Network Coding: An algorithmic perspective

T. Ho* and A. Sprintson**

*California Institute of Technology

**Texas A&M University

DIMACS workshop

Outline

Coding advantage

Connection to the Integrality Gap

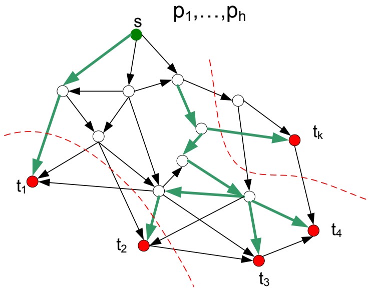
- We show that for undirected networks, the maximum coding advantage is equal to the integrality gap of the bi-directed cut relaxation for the undirected Steiner tree problem.
- We show results by Agarwal and Charikar'04

Integrality gap

- Many problems can be formulated as integer problems
- However, many such problems are NP-hard
- Let OPT be the optimal solution to the integer program P
  - We refer to is as an optimal integer solution.
- Let OPT^* be the optimal solution to the linear relaxation of P.
  - We refer to is as an optimal linear solution.
- Then, the integrality gap is equal to the ratio $\frac{OPT}{OPT*}$.

The minimum weight Steiner Tree problem

- Given:
  - Undirected graph $G = (V, E)$, $w : E \rightarrow R^+$ be an assignment of non-negative weights to the edges, a source node, a set of destination nodes
- Find: A minimum weight tree that connects s to T. We denote the weight of this tree by $OPT(G, w)$



Uni-directed cut relaxation

- We say that a set $C \subseteq V$ that contains the source node s and $V \setminus C$ contains at least one terminal $t \in T$ is a valid set.
- Denote $\delta(C) = \{(u, v) \in E \mid u \in C, v \notin C\}$

Linear Program

$$\textit{Minimize} \sum_{e \in E} w_e c_e$$

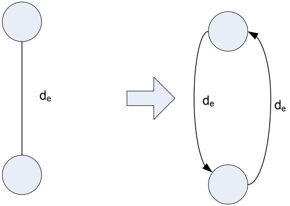
Subject to

$$\begin{aligned} \sum_{e \in \delta(C)} c_e \geq 1, \quad & \forall \text{ valid sets C} \\ c_e \geq 0, \quad & \forall e \in E \end{aligned} \tag{1}$$

Bi-directed Cut Relaxation

- For each edge $e \in E$ we introduce two directed arcs e_1 and e_2, which represent the two orientations of e
- Edges e_1 and e_2 have the same weight as e
- We denote by $D = \{e_1, e_2, \forall e \in E\}$
- Same definition of a valid set C.
- We denote

$$\delta(C) = \{(u, v) \in D \mid u \in C, v \notin C\}$$



Bi-directed Cut Relaxation (cont.)

Integer Program

$$\textit{Minimize} \sum_{a \in D} w_a c_a$$

Subject to

$$\begin{aligned} \sum_{a \in \delta(C)} c_a &\geq 1, \quad \forall \text{ valid sets C} \\ c_a &\in \{0, 1\}, \quad \forall a \in D \end{aligned} \tag{2}$$

Linear Relaxation

- Replace (relax) The last constant.
- Denote by $B(G, w)$ the cost of the resulting problem
- Linearity gap is equal to

$$\max_{G,w} \frac{OPT(G, w)}{B(G, w)}$$

Linear Program

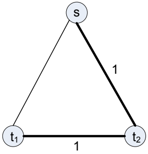
$$\text{Minimize} \sum_{a \in D} w_a c_a$$

Subject to

$$\begin{aligned} \sum_{a \in \delta(C)} c_a \geq 1, &\quad \forall \text{ Valid sets C} \\ c_a \geq 0, &\quad \forall a \in D \end{aligned} \tag{3}$$

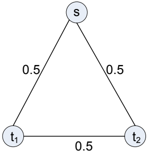
Example

Optimal Undirected solution



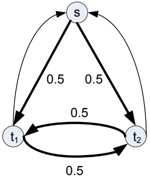
cost=2

Undirected linear relaxation



cost=1.5

Bi-directed linear relaxation



Finding network code through Integer Program (cFlow)

- Given: An Undirected graph $G = (V, E)$.
- Let $c : E \to R^+$ be an assignment of non-negative capacities to the edges
- Denote by c_e the capacity of edge $e \in E$
- For given edge capacities, we need to find a maximum throughput between s and $t_1, \cdots, t_k$
- Denote the set of directed arcs $D = \{e_1, e_2 \mid \forall e \in E\}$.
- The value of the optimal solution is denoted by $\chi(G, c)$

Linear Program

Maximize f^*

Subject to

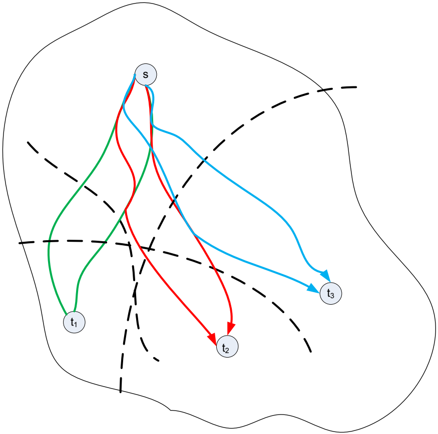
- $c_a > 0$ *for each* $a \in D$
- $c_{e_1} + c_{e_2} = c_e$ *for each* $e \in E$
- $f^n(a) \leq c_a$ *for each* $a \in D$ *and* $n, 1 \leq n \leq k$
- *for each* $n, 1 \leq n \leq k$ *and for each* $v \in V \setminus \{s, t_n\}$

$$\sum_{v_j:(v_i,v_j)\in D} f^n_{(v_i,v_j)} - \sum_{v_j:(v_j,v_i)\in D} f^n_{(v_j,v_i)} = 0$$

- $\sum_{v_j:(v_j,s)\in D} f^n_{(v_j,v_s)} = 0$ *for* $n, 1 \leq n \leq k$
- $\sum_{v_j:(t_n,v_j)\in D} f^n_{(t_n,v_j)} = 0$ *for* $n, 1 \leq n \leq k$
- $\sum_{v_j:(v_j,t_n)\in D} f^n_{(v_j,t_n)} \geq f$ *for* $n, 1 \leq n \leq k$

