Edge-Disjoint Paths in Networks

(Part 2)

Sanjeev Khanna
University of Pennsylvania

A Quick Recap

[Chekuri, K, Shepherd '04, '05]

- Start with a multicommodity flow solution but use it only to partition the graph into well-linked instances. This step can be done for any undirected graph.
- Show that any well-linked instance contains a crossbar routing structure on which EDP is easy to solve. (Planar well-linked instances have a grid crossbar.)
- Route the given source-sink pairs using the crossbar.

EDP in General Graphs

Crossbar Conjecture: Let X be a well-linked set of terminals in a graph G. Then there is a crossbar reachable from X such that given any matching M on X, we can use the crossbar to route |X|/polylog(n) pairs in M with O(1) congestion.

If the crossbar conjecture is true, then integrality gap of the flow relaxation is polylog(n) with O(1) congestion.

Thus the crossbar conjecture implies that EDP has a polylog(n) factor approximation with O(1) congestion.

Converse is Also True!

If the integrality gap of the flow relaxation is polylog(n) with O(1) congestion, then the crossbar conjecture holds.

- For a well-linked instance, given any matching M on the terminals, the fractional flow value is $\Omega(|X|/\log n)$.
- If the integrality gap is polylog(n) with O(1) congestion, we can route |X|/polylog(n) pairs in M with O(1) congestion.
- Thus the terminals form the interface of a crossbar.

Integrality gap of flow relaxation $\leftrightarrow$ Crossbar conjecture

Proving the Crossbar Conjecture …

Plan: Show that in a well-linked instance on k terminals, we can embed with constant congestion a low-degree expander of size k/polylog(n).

Given a low-degree expander H and any matching M on the vertices of H, we can route a 1/polylog(n) fraction of pairs in M in an edge-disjoint manner.

- Greedily routing pairs along shortest paths suffices.
- Low degree implies that we can actually get a vertex-disjoint routing.

α-Expanders

A graph $H(V',E')$ is an α-expander if for any $S \subseteq V'$ with $|S| \leq |V'|/2$, we have $|E(S, V' \setminus S)| \geq \alpha |S|$.

- Same definition as α-cut well-linked.
- We will be interested in degree d-bounded α-expanders where d = poly-log(n) and $\alpha = \Theta(1)$.

Embedding an Expander

We say that an expander $H(V', E')$ can be embedded with congestion c in $G(V,E)$ if there is a mapping ϕ s.t.

- for each $v \in V'$, $\phi(v)$ is a connected subgraph in G,
- for each $(u,v) \in E'$, there is a path $P_{u,v}$ in G that connects some vertex in $\phi(u)$ to some vertex in $\phi(v)$, and
- no edge appears in more than c connected subgraphs or paths.

Two Key Tools

- Solving EDP in expander graphs.
- Building expanders via a cut-matching game.

Disjoint Paths on an Expander

Theorem [Rao-Zhou '06]: Suppose G is a d-regular $\Theta(1)$-expander, and let M be any collection of $n/2$ disjoint pairs in G. Then one can route $\Omega(n/(d^2 \log n))$ pairs on vertex-disjoint paths.

Algorithm:

- Among the yet unrouted pairs, route a pair with the shortest path in the current graph.
- Remove all vertices on the path, and repeat.

Analysis of Expander Routing

For concreteness, assume $\alpha = 1$.

Set $L = 4d \log n$.

Stop as soon as the algorithm when the shortest path length exceeds L.

- Each routed pair removes at most $(L+1)d \approx d^2 \log n$ edges from the graph.
- We will show that when the algorithm terminates, many edges in the graph must have been removed.
- Combining the two facts gives the desired result.

Analysis of Expander Routing

E' = set of edges removed by the algorithm.

E'' = set of edges remaining in the graph when we stop.

Claim 1: G has a multicut of size $\leq |E'| + |E''|(\log n)/L$.

- Routed pairs are disconnected by edges in E'.
- Unrouted pairs can be disconnected by a fractional solution of size $|E''|/L$: assign a weight of $1/L$ to each edge.

Thus there is an integral solution to disconnect all unrouted pairs that has size $|E''|(\log n)/L$.

Analysis of Expander Routing

Claim 2: Any multicut of an α-expander must have at least $(\alpha\ n)/2$ edges.

- If E^* is a multicut, then removal of E^* leaves a graph where each connected component has at most $(n/2)$ vertices.
- By definition of an α-expander, each connected component C has at least $\alpha|V(C)|$ edges going out from C.
- Thus E^* must contain at least $(\alpha\ n)/2$ edges.

Putting Together …

Combining Claims 1 and 2 (with $\alpha = 1$), we get:

$|E'| + (|E''| \log n)/L \geq n/2$

Using our choice of $L = (4d \log n)$, we conclude that $|E'| \geq n/4$.

Hence at least $(n/4)/d(L+1) = \Omega(n/d^2 \log n)$ pairs must have been routed.

Cut-Matching Game [Khandekar,Rao,Vazirani'06]

Cut Player: wants to build an expander.

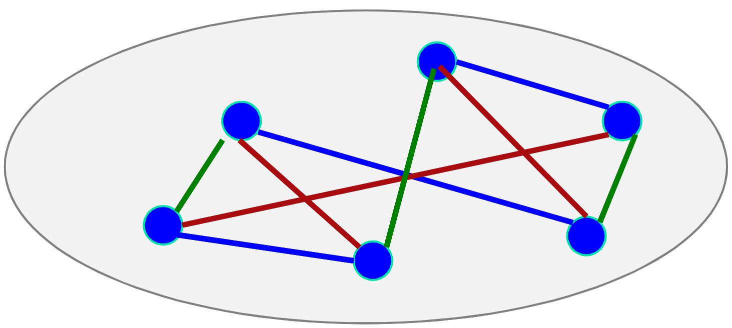
Matching Player: wants to delay its construction.

- The game proceeds in rounds where in each round
  - Cut player picks a partition of vertices into 2 equal-sized sets, say, A and B.
  - Matching player responds with an arbitrary matching between the sets A and B.
- How many rounds are needed to obtain an expander?

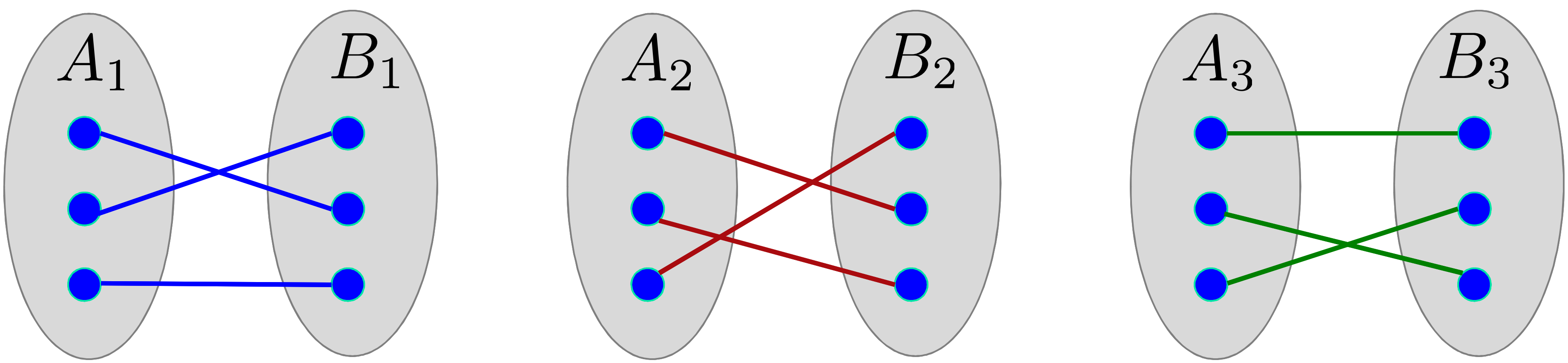
Cut-Matching Game [Khandekar,Rao,Vazirani'06]

Cut Player: wants to build an expander.

