Shannon Outer Bounds & Converse

3 Matroid Bounds and Achievability

4 Forbidden Embedding Networks

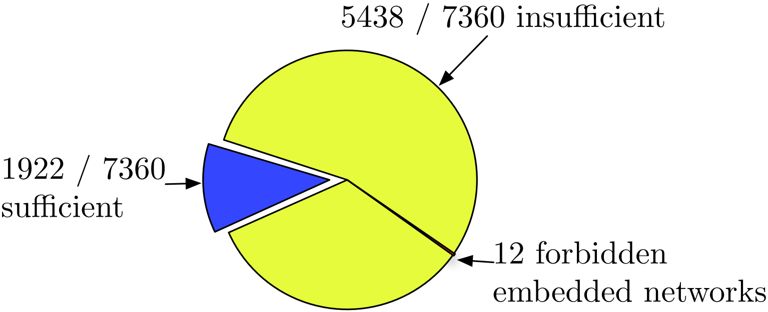
- Motivation
- Operations
- Results

5 Future work



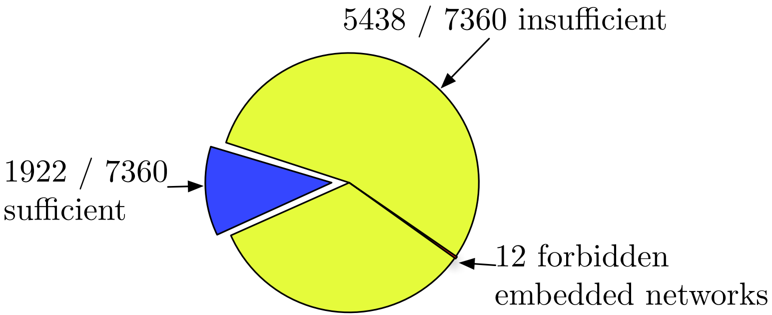
Example: Forbidden embedded networks

- Goal: minimal forbidden networks for sufficiency
- Scalar binary codes considered
- $k = 1, 2, 3$; $|\mathcal{E}| = 2, 3, 4$, 7360 non-isomorphic MDCS
- 1922 sufficient, 5438 insufficient
- 12 forbidden embedded networks (Li, *et. al* submission TransIT 2014)
- Open questions: complete list? Finite number for $\mathbb{F}_q$?



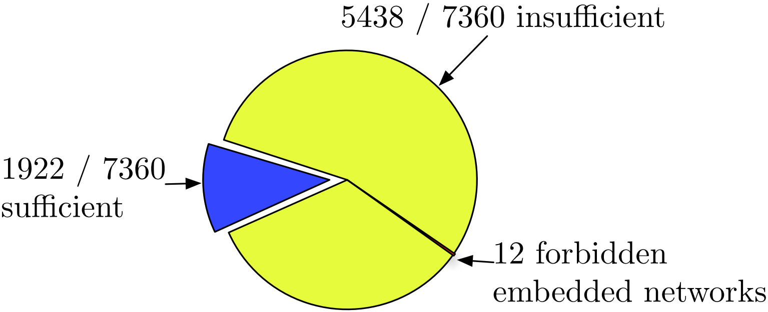
Example: Forbidden embedded networks

- Goal: minimal forbidden networks for sufficiency
- Scalar binary codes considered
- $k = 1, 2, 3$; $|\mathcal{E}| = 2, 3, 4$, 7360 non-isomorphic MDCS
- 1922 sufficient, 5438 insufficient
- 12 forbidden embedded networks (Li, *et. al* submission TransIT 2014)
- Open questions: complete list? Finite number for $\mathbb{F}_q$?



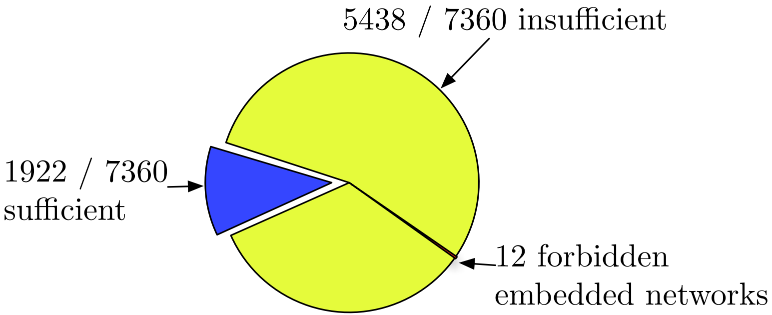
Example: Forbidden embedded networks

- Goal: minimal forbidden networks for sufficiency
- Scalar binary codes considered
- $k = 1, 2, 3$; $|\mathcal{E}| = 2, 3, 4$, 7360 non-isomorphic MDCS
- 1922 sufficient, 5438 insufficient
- 12 forbidden embedded networks (Li, *et. al* submission TransIT 2014)
- Open questions: complete list? Finite number for $\mathbb{F}_q$?



