A note on the Survivable Network Design Problem

Chandra Chekuri

Univ of Illinois, Urbana-Champaign

Joint work with Thapanapong Rukkanchanunt

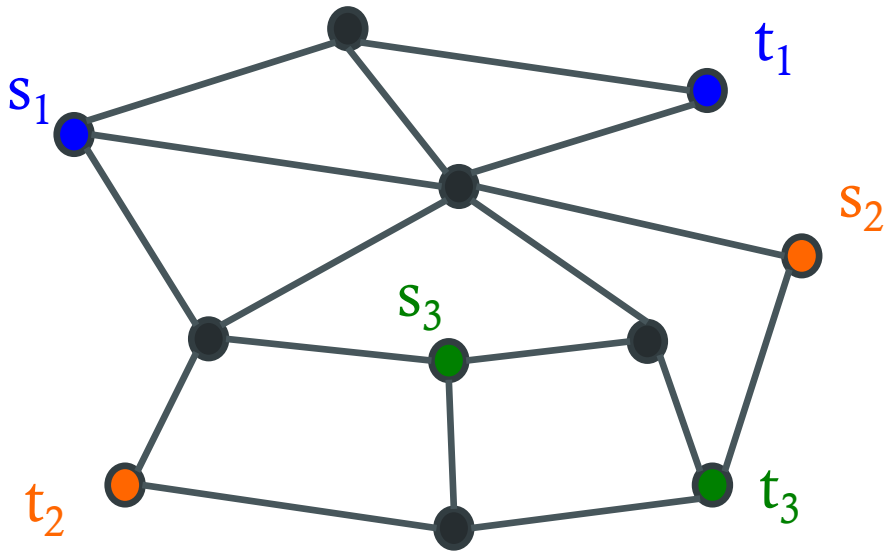
Survivable Network Design Problem (SNDP)

Input:

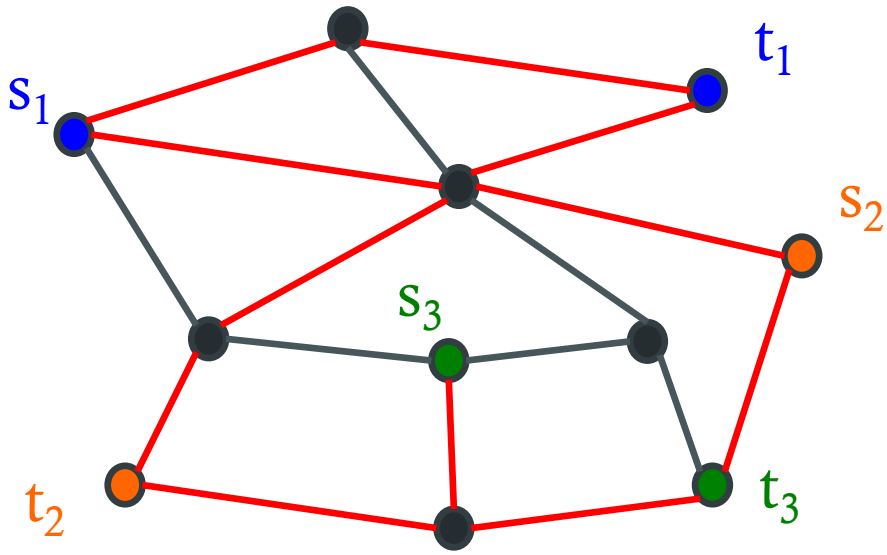
- undirected graph G=(V,E)
- integer requirement r(st) for each pair of nodes st

Goal: *min-cost* subgraph H of G s.t H contains r(st) *disjoint* paths for each pair st

$r(s_1t_1) = 2$
$r(s_2t_2) = 2$
$r(s_3t_3) = 1$



$r(s_1t_1) = 2$
$r(s_2t_2) = 2$
$r(s_3t_3) = 1$



SNDP Variants

Requirement

- EC-SNDP : paths are required to be **edge**-disjoint
- Elem-SNDP: **element** disjoint
- VC-SNDP: **vertex/node** disjoint

Cost

- edge-weights – *focus of this talk*
- node-weights

Known Approximations

	Edge Weights	Node Weights
EC-SNDP	2 [Jain'98]	O(k log n) [Nutov'07]
Elem-SNDP	2 [FJW'01]	O(k log n) [Nutov'09]
VC-SNDP	$O(k^3 \log n)$ [CK'09]	$O(k^4 \log^2 n)$ [CK'09+Nutov'09]