Example

- Observation: Linear relaxation of bidirectional formulation is similar to the network coding problem



Steiner Tree packing

- Let τ be a set of all possible Steiner trees that connect s and T
- We define a variable x_t for any possible Steiner tree $t \in \tau$
  - x_t captures the amount of information transmitted by t.
- We denote by $\Pi(G, c)$ the optimal solution for the problem
- The related linear program is:

Linear Program

$$\textit{Maximize} \sum_{t \in \tau} x_t$$

Subject to

$$\begin{aligned} \sum_{t \in \tau : e \in t} x_t \leq c_e, & \quad \forall e \in E \\ x_t \geq 0, & \quad \forall t \in \tau \end{aligned} \tag{4}$$

The Dual for Steiner Tree Packing

- Introduce a local variable y_e for each arc $e \in E$
- The dual program can be formulated as follows:

Linear Program

$$Minimize \sum_{e \in E} c_e y_e$$

Subject to

$$\begin{aligned} \sum_{e \in t} y_e \geq 1, \quad & \forall t \in \tau \\ y_e \geq 0, \quad & \forall e \in E \end{aligned} \qquad (5)$$

Linear Program

$$\textit{Maximize} \sum_{t \in \tau} x_t$$

Subject to

$$\begin{aligned} \sum_{t \in \tau : e \in t} x_t \leq c_e, & \quad \forall e \in E \\ x_t \geq 0, & \quad \forall t \in \tau \end{aligned} \tag{6}$$

Linear Program

$$\textit{Minimize} \sum_{e \in E} c_e y_e$$

Subject to

$$\begin{aligned} \sum_{e \in t} y_e \geq 1, & \quad \forall t \in \tau \\ y_e \geq 0, & \quad \forall e \in E \end{aligned} \tag{7}$$

Theorem

- $\chi(G, c)$ - the maximum throughput with network coding
- $\Pi(G, c)$ - the maximum throughput with Steiner tree packing
- $OPT(G, w)$ - minimum weight of a Steiner tree
- $B(G, w)$ - the optimal value for the bi-directed cut relaxation.

Theorem

$$\max_c \frac{\chi(G, c)}{\Pi(G, c)} \leq \max_w \frac{OPT(G, w)}{B(G, w)}$$

Proof

- Note that the coding advantage is invariant under multiplicative scaling of capacities.
- Scale the capacities so that the value of the objective function for the cFlow LP is equal to 1, i.e., $\chi(G, c) = 1$
- Consider the dual to the Steiner packing LP - this program gives an example of the integrality gap

Linear Program

$$\textit{Minimize} \sum_{e \in E} c_e y_e$$

Subject to

$$\begin{aligned} \sum_{e \in t} y_e \geq 1, &\quad \forall t \in \tau \\ y_e \geq 0, &\quad \forall e \in E \end{aligned} \tag{8}$$

Proof (cont.)

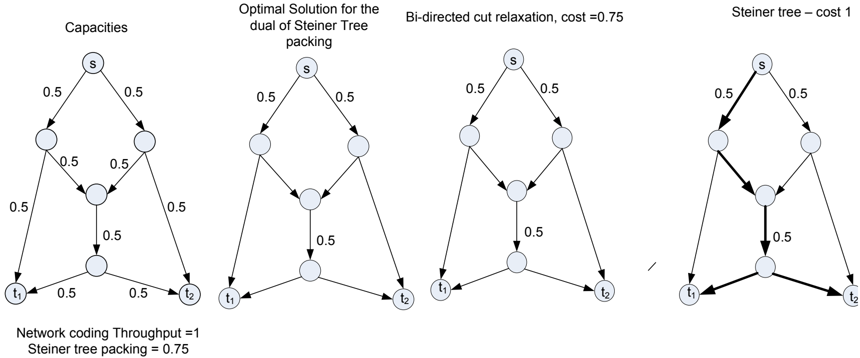
- By strong duality, the optimal value of the dual LP is equal to $\Pi(G, c)$.
- We can view the y_e's as edge costs
- The constraint implies that the cost of every integer Steiner tree is at least one.
- We claim that $\sum_{e \in E} c_e y_e$ is an upper bound on the value of $B(G, y)$
  - The graph with capacities c_e has a cFlow value of at least 1.
  - These capacities give a valid solution to the bi-directed graph relaxation problem.

Proof (cont.)

- We conclude that there exists a setting of weights w such that:
  - $OPT(G, w) \geq 1$
  - $B(G, w) \leq \Pi(G, c)$
- Weights w are assigned according to the solution to the dual problem of Steiner tree packing
- We conclude that

$$\max_c \frac{\chi(G, c)}{\Pi(G, c)} \leq \max_e \frac{OPT(G, w)}{B(G, w)}$$

Example


Capacities
Optimal Solution for the dual of Steiner Tree packing
Bi-directed cut relaxation, cost =0.75
Steiner tree – cost 1
s
0.5
t1
t2
Network coding Throughput =1
Steiner tree packing = 0.75

Conclusion

- We proved that:

Theorem

(Agarwal and Charikar'04)

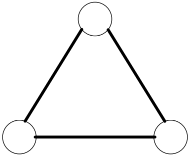
$$\max_c \frac{\chi(G,c)}{\pi(G,c)} = \max_w \frac{OPT(G,w)}{B(G,w)}$$

- The theorem shows that
  - The coding advantage is equal to the integrality gap of the bi-directional relaxation
  - The throughput advantage is equal to the cost advantage

Undirected networks

Coding advantage in undirected networks

- Introduce new notation - $\lambda(N)$
- $\lambda(N)$- the minimum edge connectivity between a pair of nodes in S
  - For two nodes $v \in V, u \in V$, the minimum edge connectivity is the minimum size of a cut that separates v and u
  - Maximum number of disjoint paths that separate v and u.

