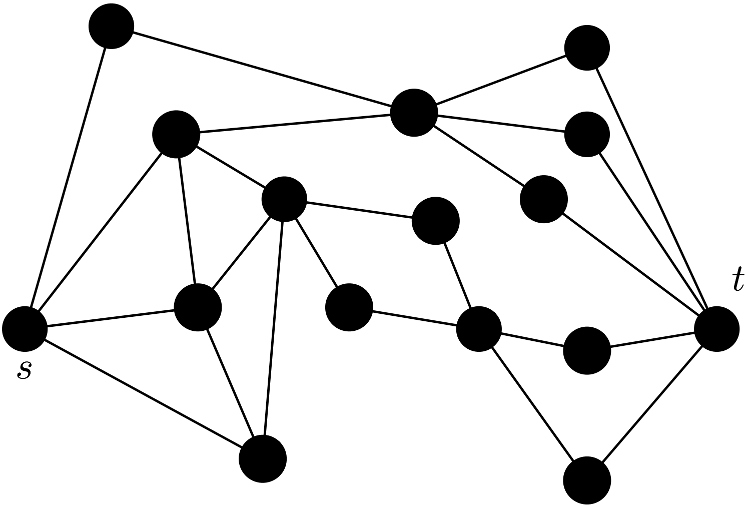
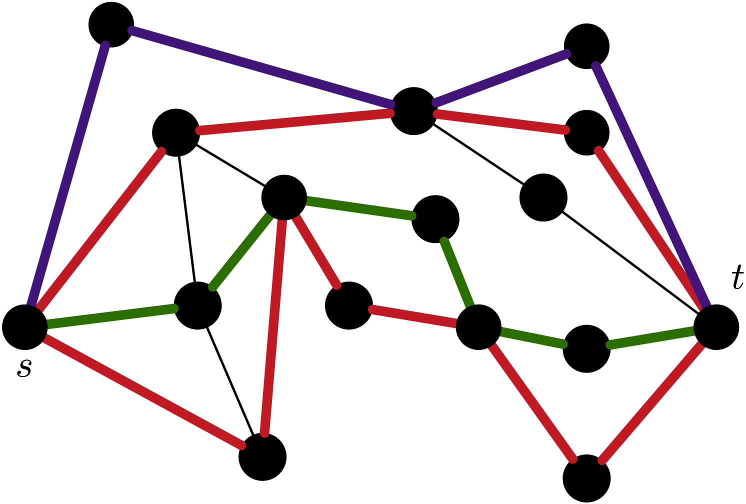
On element-connectivity preserving graph simplifications

Chandra Chekuri[1] Thapanapong Rukkanchanunt[2] Chao Xu[1]

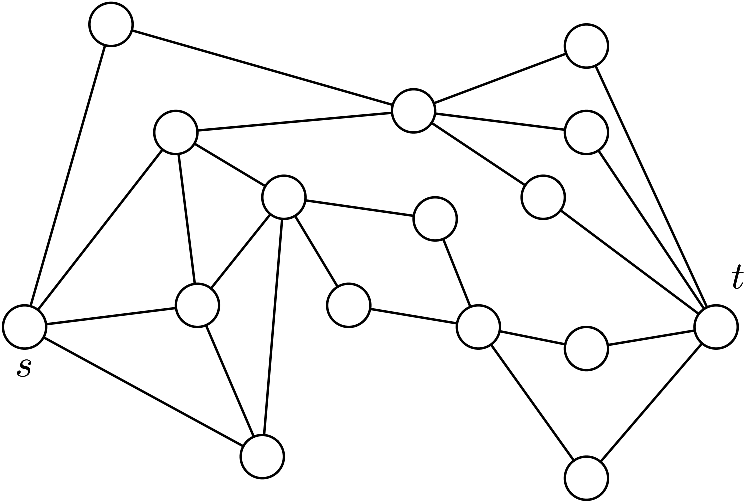
[1] University of Illinois at Urbana-Champaign
[2] Chiang Mai University