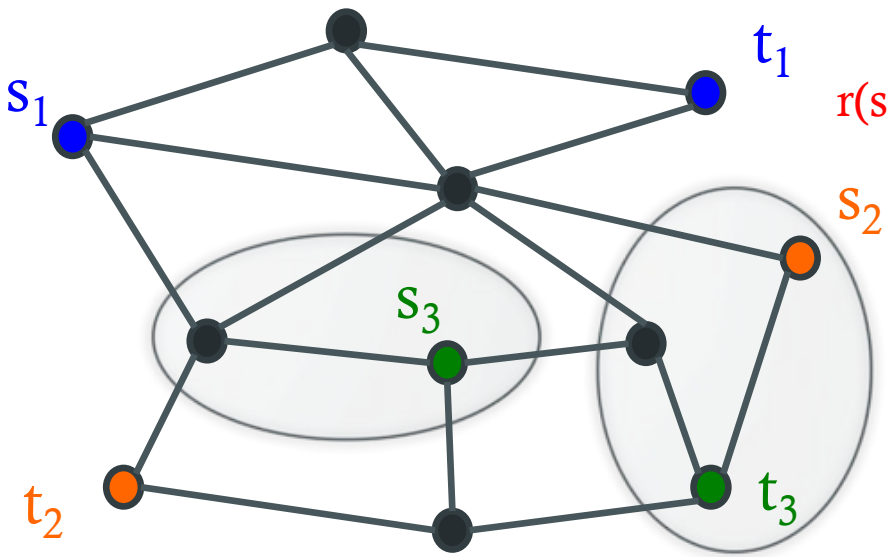
$k := \max_{st} r(st)$

$$\min \sum_e c(e)\, x(e)$$

$$x(\delta(A)) \geq r(st) \qquad A \subset V,\ A \text{ separates } st$$

$$0 \leq x(e) \leq 1$$

Cut-LP for EC-SNDP



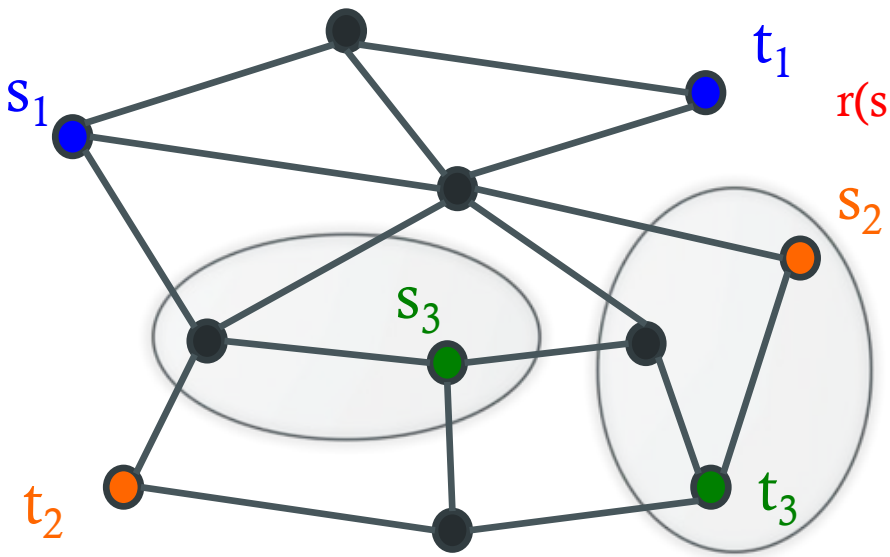
$r(s_1t_1) = r(s_2t_2) = 2$ and $r(s_3t_3) = 1$

$$\min \sum_e c(e)\, x(e)$$

$$x(\delta(A)) \geq r(st) \qquad A \subset V,\ A \text{ separates } st$$

$$0 \leq x(e) \leq 1$$

Cut-LP for EC-SNDP



$r(s_1t_1) = r(s_2t_2) = 2$ and $r(s_3t_3) = 1$

Theorem: [Jain] Integrality gap of Cut-LP is 2

Jain's iterated rounding

- Solve Cut-LP and find a *basic feasible solution* x
- There is always an e such that $x(e) = 0$ or $x(e) \geq ½$
  - x defined by a *laminar family of tight sets*
  - *Counting argument* to obtain contradiction if no edge e such that $x(e) = 0$ or $x(e) = 1$ or $x(e) \geq ½$
- Round up e with large value and recurse/iterate
  - *Residual problem* falls into more general class – *covering skew supermodular requirement by a graph*