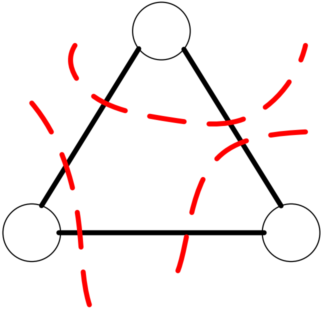


Strength of the network

Definition

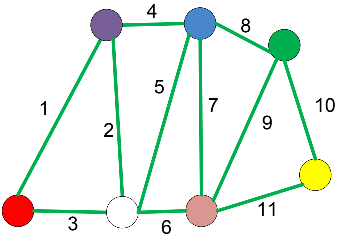
- Define $\eta(N)$ to be the minimum ratio of $\frac{E_c}{p-1}$, where
  - p is the number of components the communication group is separated into
  - E_c is the set of the inter-component links
  - Each partition is required to have at least one source or destination node
- $\eta(N)$ is referred to as the strength of the network.

Example



Example (cont.)

- $\eta(N) \leq \frac{11}{6}$



7 components
11 edges

Tutte-Nash-Williams Theorem

Theorem

A graph G has x pairwise edge-disjoint spanning trees if and only if,

- *For every vertex partition*
  - *There are at least $(p-1) \cdot x$ edges with endpoints in different components*
  - *where p is the number of components in the partition*

- The theorem characterizes the the maximum throughput with spanning tree packing

Corollary

- In the broadcast setting a Steiner tree is a spanning tree.
- We denote by $\pi(N)$ the packing number of the coding network
  - Equal to the performance we can achieve without coding0

Corollary

- *For integral spanning tree packing problem it holds that*

$$\pi(N) = \lfloor \eta(N) \rfloor$$

- *For fractional spanning tree packing problem it holds that*

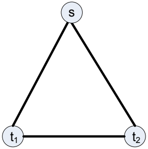
$$\pi(N) = \eta(N)$$

Nash-Williams' Weak Graph Orientation Theorem

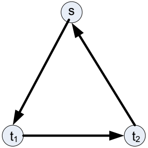
Theorem

A graph G has an x-connected orientation if and only if it is $2x$-edge connected.

Example 1

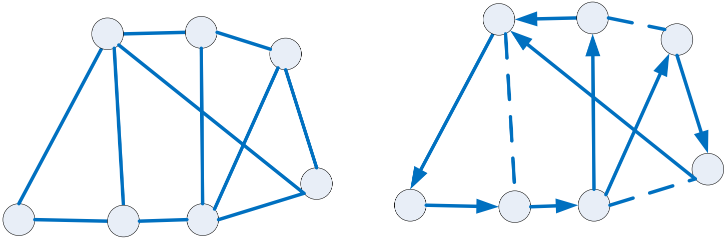


The network is two-connected



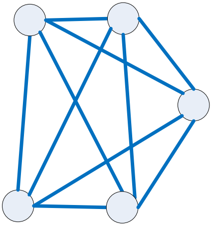
1-edge connected directed orientation

Example 2

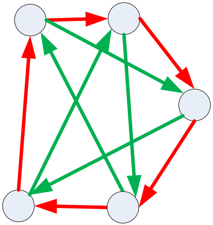


Two-connected graph One-connected orientation

Example 3



Four-connected graph



Two-connected orientation

Theorem 19-1

- $\chi(N)$ - the maximum throughput with network coding
- $\pi(N)$ - the maximum throughput with Steiner tree packing
- $\lambda(N)$ - the minimum edge connect. between a pair of nodes in S
- $\eta(N)$ - the strength of the network

Theorem (19-1)

For a broadcast transmission ($S = V$) in an undirected network N it holds that

$$\frac{1}{2}\lambda(N) \leq \pi(N) = \chi(N) = \eta(N) \leq \lambda(N)$$

- The theorem shows that there is no coding advantage for broadcast connections (one-to-all)

Proof

Proof.

- Tutte-Nash-Williams Theorem implies that

$$\pi(N) = \eta(N)$$

- The definition of $\chi(N)$ implies that

$$\pi(N) \leq \chi(N)$$

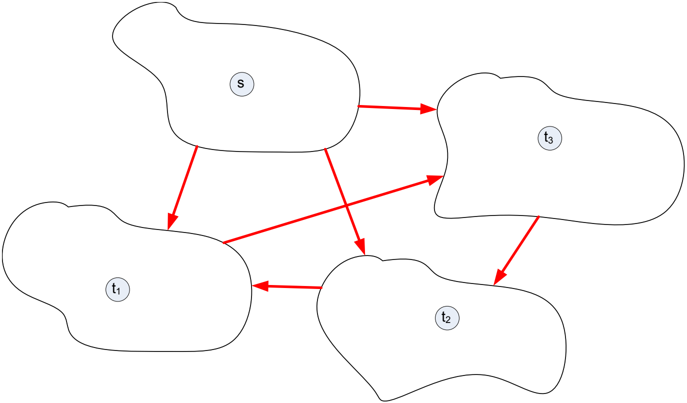
- Each component not including the source node needs a total edge capacity of x in order to achieve throughput x, hence

$$\chi(N) \leq \eta(N)$$

- Hence

$$\pi(N) = \chi(N) = \eta(N)$$

Example



4 components – number of arcs at least 6

Proof (cont)

Proof.

- From the definition of $\eta(N)$ it holds that

$$\eta(N) \leq \lambda(N)$$

- The Nash-Williams' weak orientation theorem implies that a network N always has a $\frac{1}{2}\lambda(N)$-directed orientation
  - orientation in which we can send $\frac{1}{2}\lambda(N)$ units of flow between any pair of terminals
  - assuming the fractional setting
- Combining with the result from network coding in directed networks we conclude that

$$\frac{1}{2}\lambda(N) \leq \chi(N)$$

Theorem 19-2

- $\chi(N)$ - the maximum throughput with network coding
- $\pi(N)$ - the maximum throughput with Steiner tree packing
- $\lambda(N)$ - the minimum edge connect. between a pair of nodes in S
- $\eta(N)$ - the strength of the network

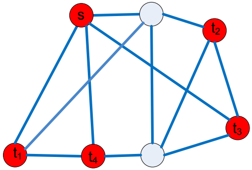
Theorem

For a multicast transmission in an undirected network N it holds that

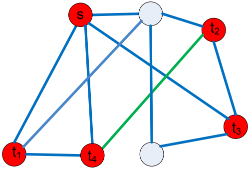
$$\frac{1}{2}\lambda(N) \leq \pi(N) \leq \chi(N) \leq \eta(N) \leq \lambda(N)$$