Matching Player: wants to delay its construction.

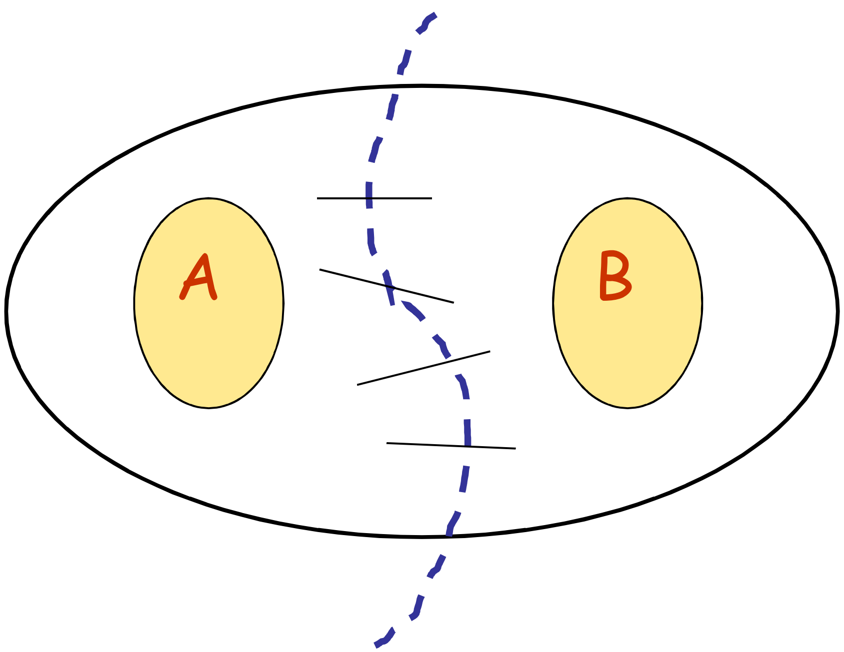


There is a strategy for the cut player s.t. after $O(\log^2 n)$ rounds, we get a $\Theta(1)$-expander with degree = $O(\log^2 n)$.

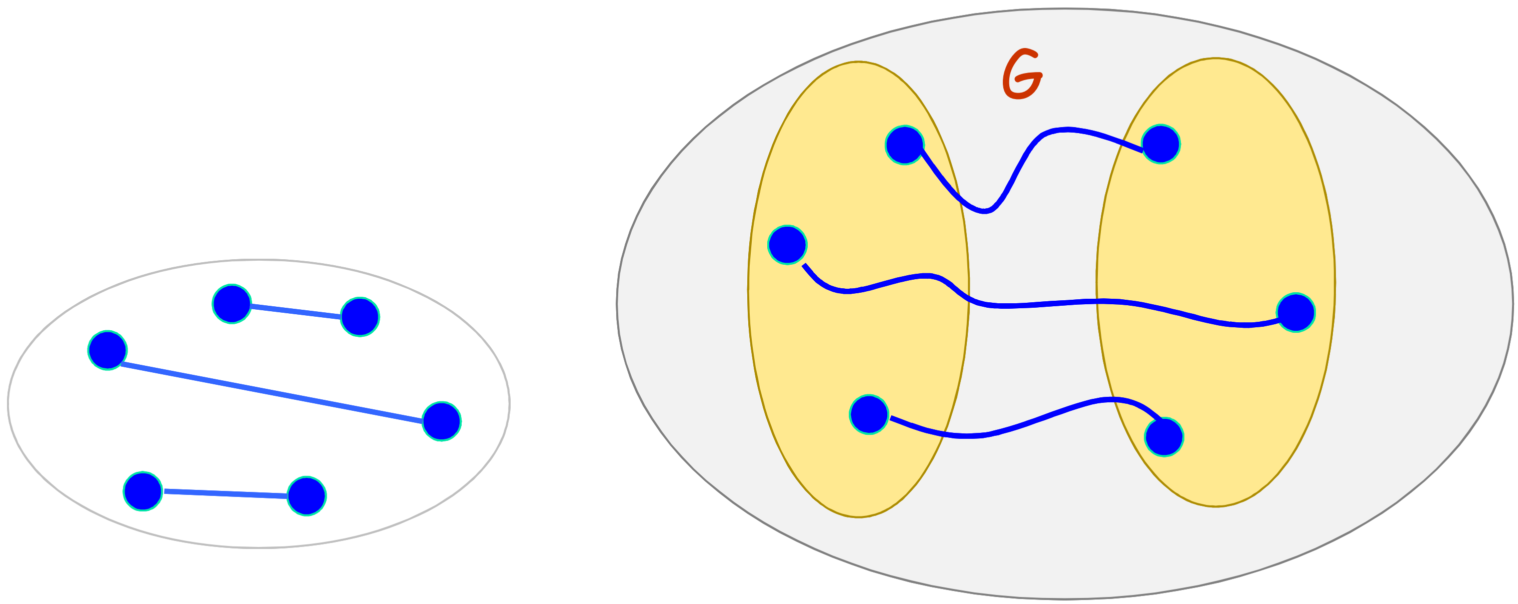


Connection to Well-Linked Sets

Claim: Let X be a well-linked set in a graph G. Then given any partition of X into 2 equal-sized sets A and B, there exist $|X|/2$ edge-disjoint paths from A to B.



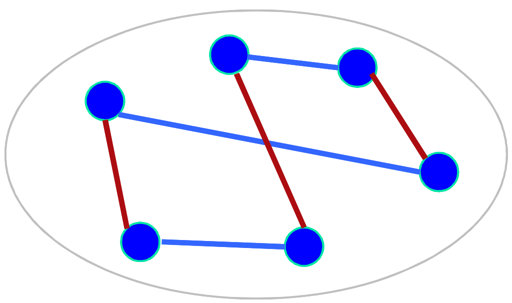
Expander Embedding on Terminals

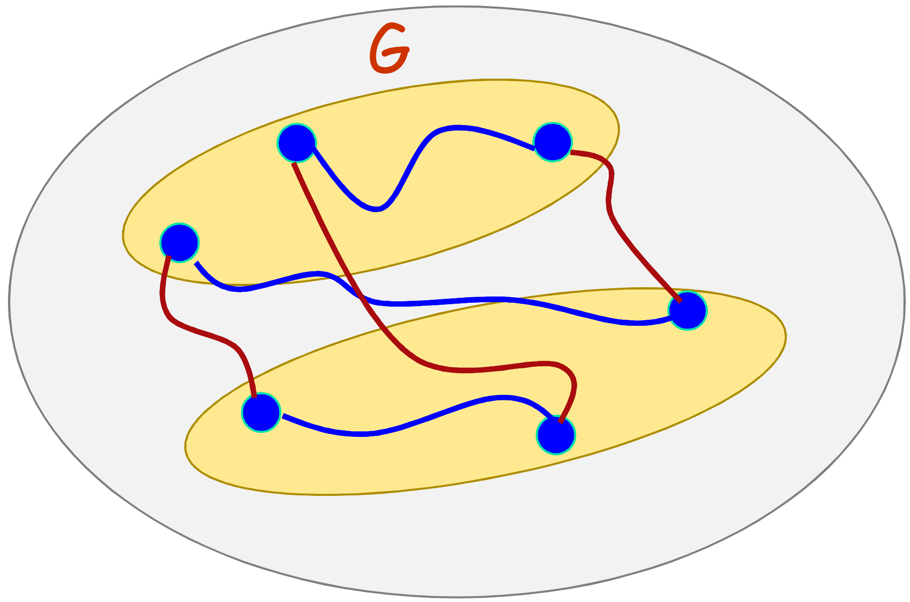


Matching Edges $\leftrightarrow$ Edge-Disjoint Paths between Terminals

Expander Embedding on Terminals

Expander on X





After $O(\log^2 k)$ iterations, we get an expander on X that can be embedded in the graph G.