Skew-supermodular functions

$f : 2^V \to \mathcal{Z}$ is skew-supermodular if for all $A, B \subseteq V$

$$f(A) + f(B) \leq f(A \cup B) + f(A \cap B)$$

or

$$f(A) + f(B) \leq f(A - B) + f(B - A)$$

Requirement function of SNDP is skew-supermodular

Covering skew-supermodular requirement by graph

- Given $G=(V,E)$ and skew-supermodular $f : 2^V \rightarrow \mathcal{Z}$
- Find $E' \subseteq E$ of min cost such that for all $A \subseteq V$

$$|\delta_{E'}(A)| \geq f(S)$$

Key point: Given G, f and subset $E' \subseteq E$ the *residual function* f' defined by $f'(A) = f(A) - |\delta_{E'}(A)|$ is skew-supermodular

Covering skew-supermodular requirement by graph

Key point: Given G, f and subset $E' \subseteq E$ the *residual function* f' defined by $f'(A) = f(A) - |\delta_{E'}(A)|$ is skew-supermodular

$|\delta_{E'}|$ is a *symmetric submodular function*

$$g(A) + g(B) \geq g(A \cup B) + g(A \cap B)$$

and

$$g(A) + g(B) \geq g(A - B) + g(B - A)$$

Residual problem

$f : 2^V \to \mathcal{Z}$ is *skew-supermodular* if for all $A, B \subseteq V$

$$f(A) + f(B) \leq f(A \cup B) + f(A \cap B) \quad \text{or}$$

$$f(A) + f(B) \leq f(A - B) + f(B - A)$$

$g : 2^V \to \mathcal{Z}$ is a *symmetric submodular function*

$g(A) + g(B) \geq g(A \cup B) + g(A \cap B)$ and

$g(A) + g(B) \geq g(A - B) + g(B - A)$

Implies $f - g$ is skew supermodular

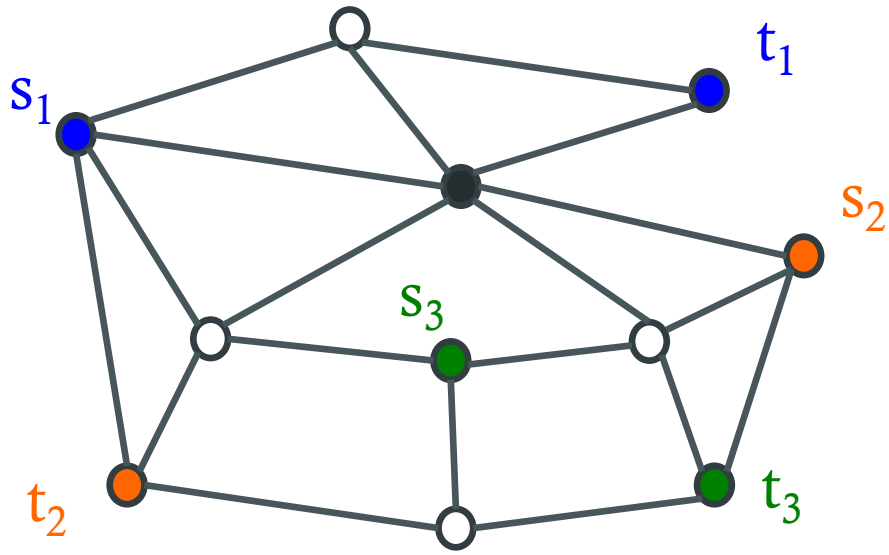
Contributions of paper

- Counting argument delicate – several different proofs over the years
  - This paper: another proof which is more **"combinatorial"** based on a new inductive claim
- Reduction of element connectivity to edge connectivity via hypergraphs
  - Based on paper of [Zhao-Nagamochi-Ibaraki'01]
  - Eliminates complex notation of set pairs etc
  - Easy to teach or give as an exercise
  - Connections to node weights – see paper