- The theorem implies that the coding advantage is bounded by 2

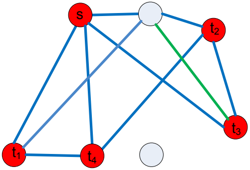
Example



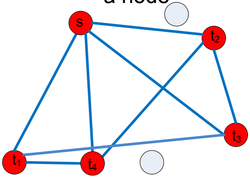
(a)Local connetivity -3



(b) After splitting first two edges

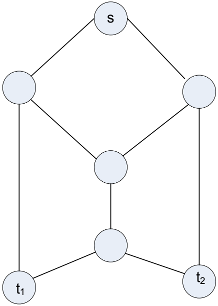


(c) After splitting off a node

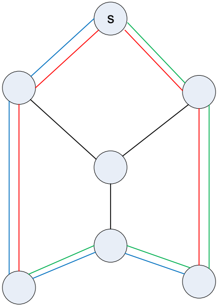


(d) After splitting off all non-terminal nodes

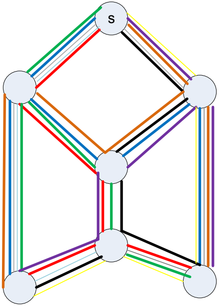
Example



Network coding 2



Half-integer routing 1.5

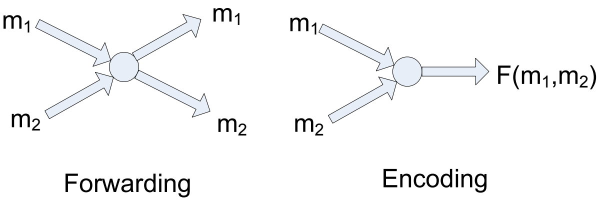


Fractional routing 1.875
Thin trees -capacity 0.125
Thick trees – capacity 0.25

Encoding Complexity of Network Coding

Encoding Complexity of Network Coding

- Traditional approach - nodes only forward or duplicate information
- Network coding - requires encoding capability at certain nodes
- Network coding are more expensive - need additional functionality
- Important question- how many encoding nodes are needed to achieve capacity?



Bounds - Directed Acyclic Graphs

Theorem

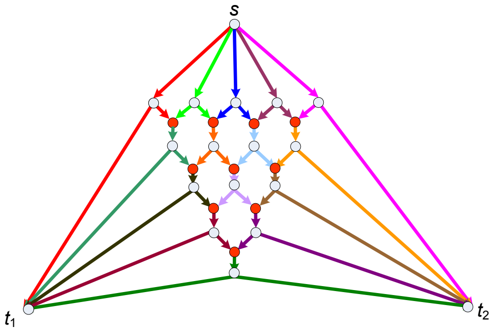
An acyclic coding network with h packets and k destinations requires at most $O(h^3 k^2)$ encoding nodes

Theorem

There exists an acyclic coding network with h packets and k destinations that requires at least $\Omega(h^2 k)$ encoding nodes.

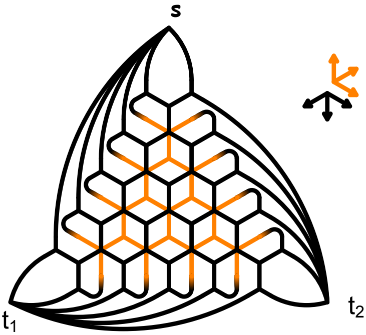
Bounds - Directed Acyclic Graphs

- A $\frac{h^2}{2}$ bound



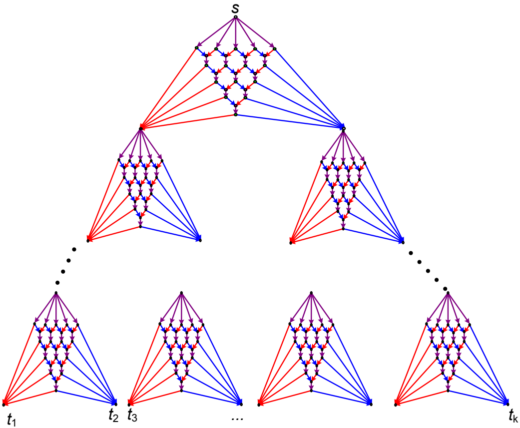
Bounds - Directed Acyclic Graphs

- A h^2 bound



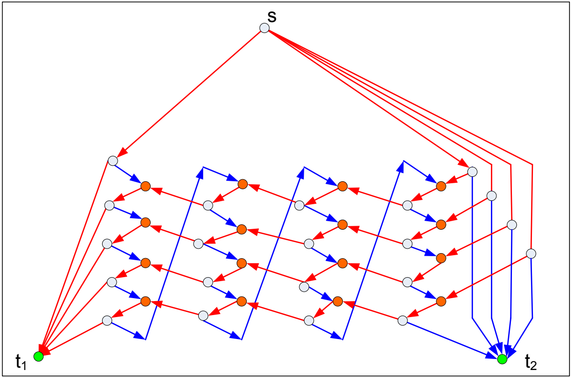
Bounds - Directed Acyclic Graphs

- A h^2k bound



Bounds - Networks with Cycles

- Number of encoding nodes does not depend on *h* and *k*
- Can be bounded by the size of Minimum feedback set



Summary

	Directed Acyclic	Undirected	Directed Cyclic
Upper bound	h^3k^2	$O(h^3k^2)$	$(2B+1)\ h^3k^2$
Lower bound	$\Omega(h^2k)$	$\Omega(h^2k)$	$(h-1)B$ $\lvert V\rvert/2$

Instantaneous Recovery from Link Failures

Network Model

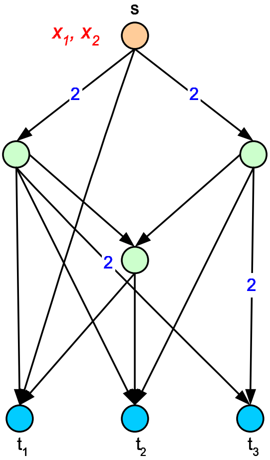


Figure: Example of a multicast network.