Problem: $\Omega(\log^2 k)$ congestion!

The [Rao-Zhou '06] Approach

Theorem: If min-cut in G is $\Omega(\log^3 n)$, then an expander on the terminals X can be embedded with congestion 1.

- Randomly partition G into $\log^2 n$ edge-disjoint graphs $G_1,\ldots,G_h$.
- Use the large min-cut condition to show that each G_i is still well-linked for the terminals.
- Run the cut-matching game: use G_i to route the matching in iteration i.

The [Andrews '10] Approach

[Andrews '10] Min-cut condition can be eliminated provided we allow poly(log log n) congestion.

- Contract regions in the graph that violate the min-cut condition to a single node.
- Now use [Rao-Zhou '06] approach to embed an expander.
- Poly(log log n) congestion is needed to route through the contracted regions.

The [Chuzhoy' 12] Approach

- Find $\Theta(\log^2 k)$ vertex-disjoint well-linked sets in G of size $k/\text{polylog}(k)$ each.
- Each round of the cut-matching game can be run in a distinct well-linked set – no accumulation of congestion.
- Show that terminals can be routed to these well-linked sets with constant congestion.
- A constant congestion expander embedding.

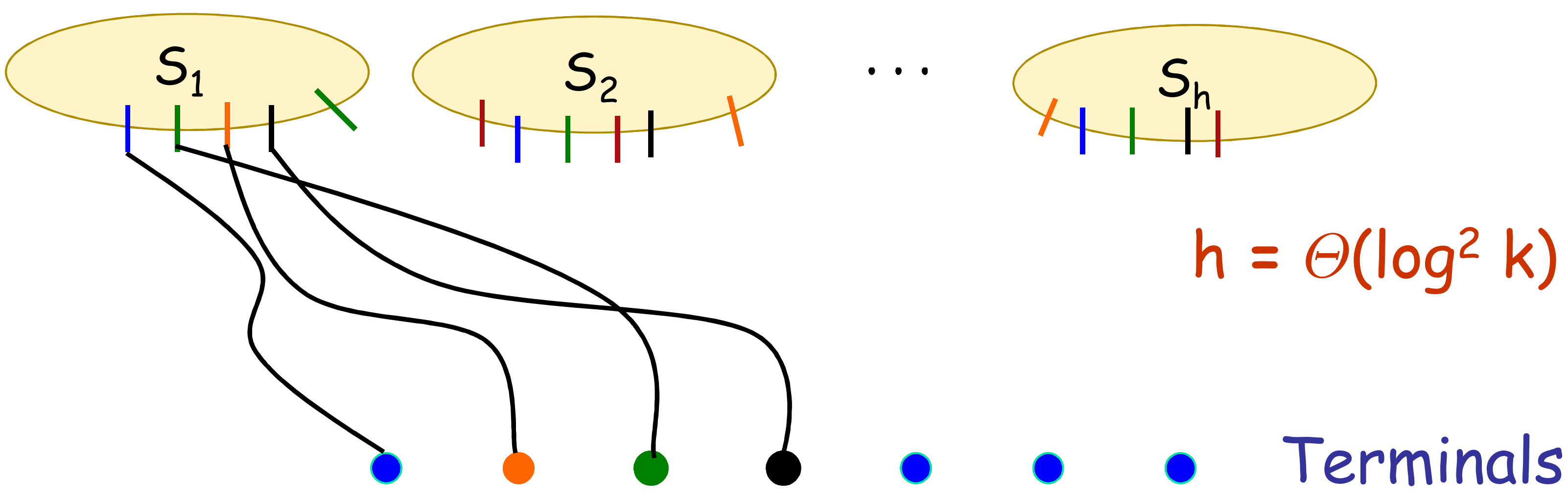
Good Family of Sets

- Identify $h = \Theta(\log^2 k)$ vertex-disjoint sets $S_1,\ldots,S_h$ s.t.
  - Each S_j has a boundary $out(S_j)$ of size $k/polylog(k)$.
  - Each S_j is well-linked w.r.t. its boundary.
  - Each S_j can reach $k/polylog(k)$ terminals using edge-disjoint paths.

- The set S_j is used to implement round j of the cut-matching game.

Such a family of sets is called a good family of sets.

Good Family of Sets



$|out(S_j)|$= # of edges on the boundary of S_j = k/polylog(k).

Each S_j is well-linked w.r.t. its boundary i.e. $out(S_j)$.

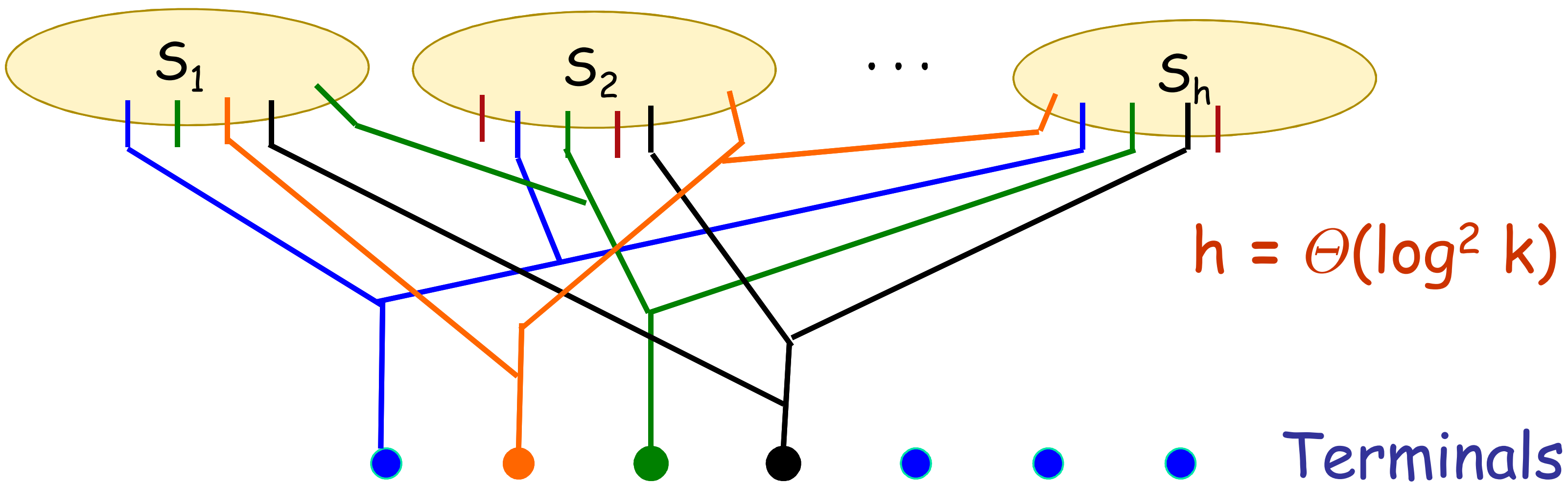
Each S_j is connected by edge-disjoint paths to k/polylog(k) terminals.