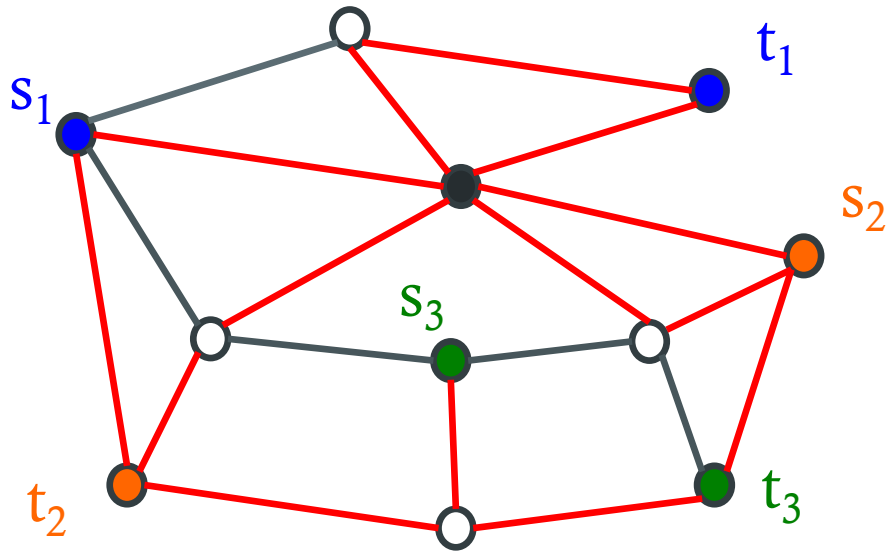
Element-SNDP

- Interpolates EC-SNDP and VC-SNDP
- Independent interest, other connections etc
- G=(V,E) and V partitioned into *terminals* T and *non-terminals* N
  - Requirement **only** between terminals
  - Paths for a terminal pair should be disjoint in ***elements*** (edges and non-terminals)
  - Wlog assume T is an independent set: then paths should be disjoint in non-terminals

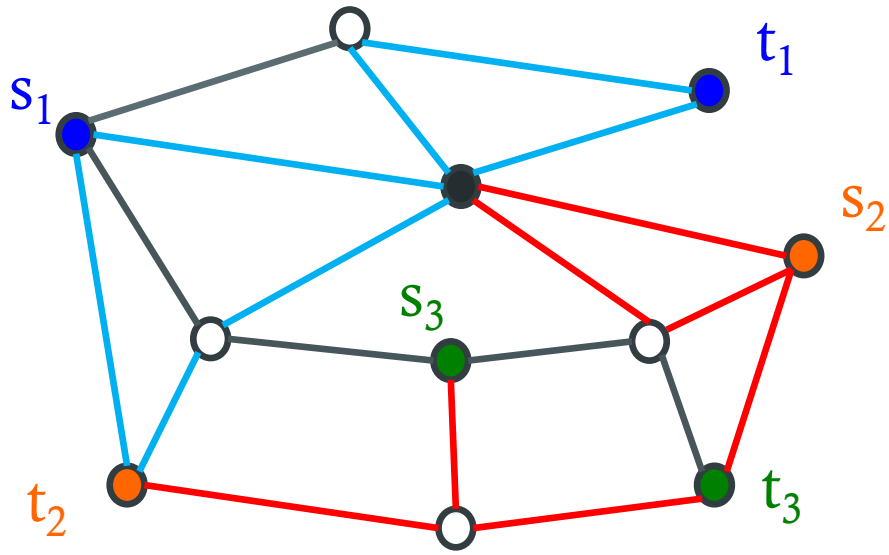
$r(s_1t_1) = 2$
$r(s_2t_2) = 3$
$r(s_3t_3) = 1$