- Network represented by a directed graph $G(V, E)$
- Capacity function $c(e) : E \rightarrow \mathbb{N}$
- Source s needs to send h packets $x_1, \ldots, x_h$
  - to a single destination node t (unicast)
  - to a set of destination nodes $t_1, \ldots, t_k$ (multicast)
- Edges are susceptible to failures

Failure model

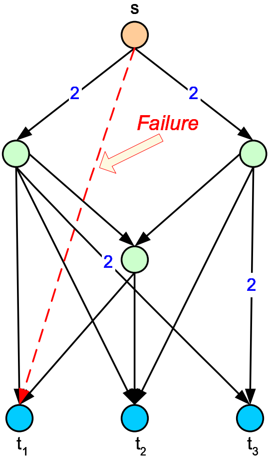


Figure: Failed edge in a network

- A failed edge is deleted from the network

Goal

Find a communication scheme that will guarantee the delivery of all the packets to the destination(s) in the case of any single edge failure.

Standard Approach

- Rerouting upon a failure

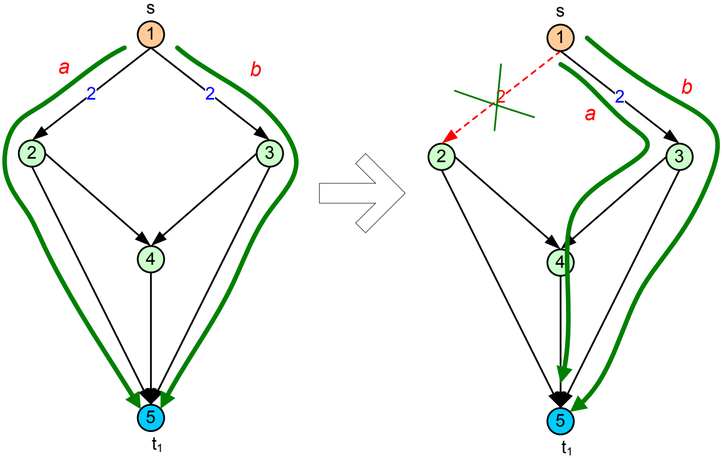
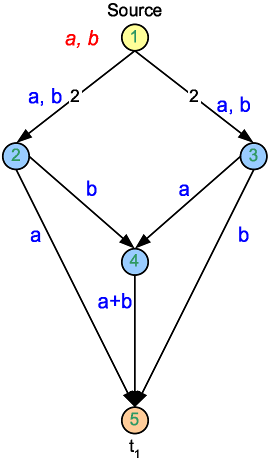


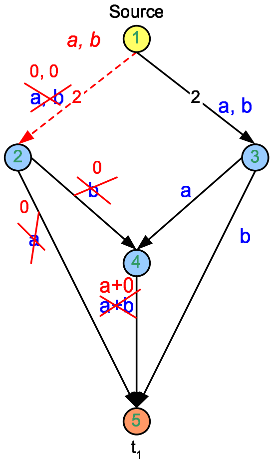
Figure: A feasible unicast network.

Restoration with Network Coding



- *a* and *b* are bits
- "+" is the bitwise XOR operation
- the packet carried by a failed edge is always zero
- v_4 is an encoding node

Restoration with Network Coding

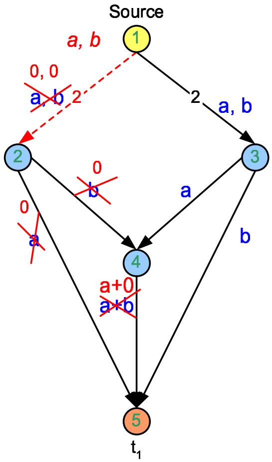


- Single edge failure
- The destination will always be able to decode the original packets *a* and *b*

failure of	m_{25}	m_{45}	m_{35}
ϕ	a	$a+b$	b
(v_1, v_2)	0	a	b
(v_1, v_3)	a	b	0
(v_2, v_4)	a	a	b
(v_3, v_4)	a	b	b
(v_2, v_5)	0	$a+b$	b
(v_3, v_5)	a	0	b
(v_4, v_5)	a	$a+b$	0

- Instantaneous recovery!

Restoration with Network Coding

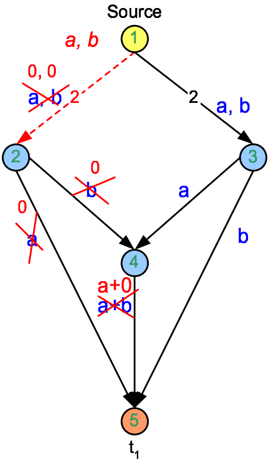


- Single edge failure
- The destination will always be able to decode the original packets a and b

failure of	m_{25}	m_{45}	m_{35}
ϕ	a	$a+b$	b
(v_1, v_2)	0	a	b
(v_1, v_3)	a	b	0
(v_2, v_4)	a	a	b
(v_3, v_4)	a	b	b
(v_2, v_5)	0	$a+b$	b
(v_3, v_5)	a	0	b
(v_4, v_5)	a	$a+b$	0

- Instantaneous recovery!

Restoration with Network Coding



- Single edge failure
- The destination will always be able to decode the original packets *a* and *b*

failure of	m_{25}	m_{45}	m_{35}
ϕ	a	$a+b$	b
(v_1, v_2)	0	a	b
(v_1, v_3)	a	b	0
(v_2, v_4)	a	a	b
(v_3, v_4)	a	b	b
(v_2, v_5)	0	$a+b$	b
(v_3, v_5)	a	0	b
(v_4, v_5)	a	$a+b$	0

- Instantaneous recovery!

Robust Network Codes

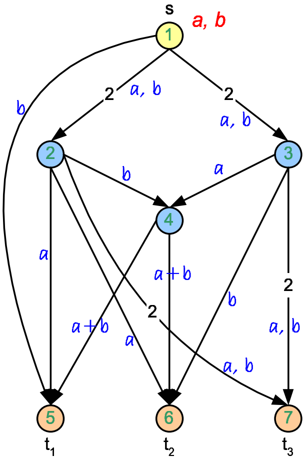


Figure: Example of a robust network code

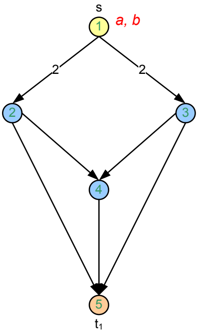
Definition

A robust network code, for a multicast network, is a linear network code that, in the case of *a single edge failure* will guaranty

- the delivery of all the packets
- to all the destinations

Coding Advantage

- How many packets can be sent reliably from s_1 to s_5?
  - With instantaneous recovery
  - Without rerouting
- Only one with the traditional approach
- Two with the network coding approach



Resilient Capacity

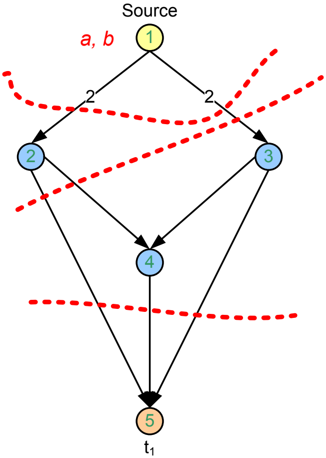
Definition

Resilient capacity C of a unicast network $G(V, E)$ is the maximum number of packets that can be sent reliably from s to t.