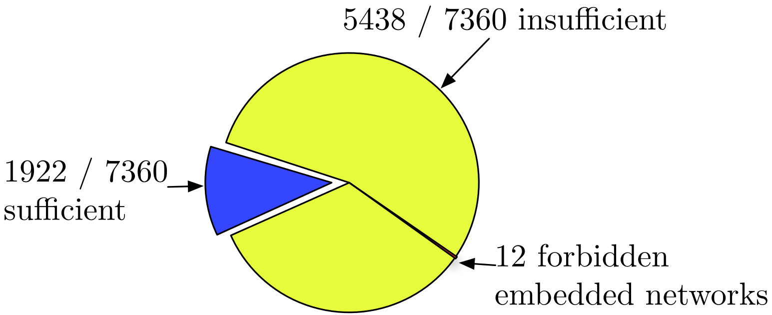
Example: Forbidden embedded networks

- Goal: minimal forbidden networks for sufficiency
- Scalar binary codes considered
- $k = 1, 2, 3; |\mathcal{E}| = 2, 3, 4$, 7360 non-isomorphic MDCS
- 1922 sufficient, 5438 insufficient
- 12 forbidden embedded networks (Li, *et. al* submission TransIT 2014)
- Open questions: complete list? Finite number for $\mathbb{F}_q$?



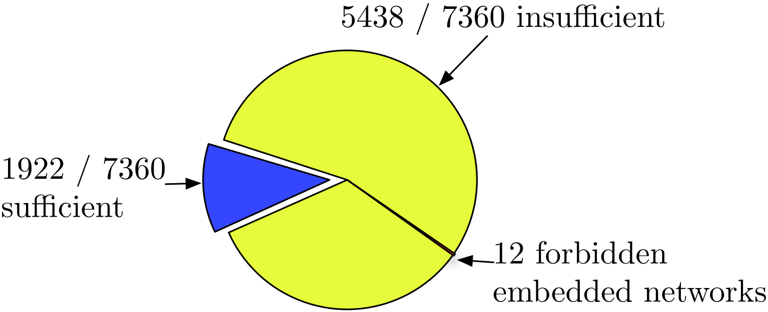
Example: Forbidden embedded networks

- Goal: minimal forbidden networks for sufficiency
- Scalar binary codes considered
- $k = 1, 2, 3$; $|\mathcal{E}| = 2, 3, 4$, 7360 non-isomorphic MDCS
- 1922 sufficient, 5438 insufficient
- 12 forbidden embedded networks (Li, *et. al* submission TransIT 2014)
- Open questions: complete list? Finite number for $\mathbb{F}_q$?

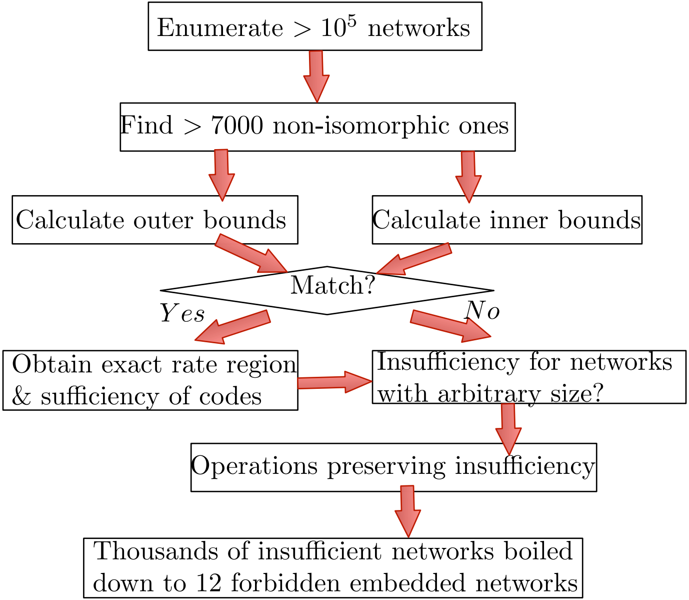


Example: Forbidden embedded networks

- Goal: minimal forbidden networks for sufficiency
- Scalar binary codes considered
- $k = 1, 2, 3; |\mathcal{E}| = 2, 3, 4$, 7360 non-isomorphic MDCS
- 1922 sufficient, 5438 insufficient
- 12 forbidden embedded networks (Li, *et. al* submission TransIT 2014)
- Open questions: complete list? Finite number for $\mathbb{F}_q$?