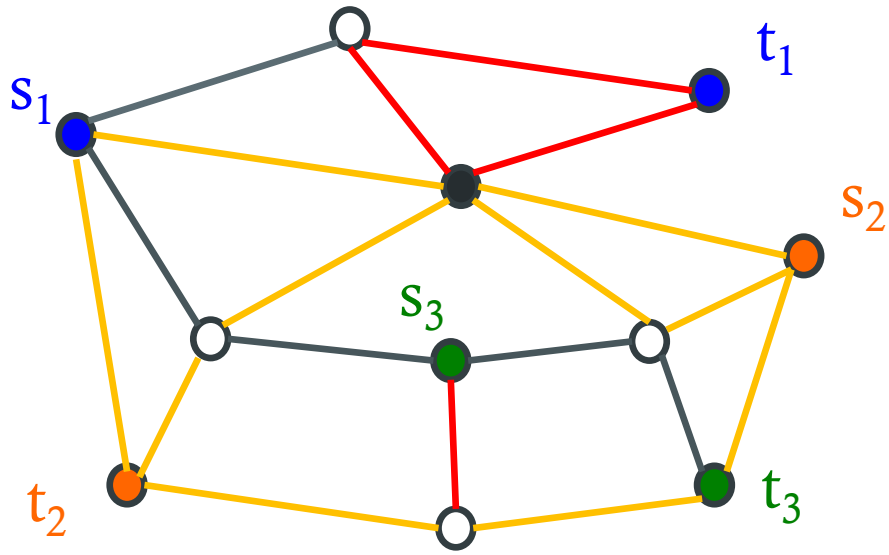
$r(s_1t_1) = 2$
$r(s_2t_2) = 3$
$r(s_3t_3) = 1$



$r(s_1t_1) = 2$
$r(s_2t_2) = 3$
$r(s_3t_3) = 1$



$r(s_1t_1) = 2$
$r(s_2t_2) = 3$
$r(s_3t_3) = 1$



2-approx for Elem-SNDP

- Based on iterated rounding [Fleischer-Jain-Williamson, Cheriyan-Vetta-Vempala]
- Needs a set-pair based relaxation and generalization of Jain's argument

Hypergraphic SNDP

[Zhao-Nagamochi-Ibarakai]

$G=(V,E)$ a hypergraph, each edge e in E a subset of V

Integer requirements $r(st)$ for each pair st of vertices

Find min cost subset of hyperedges such that for each cut $\delta(A)$ that separates st need $r(st)$ hyperedges cross A

Generalizes EC-SNDP to Hypergraphs

Residual problem

$f : 2^V \to \mathcal{Z}$ is *skew-supermodular*

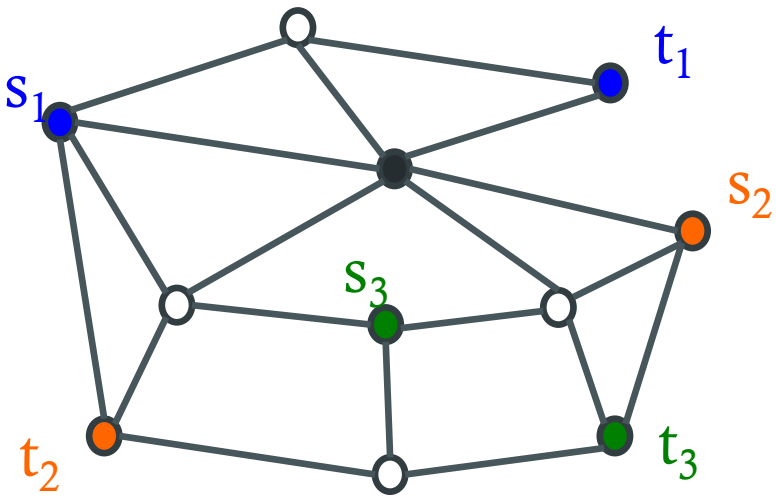
$g : 2^V \to \mathcal{Z}$ is a *symmetric submodular function*