- Necessary condition:
  - For every $e \in E$, $G \setminus \{e\}$ must have C paths between s and t
- Cut condition:
  - For every cut $(S, V \setminus S)$ that separates s and t it must hold that

$$C \leq \sum_{e \in E(S, V \setminus S)} c(e) - \max_{e \in E(S, V \setminus S)} c(e)$$

Example



Achieving Capacity

- Resilient capacity can be achieved by using linear network codes
  - [Koetter and Medard'03]
- Robust network codes can be found in polynomial time, for both unicast and multicast
  - [Jaggi et al.03]
  - Require large field size $O(k|E|)$, where k is the number of terminals

Our goal

- For $h = 2$
  - Design robust codes of $GF(2)$
  - Design an efficient algorithm for computing efficient network codes
  - Introduce the concept of simple network
  - Show that simple networks have certain structure
  - Show that for multicast networks with k terminals, a field size $\geq 5k$ is sufficient.
- For $h > 2$
  - Show that it is possible to design a network code over a field size which is independent on the size of the network
  - Depends only on h

Minimal Networks

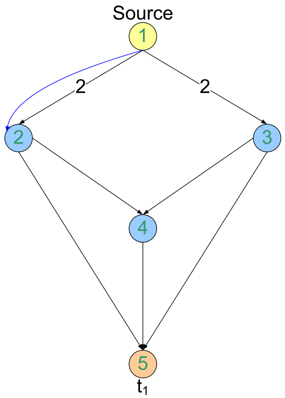


Figure: A non minimal unicast network

Definition

A multicast network is minimal if all its *subnetworks* , obtained by deleting an edge or reducing its capacity, are *not feasible*.

Definition

A unicast network ($h = 2$) is a simple network iff it is

- feasible
- minimal
- All of its nodes are of degree 3

Reduction to Simple Networks

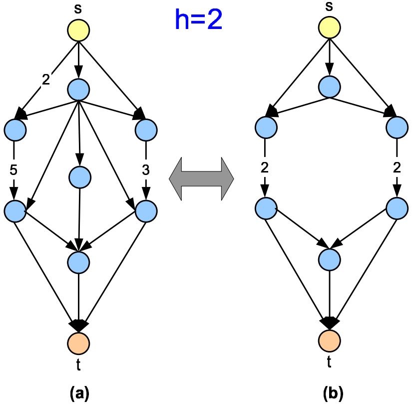


Figure: (a) unicast net. (b) corresponding simple net.

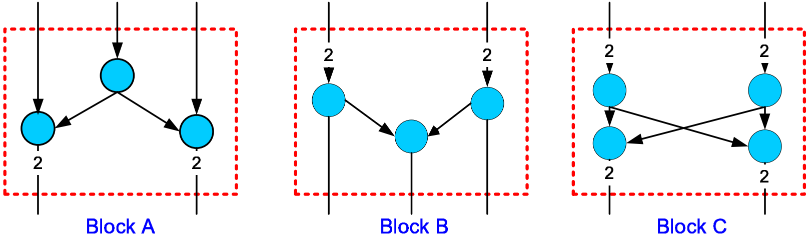
Theorem

Let $\mathbb{N}$ be a feasible unicast network ($h = 2$). Then, there exists a simple network $\mathbb{N}'$ such that if $\mathbb{N}'$ has a robust network code over $GF(q)$, then $\mathbb{N}$ has also one over the same field.

Structure of Simple Networks

Theorem

All simple networks $\mathbb{N}$ can be decomposed into the blocks A, B and C depicted below.



Example

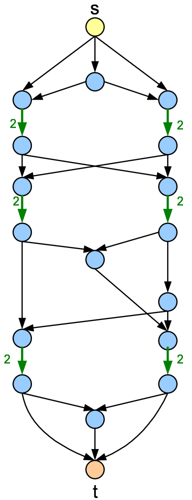


Figure: A simple network

Example

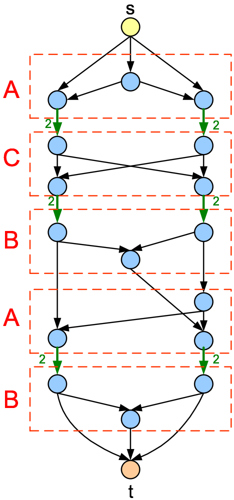
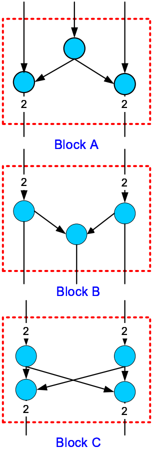


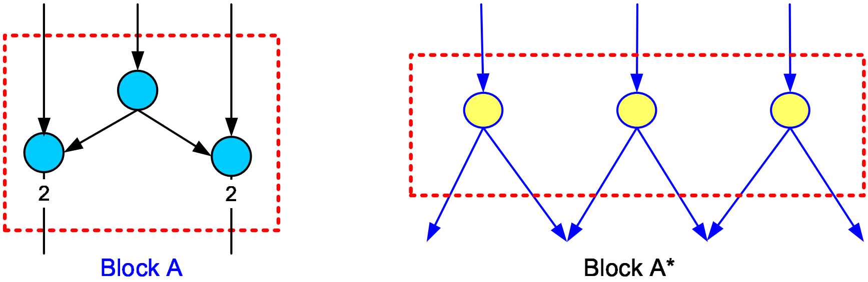
Figure: Block decomposition of a simple network