Routing Trees

Theorem [Chuzhoy '12]: Given a good family of sets, we can find $k/\text{polylog}(k)$ trees in G, say, T_1, T_2, ... such that

- each tree T_i is rooted at a distinct terminal,
- each tree T_i connects to a distinct edge on the boundary $out(S_i)$ of each S_i, and
- no edge in the graph is used by more than $O(1)$ trees.

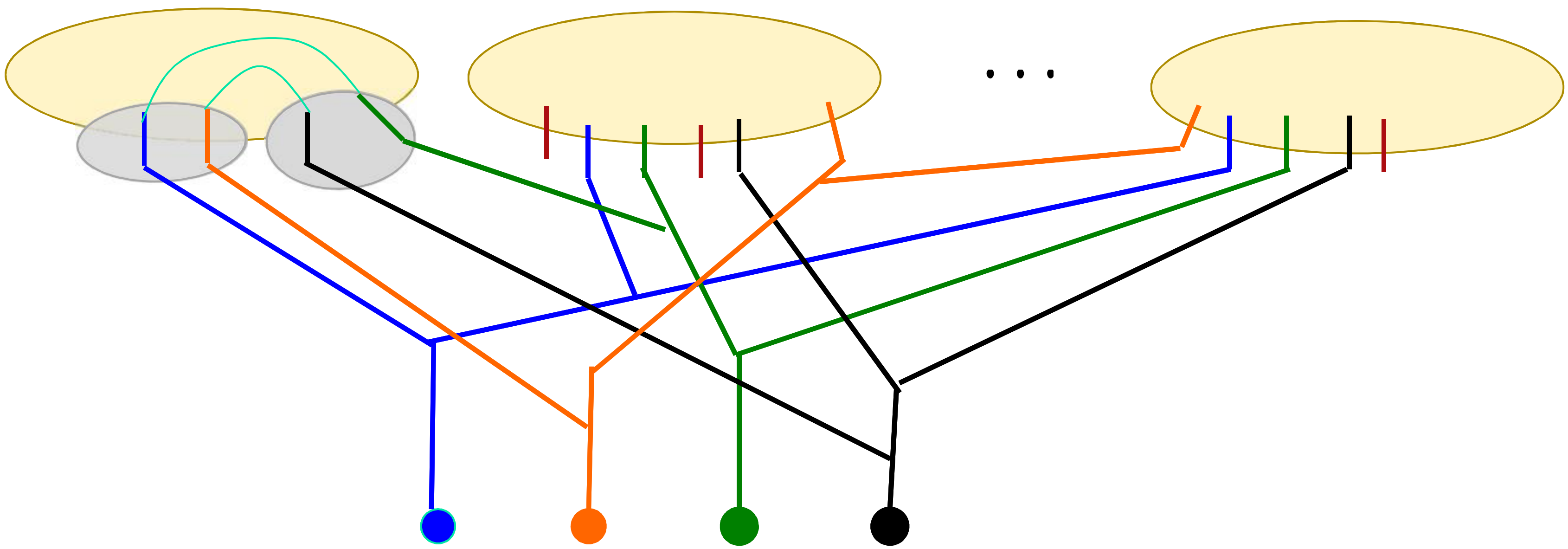
Good Family of Sets



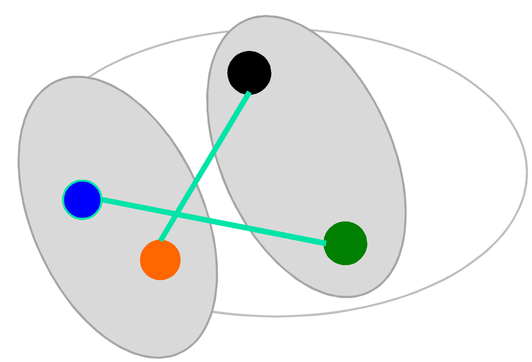
Each S_j is well-linked w.r.t. its boundary i.e. $out(S_j)$.

For each terminal t_i, there is a tree T_i that spans t_i and a distinct edge e_{ij} in $out(S_j)$ for each j.

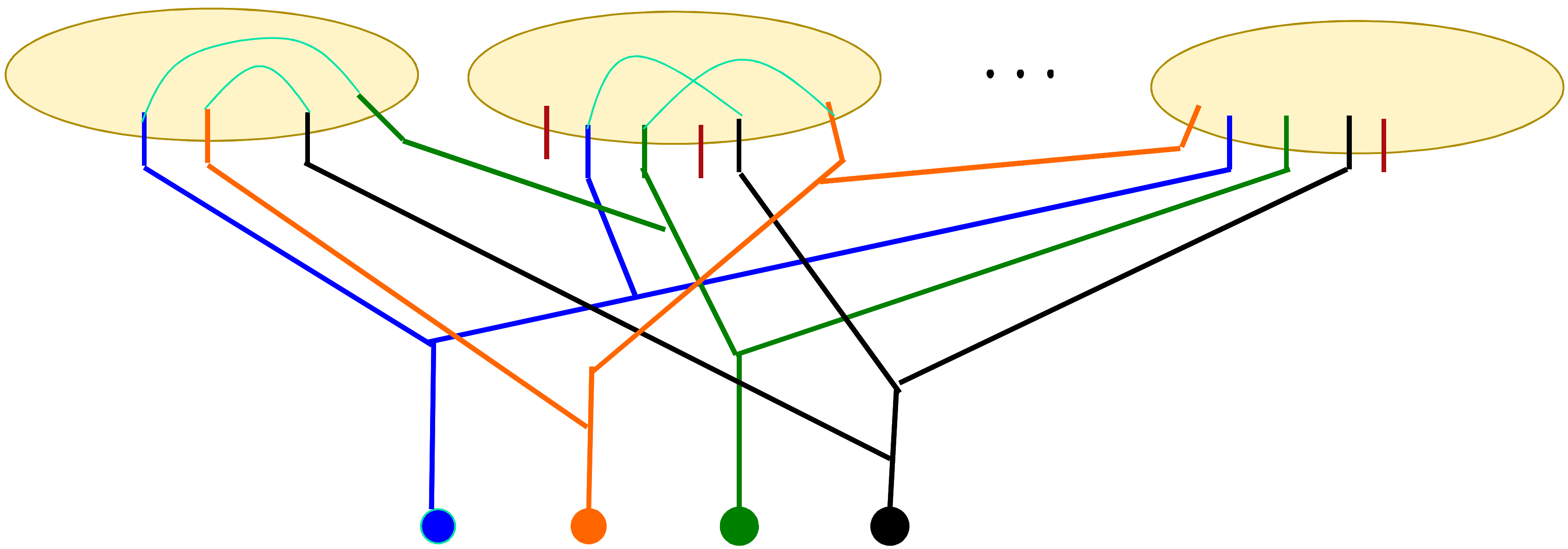
Embedding an Expander



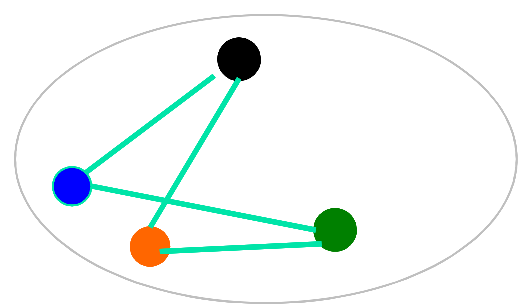
Implementing one round of the cut-matching game.