Summary of work thus far



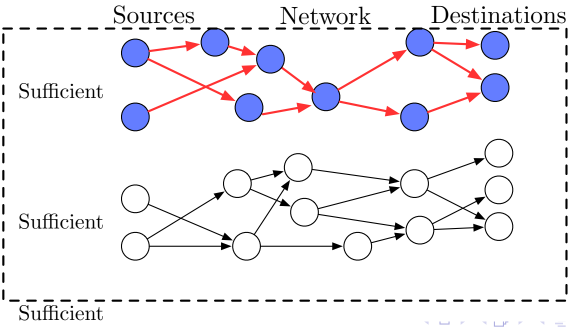
Publication List

1. Steven Weber, Congduan Li, and John Walsh, Rate Region for a class of delay mitigatting codes and P2P networks, 46th Annual Conference on Information Sciences and Systems, Princeton University, March, 2012, Invited Paper.
2. Congduan Li, John Walsh, and Steven Weber, "A Computational Approach for Determining Rate Regions and Codes using Entropic Vector Bounds", 50th Allerton Conference, UIUC, Oct 2012.
3. Congduan Li, Jayant Apte, John M. Walsh, and Steven Weber, "A new computational approach for determining rate regions and optimal codes for coded networks", The 2013 IEEE International Symposium on Network Coding (NetCod 2013), Calgary, Canada, Jun 7-9, 2013.
4. Congduan Li, John M. Walsh, and Steven Weber, "Matroid bounds on region of entropic vectors", 51th Allerton conference on Communication, Control and Computation, UIUC, IL, Oct 2013.
5. Jayant Apte, Congduan Li, John MacLaren Walsh, and Steven Weber, Exact pepair problems with multiple sources, in The 48th annual IEEE Conference on Information Sciences and Systems(CISS), Mar. 2014.
6. Jayant Apte, Congduan Li, and John MacLaren Walsh, Algorithms for Computing Network Coding Rate Regions via Single Element Extensions of Matroids, IEEE International Symposium on Information Theory (ISIT) 2014, Honolulu, Hawaii.
7. Congduan Li, Steven Weber, and John M. Walsh, "Network Embedding Operations Preserving the Insufficiency of Linear Network Codes", 52th Allerton conference on Communication, Control and Computation, UIUC, IL, Oct 2014.
8. Congduan Li, Steven Weber, and John M. Walsh, "Multilevel Diversity Coding Systems: Rate Regions, Codes, Computation, & Forbidden Minors", IEEE Transactions on Information Theory, 2014, SUBMITTED.

Outline

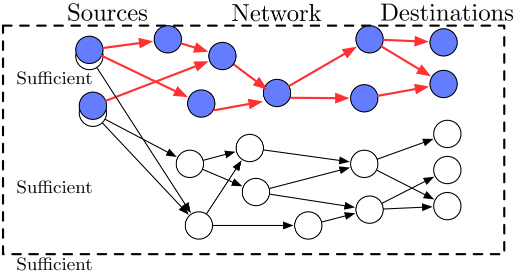
1: network combination operations

- Recall: embedding operations (tools) to certify insufficiency of class of codes in arbitrary size networks but not sufficiency (no tools)
- Extension: certify sufficiency of class of codes
- New operations: combine smaller networks into a bigger network
- Preserving: sufficiency of class of codes
- Existence: concatenation of two independent sufficient networks



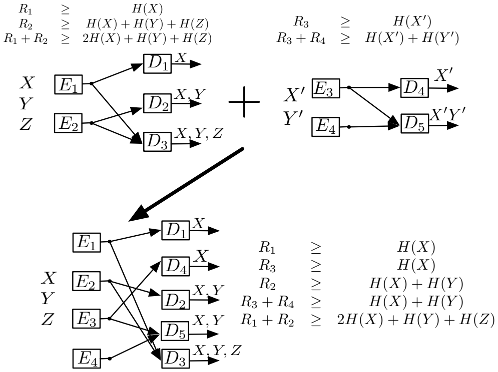
1: network combination operations

- Objectives: other operations preserving sufficiency
- Source Merge?



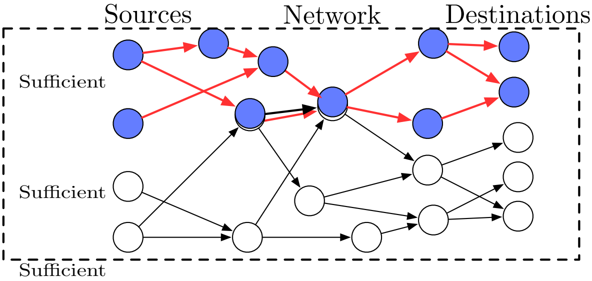
1: network combination operations

- Example: Source merge in MDCS
- Scalar binary sufficiency and tightness of Shannon outer bound are preserved



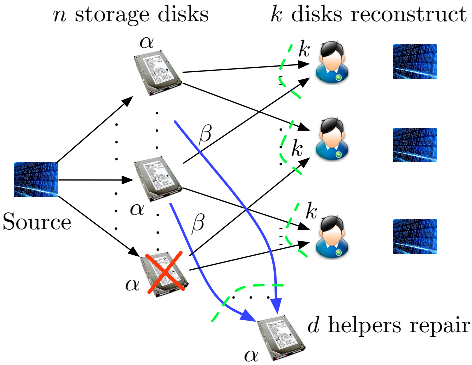
1: network combination operations

- Objectives: other operations preserving sufficiency
- Edge Merge? Intermediate nodes merge?