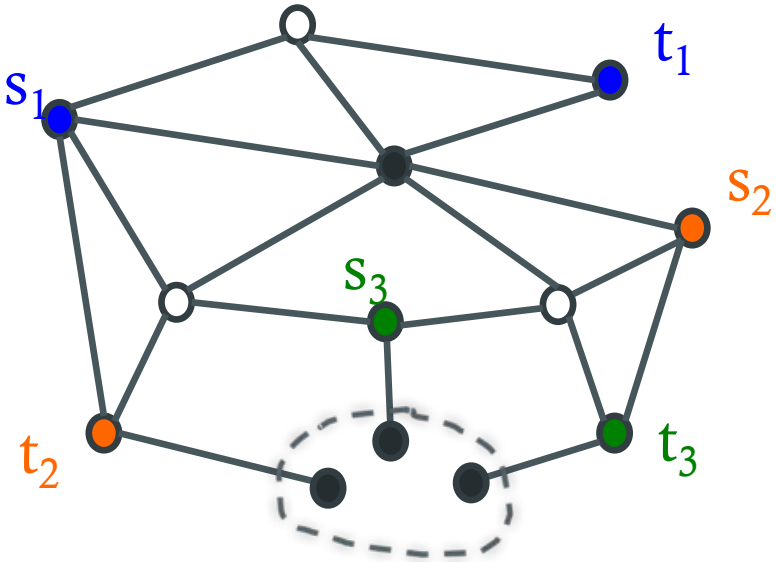
Implies $f - g$ is skew supermodular

Hypergraph cut function is symmetric submodular

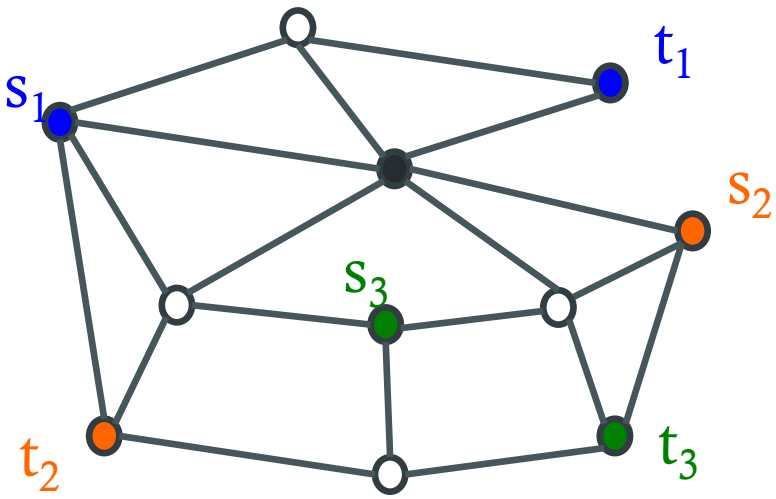
Main issue for iterated rounding: counting argument

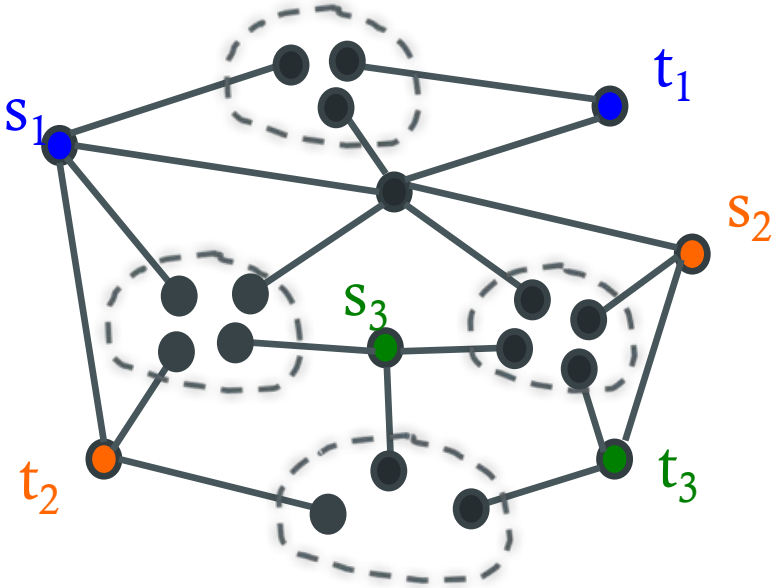
Reduction of Elem-SNDP to Hypergraphic SNDP