September 15, 2015

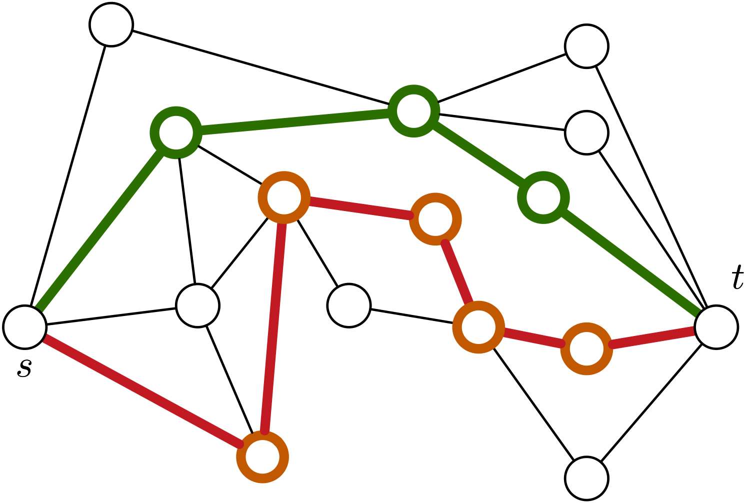

t
s

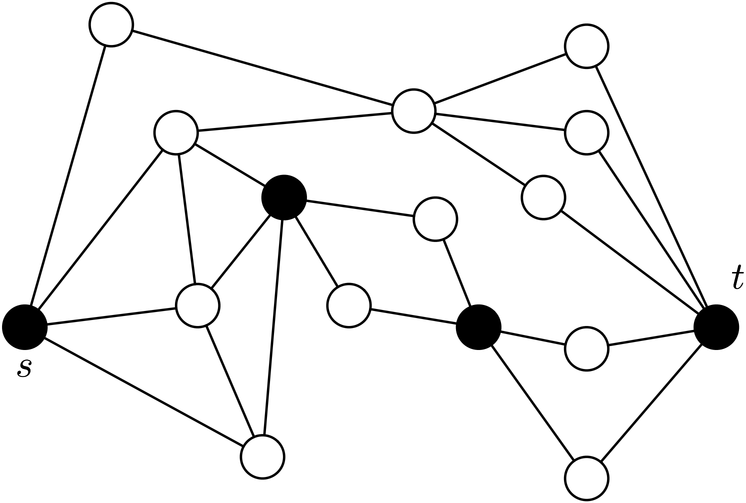


t
s

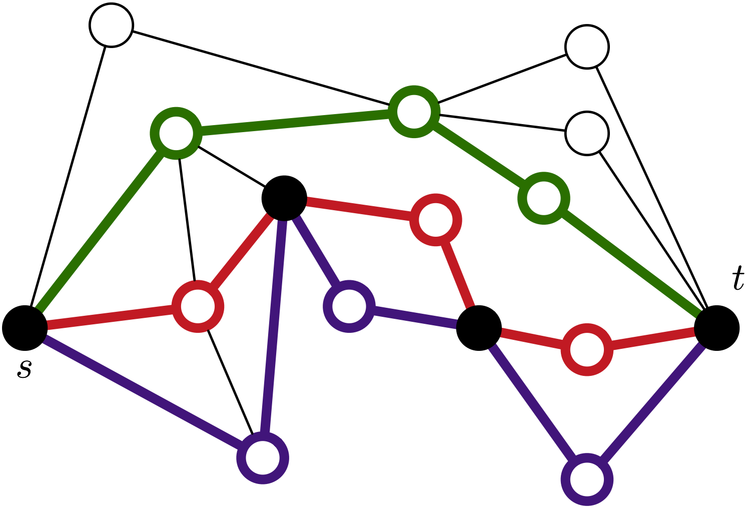


t
s


t
s



t
s


t
s

element-connectivity

Undirected graph $G = (V, E)$. $T \subset V$ is a set of terminal vertices.
$S = V \setminus T$ is the set of non-terminal(Steiner) vertices.

Definition

The *local element-connectivity* $\kappa'_G(x, y)$ is the number of maximum element-disjoint paths between x and y in G, where $x, y \in T$. The *global element-connectivity* is the minimum of all local element-connectivity.

Applications

1. Survivable network design
2. Packing element-disjoint Steiner trees
3. Network routing
4. ...

Reduction lemma

Edges between two non-terminals are called *reducible*.

Reduction lemma

Edges between two non-terminals are called *reducible*.

Theorem (Hind and Oellermann 1996)

If e is an reducible edge of G, then either $G - e$ or G/e preserves **global** *element-connectivity.*

Reduction lemma

Edges between two non-terminals are called *reducible*.

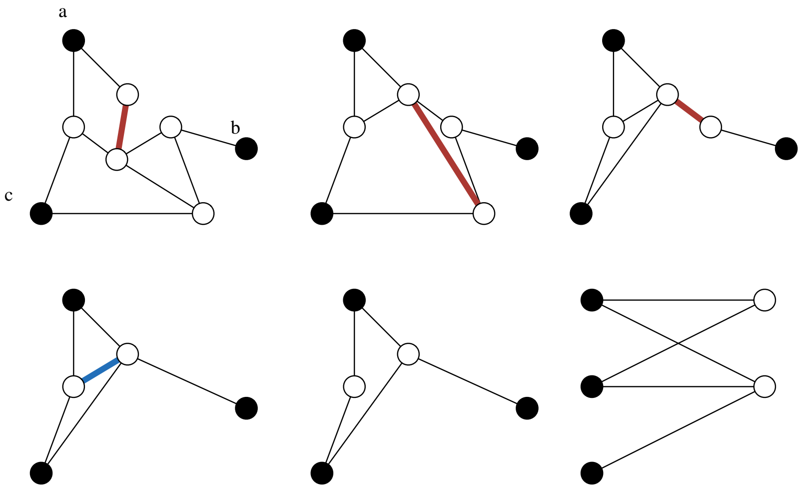
Theorem (Hind and Oellermann 1996)

If e is an reducible edge of G, then either $G - e$ or G/e preserves **global** *element-connectivity.*

Theorem (Reduction Lemma (Chekuri and Korula 2009))

If e is an reducible edge of G, then either $G - e$ or G/e preserves **local** *element-connectivity.*

Applying the reduction lemma



$\kappa'_G(a, c) = 2$, $\kappa'_G(x, y) = 1$ if $\{x, y\} \neq \{a, c\}$.