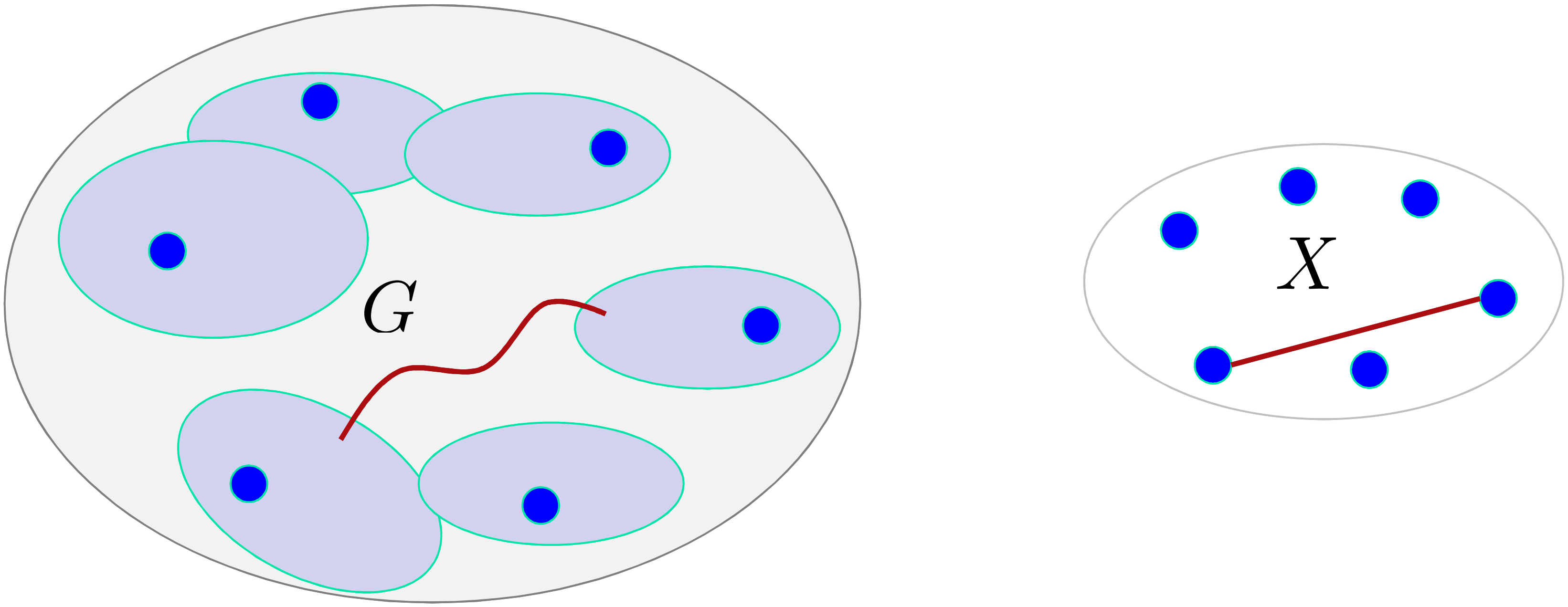
Embedding an Expander



After $\Theta(\log^2 k)$ iterations, we obtain an expander on terminals embedded in G.



Routing on the Embedded Expander



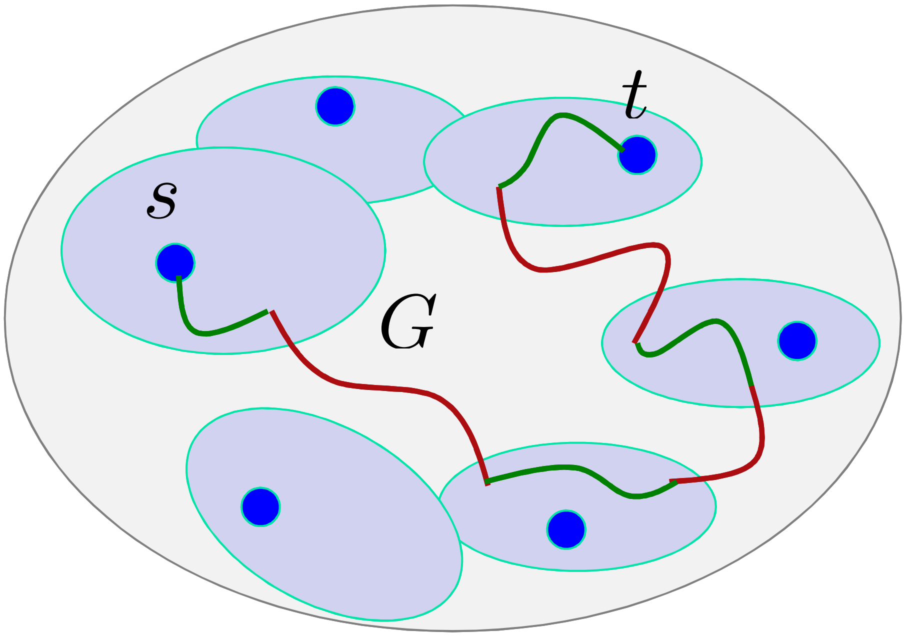
Expander vertex: a connected component in G containing the terminal.

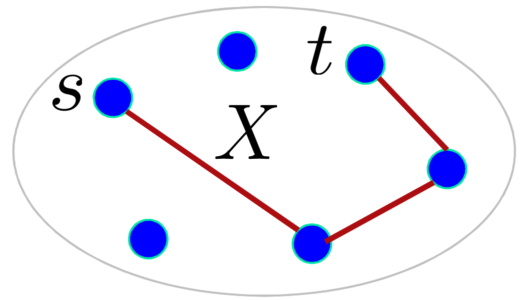
Expander edge: a path in G connecting some pair of vertices in the two components.

An edge of G belongs only to O(1) components/paths.

Degree of each expander vertex is $\Theta(\log^2 k)$.

Routing on the Embedded Expander





Routing on vertex-disjoint paths in the expander corresponds to a constant congestion routing in G !

Further Improvement: Polylog(n) approximation with congestion 2 [Chuzhoy, Li '12]

Expander Embedding Details

Starting Point:

- A graph $G(V,E)$ that has a well-linked terminal set X of size k, the degree of each vertex in the graph is at most 4, and the degree of each terminal is 1.

Goal:

- Embed a low-degree expander of size $k/\text{polylog}(k)$ on the terminals with constant congestion on the edges.

Two Challenges

- How does one find a good family of sets?
- How do you use a good family to find the routing trees?

We will primarily focus on the second task.

Routing Trees for Terminals

- We will use a good family of sets to construct a tree for each terminal that allows the terminal to reach every good set – a unique edge on the boundary of each S_i.