Reduction of Elem-SNDP to Hypergraphic SNDP





After reduction

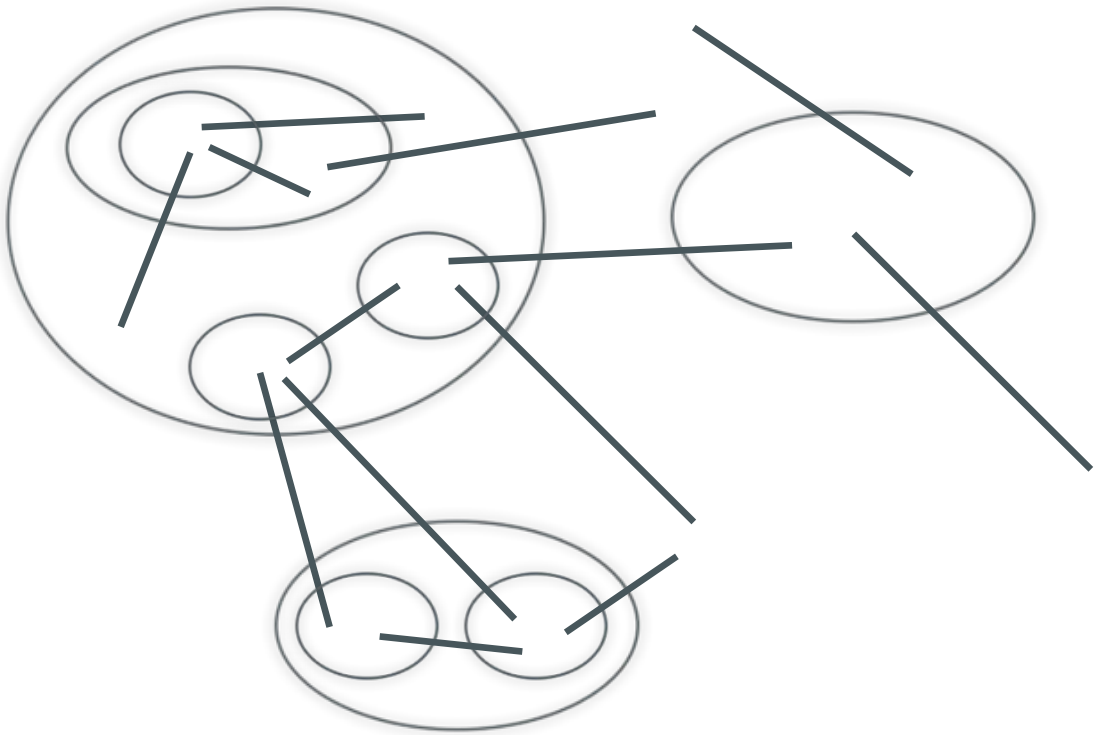
- New hyperedges have zero cost
- Include all hyperedges in solution
- Residual problem is covering skew supermodular function by *edges*
- 2-approximation for residual problem follows from Jain's argument for EC-SNDP

Jain's iterated rounding

- Solve Cut-LP and find a *basic feasible solution* x
- There is always an e such that $x(e) = 0$ or $x(e) \geq ½$
  - x defined by a *laminar family of tight sets*
  - *Counting argument* to obtain contradiction if no edge e such that $x(e) = 0$ or $x(e) = 1$ or $x(e) \geq ½$
- Round up e with large value and recurse/iterate
  - *Residual problem* falls into more general class – *covering skew supermodular requirement by a graph*

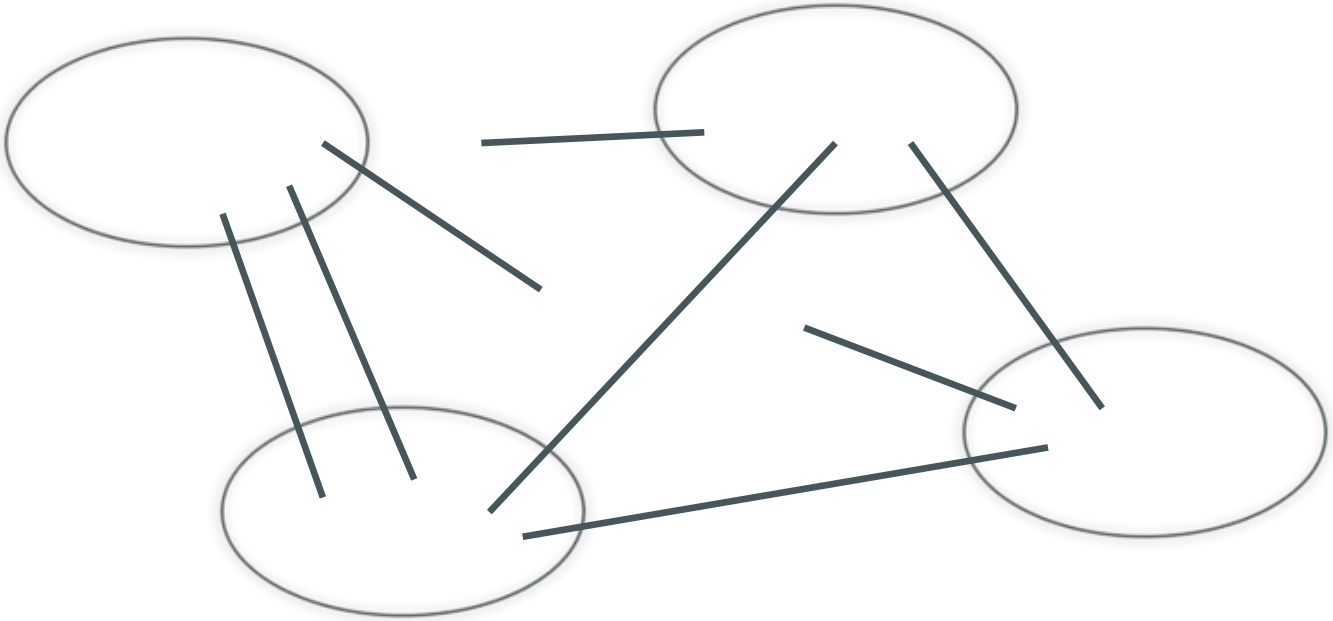
x is basic feasible solution and $0 < x(e) < 1$ for all e
Set S tight if $x(\delta(S)) = f(S)$