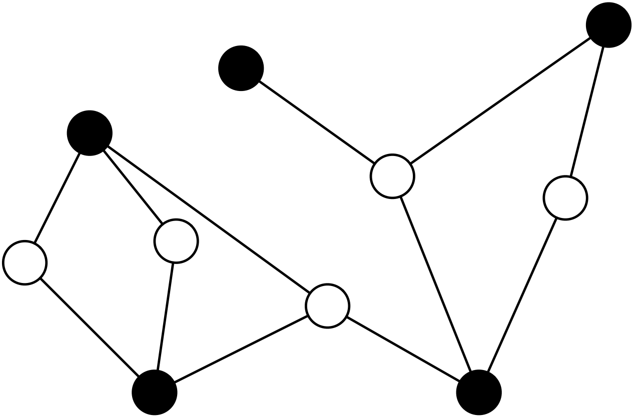
Flow-equivalent Tree

Definition

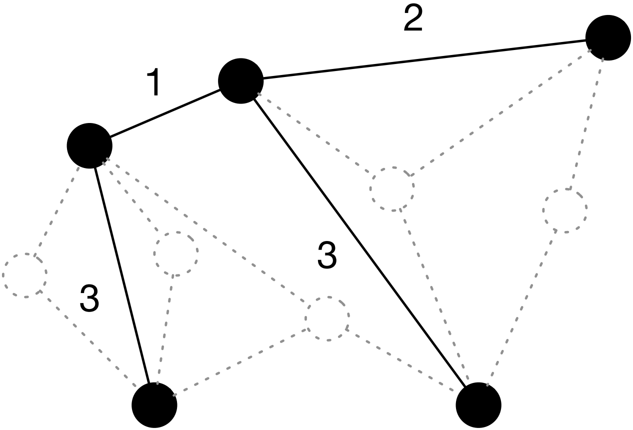
(R, w) is a *flow-equivalent tree* of G if

1. R is a tree on T.
2. w is a weight function on $E(R)$.
3. For all $s, t \in T$, $\kappa'_G(s,t) = \min_{uv \in P_{st}} w(uv)$, where P_{st} is the st-path in R.

Example of a flow-equivalent tree



Example of a flow-equivalent tree



Our results

- New proof of reduction lemma using bisubmodularity.
- Computational aspects of element-connectivity.

MF is the running time for maximum flow on a directed unit capacity graph on n vertices and m edges.

1. Local element-connectivity: $O(MF)$
2. All-pair element-connectivity: $O(tMF)$
3. Global element-connectivity: same as all-pair element-connectivity
4. Reduction: $O(tnm)$

Comparison of running time

	Global-conn	WHP	All-pair	WHP
edge	$\tilde{O}(m)$	$\tilde{O}(m)$	$\tilde{O}(n^{3.375})$	$\tilde{O}(nm)$
element	same as	all-pair	$O(t\ MF)$	$\tilde{O}(m^{\omega})$
vertex	$O(n^{1.75}m)$	$\tilde{O}(nm)$	$O(n^{4.5})$	$\tilde{O}(n^{2+\omega})$

Reduction: naive algorithm

1. Pick a reducible edge e in G
2. $$G \leftarrow \begin{cases} G - e & \text{if } \kappa'_G(x,y) = \kappa'_{G-e}(x,y) \text{ for all } x, y \in T \\ G/e & \text{otherwise} \end{cases}$$
3. If G is reduced, then we are done. Otherwise repeat.

Reduction: naive algorithm

1. Pick a reducible edge e in G
2. $$G \leftarrow \begin{cases} G - e & \text{if } \kappa'_G(x,y) = \kappa'_{G-e}(x,y) \text{ for all } x, y \in T \\ G/e & \text{otherwise} \end{cases}$$
3. If G is reduced, then we are done. Otherwise repeat.

- $O(m)$ iterations.
- $O(t^2)$ local element-connectivity computations in each iteration.
- running time $O(t^2 mMF)$.

Simple speed-up

1. flow-equivalent tree
    1. reduce to $t-1$ maximum flows.
    2. The flow-equivalent tree does not change through out the algorithm.
2. Maintain all the flows
    1. Remove flow path using reduced edge.
    2. Search for an augmenting path.

Running time improves to $O(tm^2)$.

Replace m with n

Vertex elimination, pick a non-terminal v, either

1. remove all reducible edges incident to v,
2. contract an reducible edge incident to v.

The right operation can be found and applied in $O(tm)$ time, without guess and check.

Open problems

1. Even faster algorithm for finding the reduced graph?
2. What if only global element-connectivity has to be preserved?
3. Find global element-connectivity faster than all-pair element-connectivity?