Sketch of Proof



- The proof of the *block decomposition theorem* of simple networks is based on
  - Residual networks
  - The augmenting cycle theorem
- Any configuration, other than blocks A, B and C
  - will result in a flow with some edge carrying a *zero* flow.
  - contradicts the minimality of the simple network

Proof: Example



Proof: Example

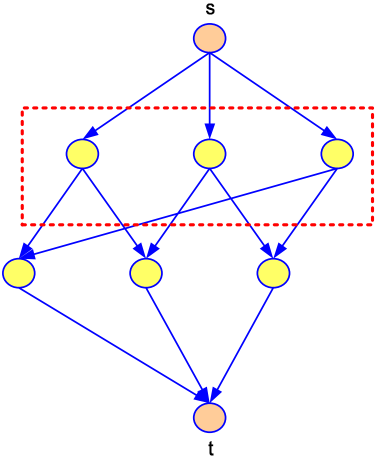
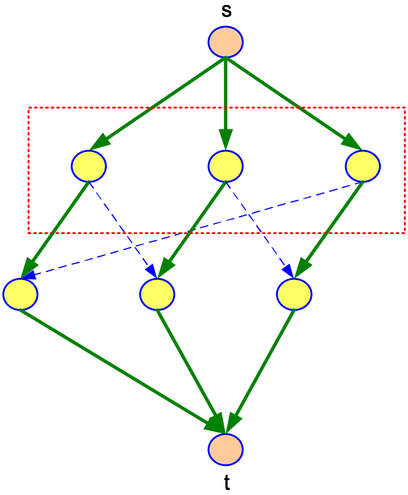


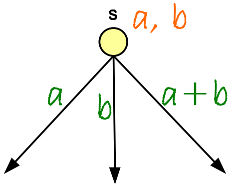
Figure: A network containing block A^*

Proof: Example

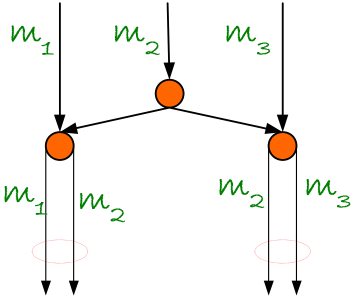


- This network has a flow of value 3 where some edges carry a flow zero $\Rightarrow$ The original network is not minimal
- One can show that all network containing bloc A^* are not minimal

Robust Network Code

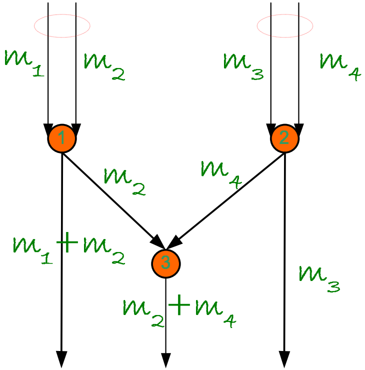


Source



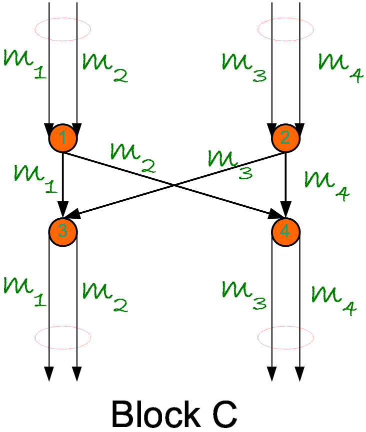
Block A

Robust Network Code



Block B

Robust Network Code



Proof of Robustness

- A simple network always ends by a block B
- The proof of the robustness of the network code is done by induction on the number of blocks B in the network
  - we show that the output of any block B is always a subset of at least two elements of the set $\{a, b, a + b\}$

Beyond Unicast

Theorem

Consider a multicast network $\mathbb{N}$ with $h = 2$ packets and k destinations. Then, there exists a robust network code for $\mathbb{N}$ over $GF(q)$ for all $q \geq 5k$.

Lemma (Jaggi et al. 04)

If m flows are needed to protect against all single edge failures in $\mathbb{N}$, then there exists a robust network code $\mathbb{N}$ over $GF(q)$ for all $q \geq m$

Beyond Unicast

Theorem

Consider a multicast network $\mathbb{N}$ with $h = 2$ packets and k destinations. Then, there exists a robust network code for $\mathbb{N}$ over $GF(q)$ for all $q \geq 5k$.

Lemma (Jaggi et al. 04)

If m flows are needed to protect against all single edge failures in $\mathbb{N}$, then there exists a robust network code $\mathbb{N}$ over $GF(q)$ for all $q \geq m$

Practical Implementation

Practical Implementation

- Need to address many practical issues:
  - Delays, packet losses
  - Unknown link capacities
  - No centralized knowledge of network topology
  - Frequent network changes, e.g., due to link failures

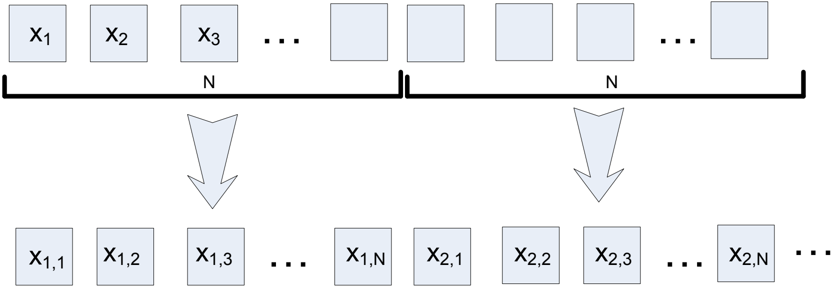
Practical Implementation

- Chou et. al. 2004
  - Content distribution network, $|E| \leq 256$
  - Field size 2^{16}
  - Use random network coding
  - Each matrix has full rank with probability at least 0.996
  - Maximum packet size in the Internet 1400 bytes
  - Each IP packet can carry about 700 symbols