Theorem [Jain] x unique solution to system of m **tight** sets that form a *laminar* family



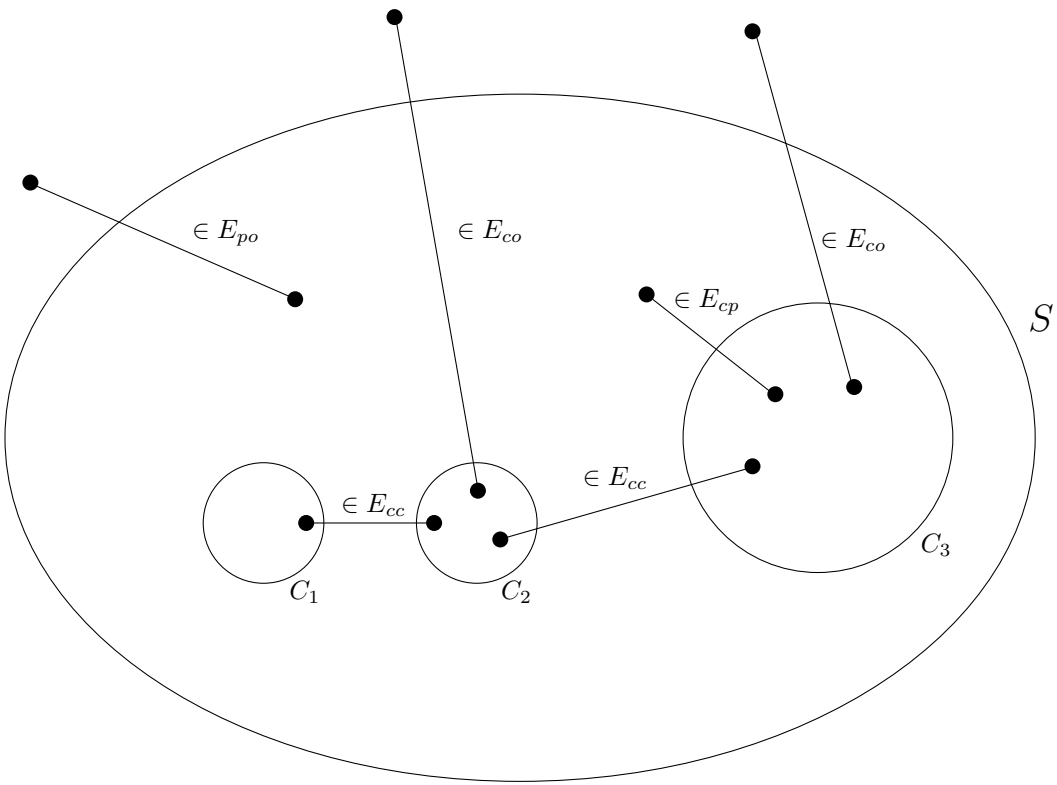
Counting argument: easy case

x is basic feasible solution and $0 < x(e) < 1$ for all e
x unique solution to system of m **tight** laminar sets



Claim: $f(S) \geq \alpha(S) - \beta(S)$

$\alpha(S)$: number of descendants of S
$\beta(S)$: number of edges with both end points inside S



Open Problem(s)

Can ideas help with *improvements/simplifications* for **degree bounded** network design? Especially element connectivity.

Thank You!