- Specifically, we will find k/polylog(k) trees in G, say, T_1, T_2, ... such that
  - each tree T_i is rooted at a distinct terminal,
  - each tree T_i connects to a distinct edge on the boundary out(S_i) of each S_i, and
  - no edge in the graph is used by more than O(1) trees.

Some More Tools

The Splitting Off Operation

An operation to modify edges in a graph while preserving pairwise connectivity.

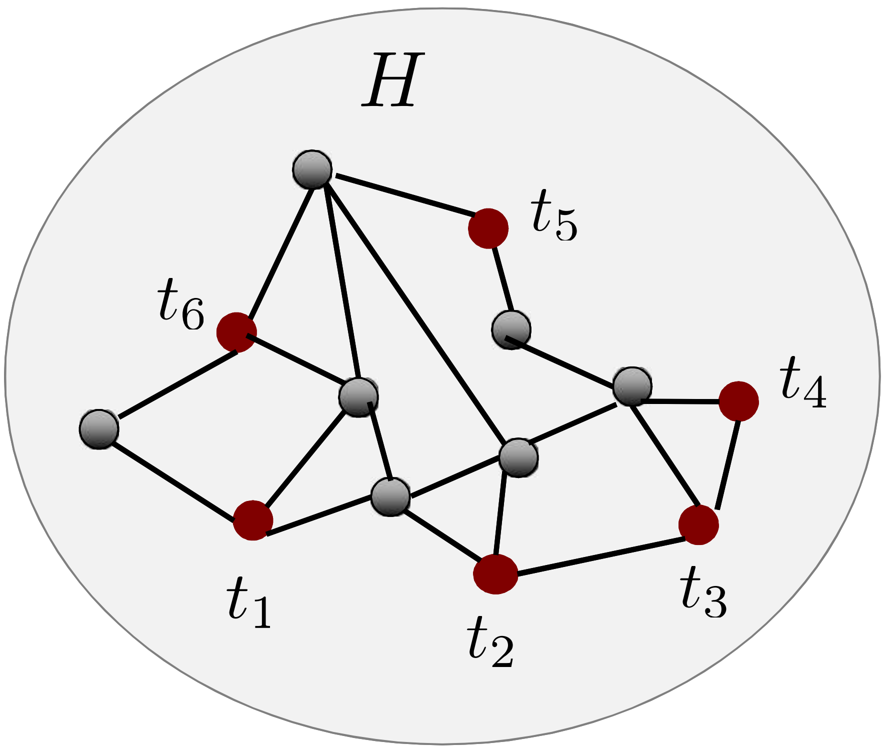
Splitting Off operation: Given a pair (v,y) and (v,z) of edges in an undirected graph, the splitting off operation replaces them with edge (y,z).

Splittable Pair of Edges: A pair of edges (v,y) and (v,z) is splittable if replacing them with the edge (y,z) preserves all pairwise edge-connectivities (except for pairs involving v).

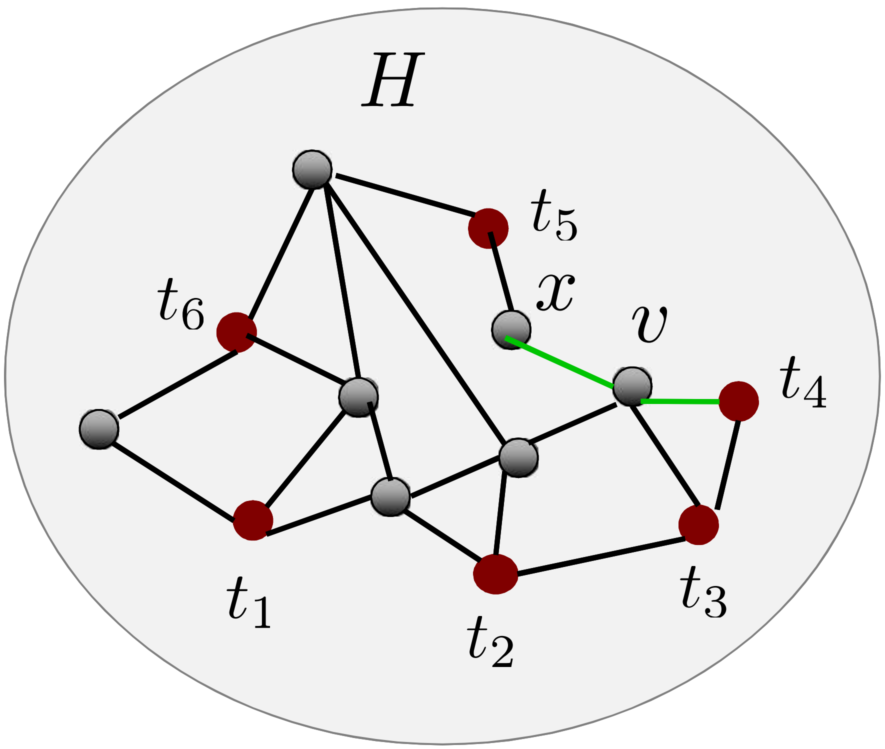
Mader's Theorem

Mader's Theorem: Given any undirected graph G and a vertex v of degree not equal to 3 such that there is no cut-edge incident on v, there always exists a splittable pair of edges incident on v.

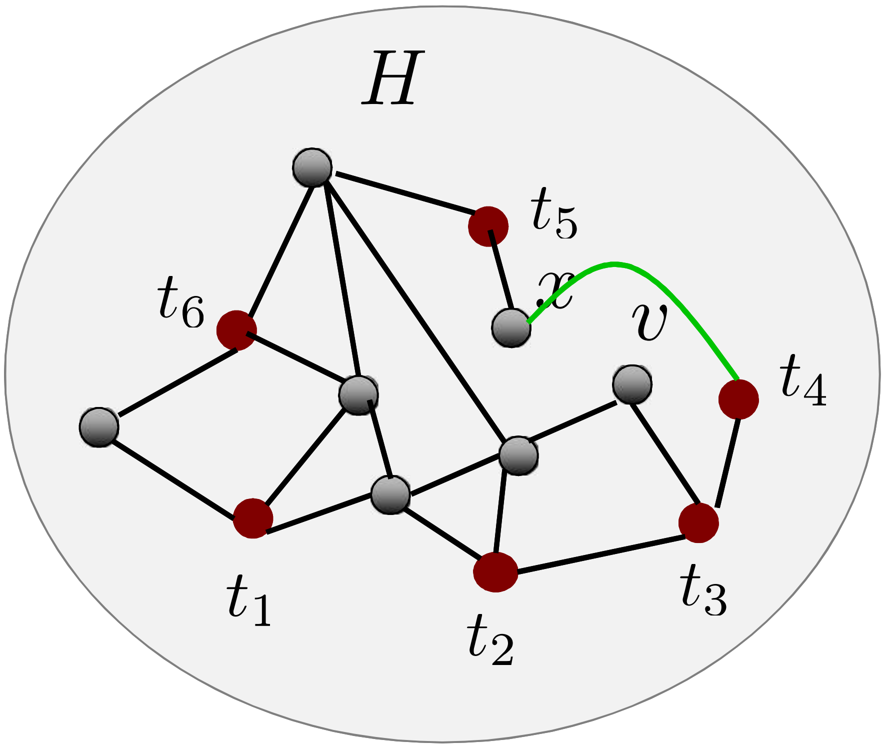
We can repeatedly apply this theorem to preserve connectivity between a special set of vertices while eliminating edges incident on other vertices.