Practical Implementation

- Idea: packetize the source symbols x_i flowing into the sender into vectors

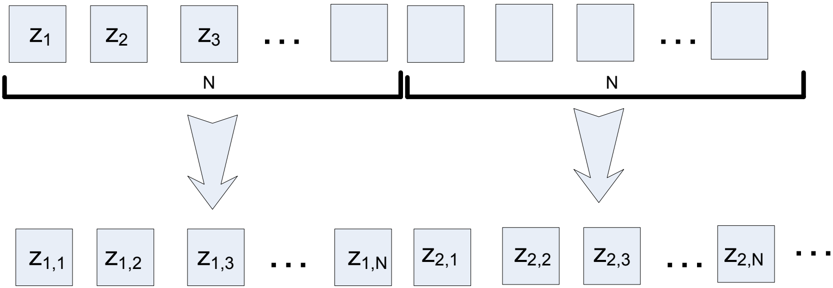
$$x_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,N}]$$



Practical Implementation

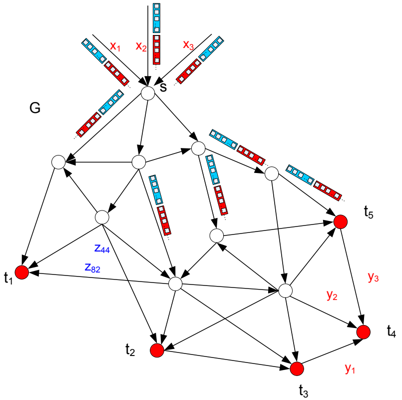
- Similarly, the symbols flowing on each link are also packetized.

$$z_i = [z_{i,1}, z_{i,2}, \ldots, z_{i,N}]$$



Practical Implementation

- The same code is applied to all symbols in the packet



Practical Implementation

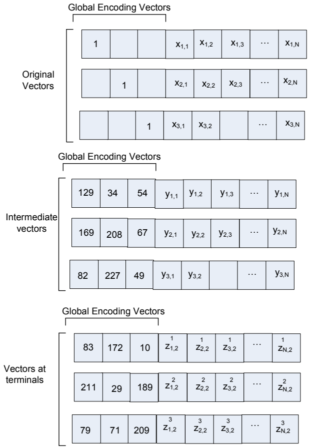
- Include within each packet the corresponding global encoding vector

$$y(e) = \sum_{i=1}^{h} g_i(e)x_i$$

- Implementation: Can be accomplished by prefixing a unit vector to each source vector x_i, $i = 1, \ldots, h$.
- The global encoding vector required for decoding can be found in the received packets

Practical Implementation

- Global encoding vectors are attached to each packet



Practical Implementation

- Decoding is possible even if
  - Network topology & encoding functions unknown
  - Topology dynamically changes
  - Packets lost, link failures in unknown locations
  - Local encoding vectors are time-varying
- Cost: (small) overhead

Encoding overhead

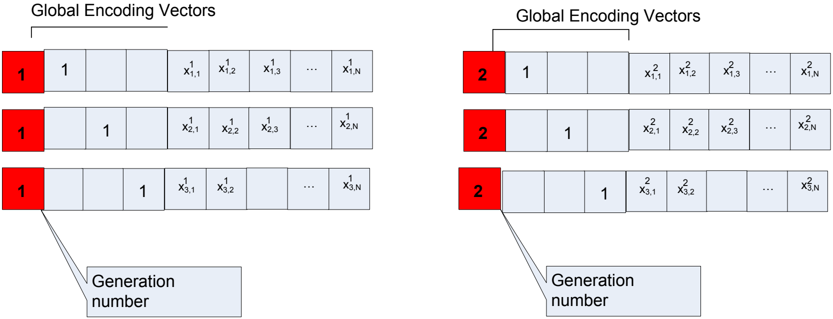
- Example 1
  - $h = 50$, filed size 2^{16}
  - Overhead $\frac{50}{700} \approx 6\%$
- Example 2
  - $h = 50$, field size 2^8 (sufficient for most practical applications)
  - Overhead $\frac{50}{1400} \approx 3\%$

Generations

- Introduced to improve robustness, eliminate redundancy, and deal with synchronization issues
- All packets related to same source vectors $x_1, \ldots, x_h$ are in the same generation
- All packets in same generation are tagged with same generation number; one byte (mod 256) is sufficient for practical purposes

Generations

- Each generation has exactly *h* packets

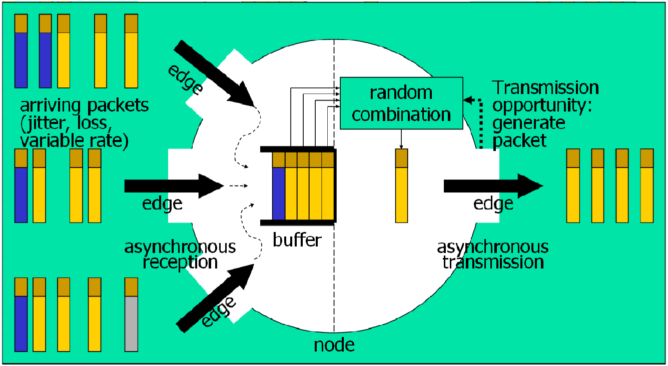


Dealing with Redundant Packets

- Packets received by a buffer are classified into two categories:
  - Innovative packets- lie outside the subspace spanned by vectors already in the buffer
  - Non-innovative packets - lie inside the subspace
    - Do not introduce new information, hence they are discarded

Router implementation

- Picture courtesy Chou et al. ’04

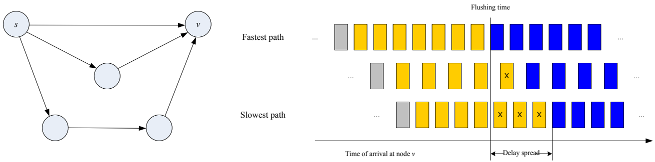


Decoding

- Block decoding
  - Collect h or more packets, try to invert G_t
- Earliest decoding
  - Perform Gaussian elimination at after each packet
  - G_i tends to be lower triangular, so can decode $x_1, \ldots, x_k$ with k packets
  - Lower decoding delay than the block decoding

Flashing

- Simple policy: flush when first packet of next generation arrives on any edge
- May result in a small loss of throughput
- Picture courtesy Chou et al. '04



Conclusion

- New research area
  - Requires tools from different disciplines:
    - Algebra, graph theory, combinatorics.
- Rich in challenging problems, many of them open
  - Multicast problems are well-understood
  - Beyond multicast -problems are very hard, very little is known.
- Both theoretical and practical interest
  - Many applications are yet to be discovered