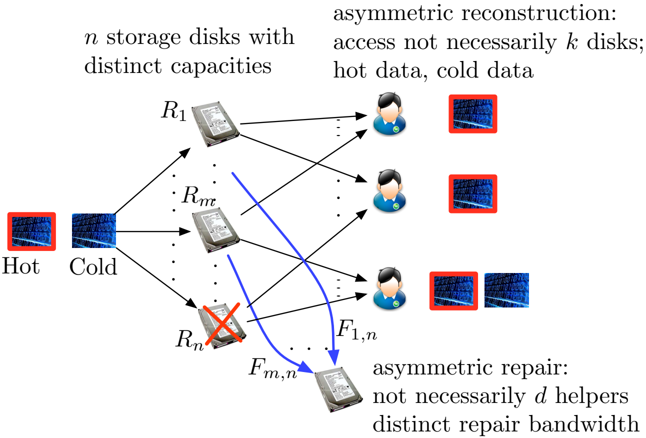
2: asymmetric distributed storage exact repair

- Computation tools work for general networks, will try storage exact repair problems
- Common setup: symmetric, (n, k, d, α, β)



Task 2: asymmetric distributed storage exact repair

- Will investigate asymmetric setups with prioritized data
- Initial work with Jayant Apte [6, CISS 2014]: symmetric in disks and repair, asymmetric in reconstruction



3: polymatroid extremality

- Recall: computation cares exclusively about matroids that are extreme rays,
- Connected matroids are extremal: $r(\mathcal{A}) + r(\mathcal{E} \setminus \mathcal{A}) > r(\mathcal{E})$, $\forall \mathcal{A} \subsetneq \mathcal{E}$
- Extremal matroids are extremal polymatroids
- Connected polymatroid $=$ extremal polymatroids?

3: polymatroid extremality

- Extremal polymatroids $\Rightarrow$ Connected: otherwise separable, sum of two polymatroids
- Connected polymatroid $\not\Rightarrow$ extremal polymatroids
- Investigate the sufficient condition for polymatroid extremality
- Potential reduction in computation complexity

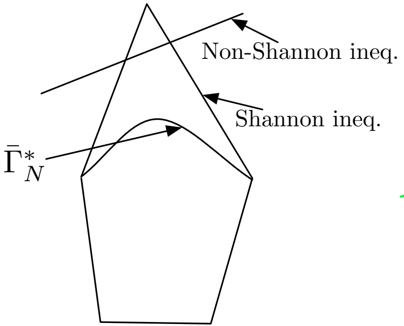
$$[H(X)\ H(Y)\ H(XY)\ H(Z)\ H(XZ)\ H(YZ)\ H(XYZ)]$$

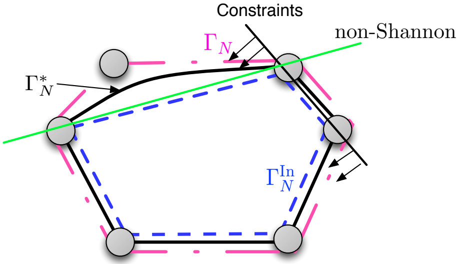
$$[2\ 2\ 4\ 2\ 4\ 4\ 5] = [1\ 1\ 2\ 1\ 2\ 2\ 2] + [1\ 1\ 2\ 1\ 2\ 2\ 3]$$

Connected but Not extremal — Not connected — Connected

4: necessity of non-Shannon inequalities

- Tighter outer bound on $\bar{\Gamma}_N^*$: Shannon + non-Shannon
- Non-Shannon to close the gap?
- Investigate necessity of non-Shannon: try to construct distributed storage exact repair network associated with Vámos matroid
- Preserving property: if a network requires non-Shannon, what about its extensions?





References

X. Yan, R.W. Yeung, and Z. Zhang (2012)
An Implicit Characterization of the Achievable Rate Region for Acyclic Multisource Multisink Network Coding
IEEE Transactions on Information Theory 58(9), 5625-5639.

R.W. Yeung, R. S.Y. Li, N. Cai, Z. Zhang (2006)
Network Coding Theory
now publishers Inc

C. Tian (2013)
Rate region of the (4, 3, 3) exact-repair regenerating codes
IEEE International Symposium on Information Theory (ISIT), 1426-1430.

Thank you!