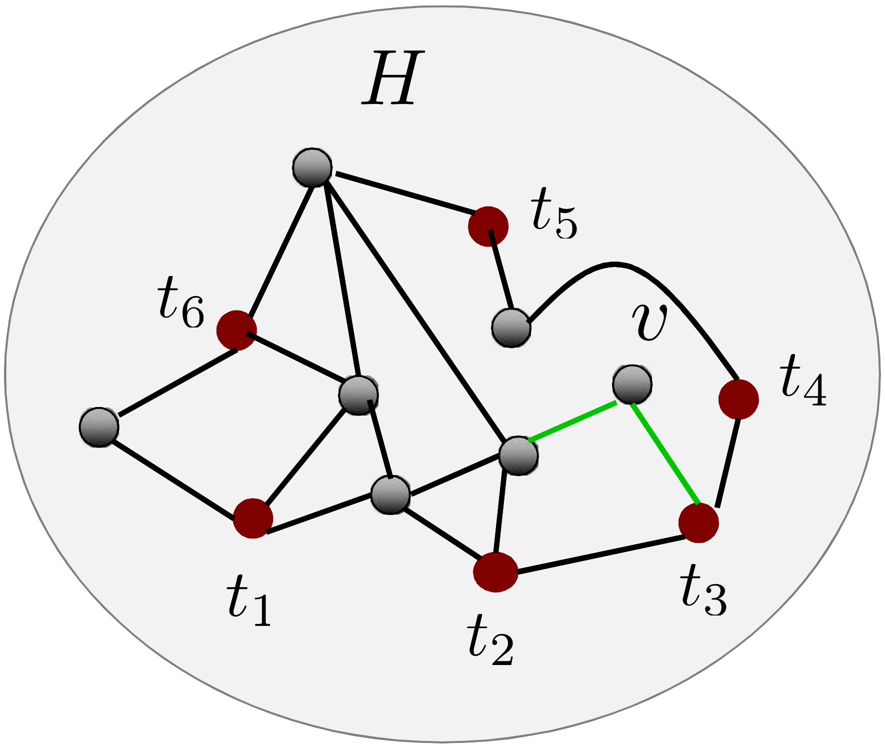
Splitting off to preserve pairwise edge connectivities between the t_i vertices.



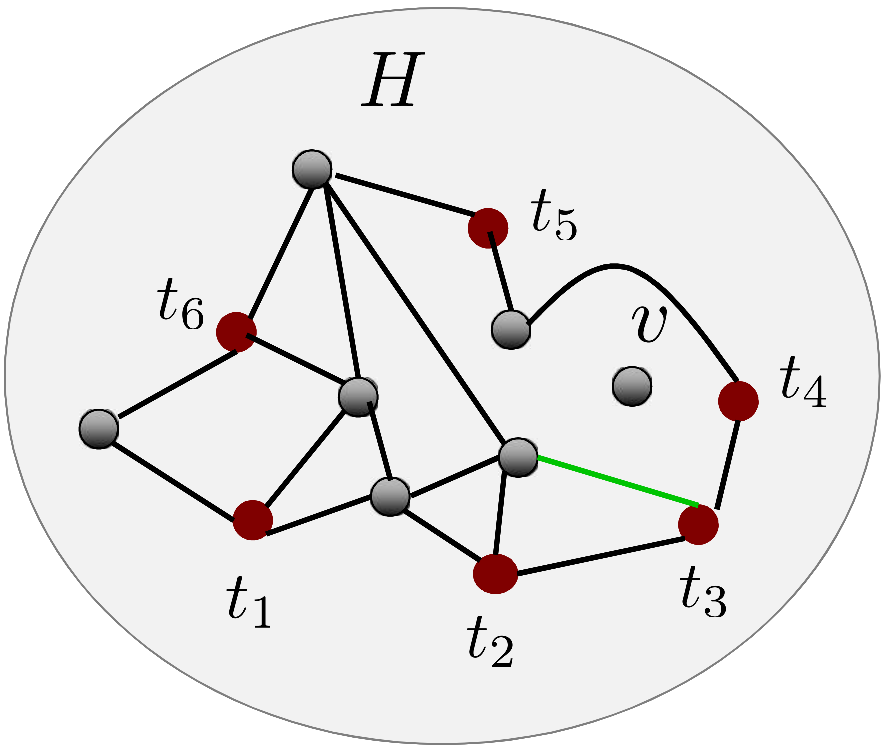
Splitting off operation
at vertex v



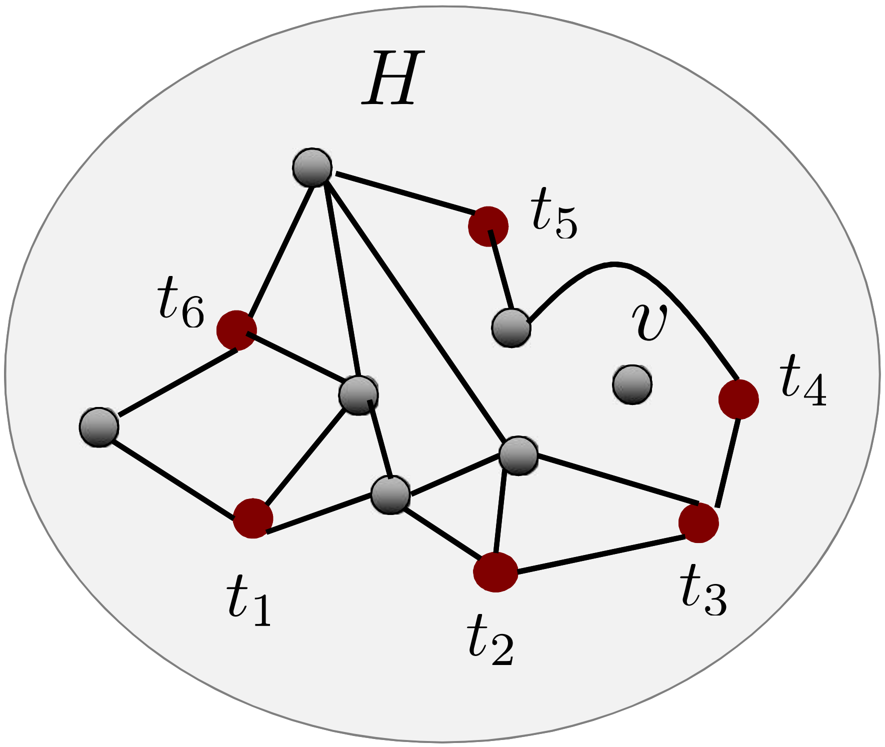
Splitting off operation at vertex v



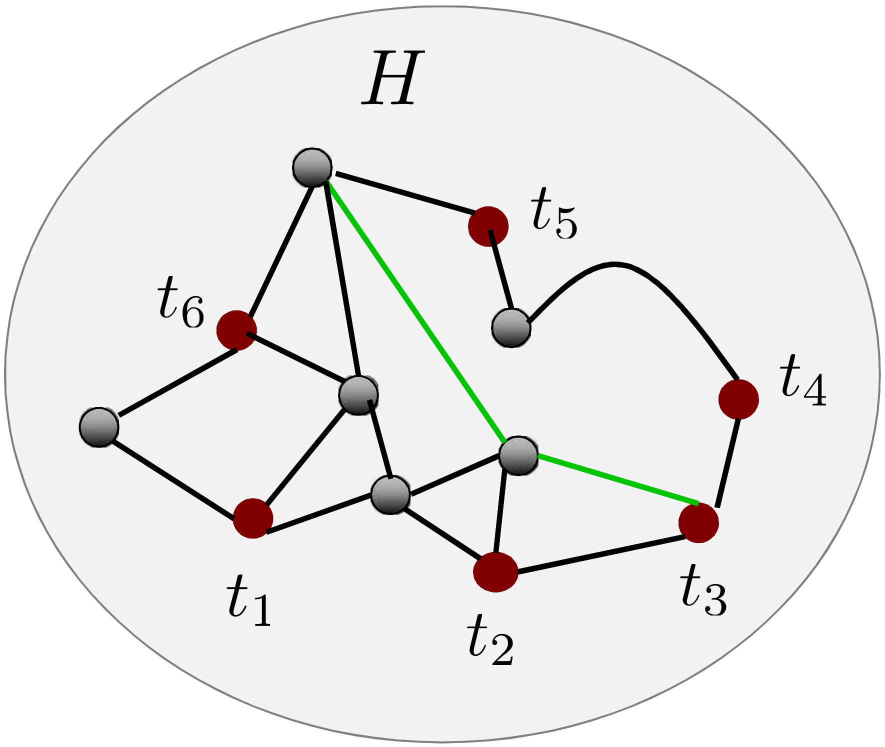
Splitting off operation at vertex v



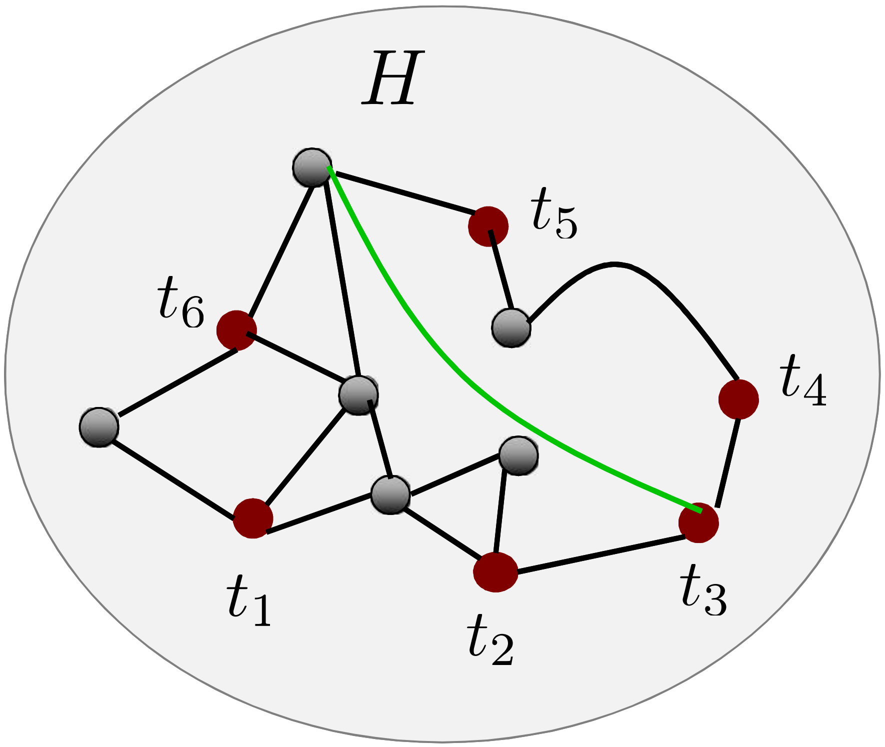
Splitting off operation at vertex v



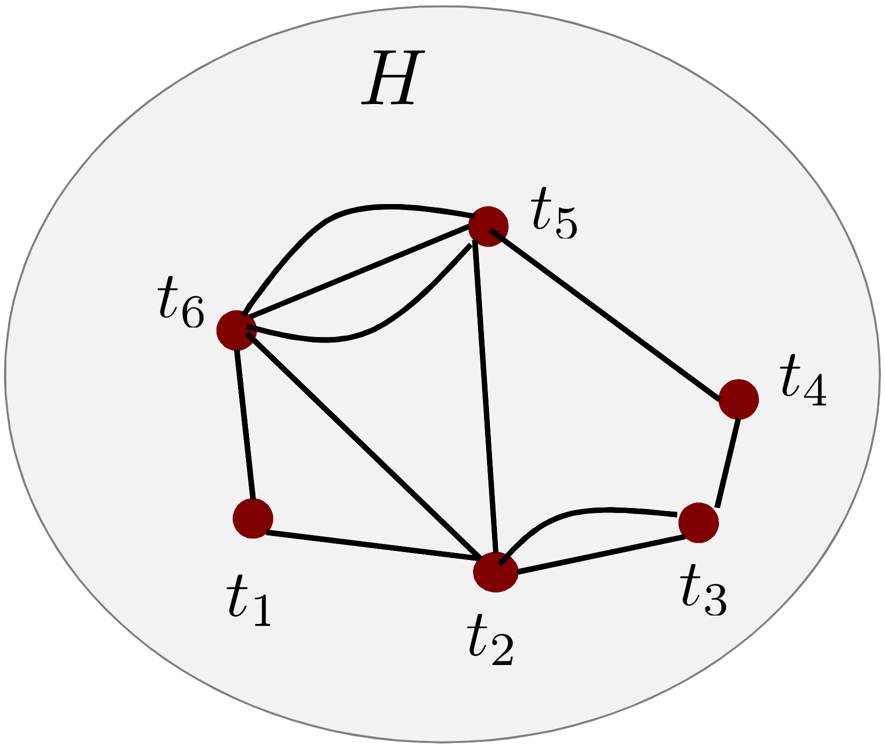
H
t_5
t_6
v
t_4
t_1
t_2
t_3



H
t_5
t_6
t_4
t_1
t_2
t_3



H
t_5
t_6
t_4
t_1
t_2
t_3



- Every edge in new graph is a path in the old graph.
- These paths are edge-disjoint.
- Degree of each t_i vertex remains unchanged.
- Edge-connectivity between the t_i vertices is preserved.

Mader's Theorem

Mader's Theorem: Given any undirected graph G and a vertex v of degree not equal to 3 such that there is no cut-edge incident on v, there always exists a splittable pair of edges incident on v.

Corollary: Let H(V,E) be an Eulerian graph, and let (S,T) be any partition of V. Then one can create a new graph H'(T,E') such that H' preserves all pairwise edge connectivities between vertices in T.

Toughness of a Graph

The toughness $\tau(G)$ of a connected undirected graph G is defined as the ratio

$$\tau(G) = \min_S |S|/c(S)$$

where the minimum is taken over $c(S) > 1$.

- Toughness of a clique is defined to be infinite.
- Toughness of a star is $1/(n-1)$.
- Toughness of a cycle is 1.

Toughness and Bounded Degree Spanning Trees

There has been much work on understanding the connection between toughness and existence of low degree spanning trees and Hamiltonian cycles.

Theorem [Furer and Raghavachari '94]
In any connected graph G, one can find in poly-time a spanning tree T such that the maximum degree in T is bounded by $1/\tau(G) + 3$.

Next …

We will use Mader's theorem along with the connection between toughness and bounded degree spanning trees to find our routing trees.

Thank You!