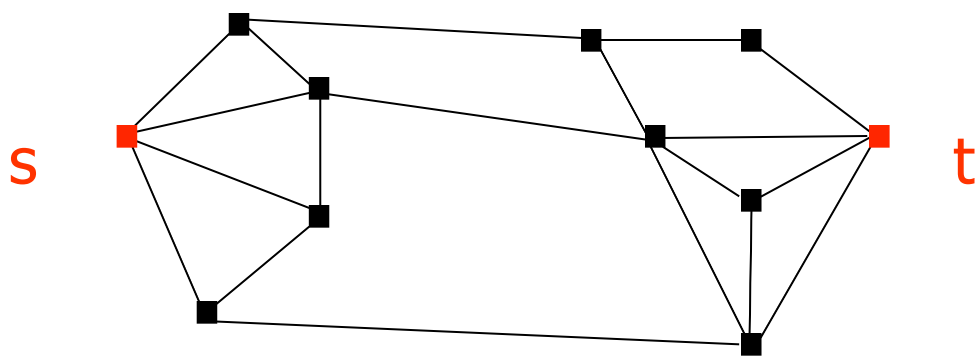
New algorithms for Disjoint Paths and Routing Problems

Chandra Chekuri
Dept. of Computer Science
Univ. of Illinois (UIUC)

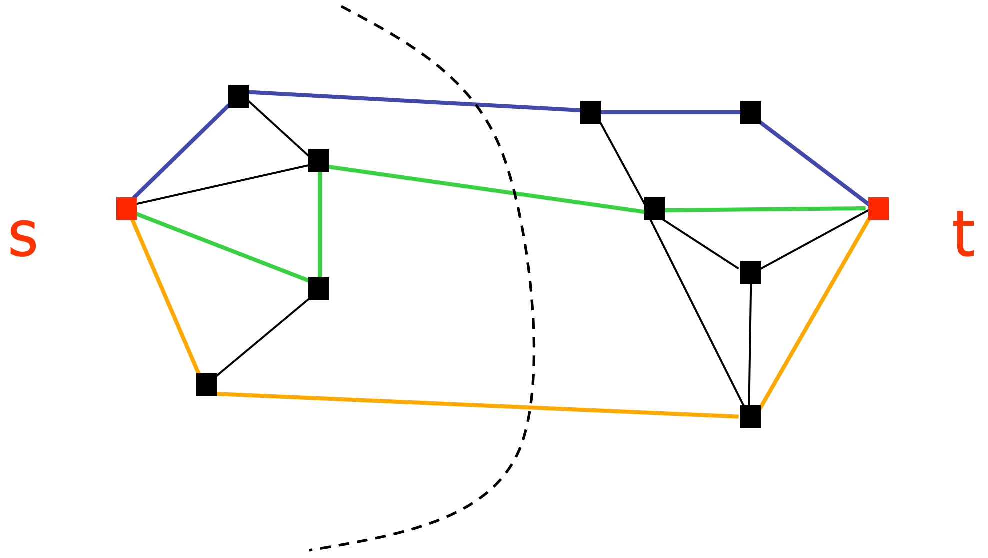
Menger’s Theorem

Theorem: The maximum number of s-t edge-disjoint paths in a graph G=(V,E) is equal to minimum number of edges whose removal disconnects s from t.



Menger's Theorem

Theorem: The maximum number of s-t edge-disjoint paths in a graph G=(V,E) is equal to minimum number of edges whose removal disconnects s from t.

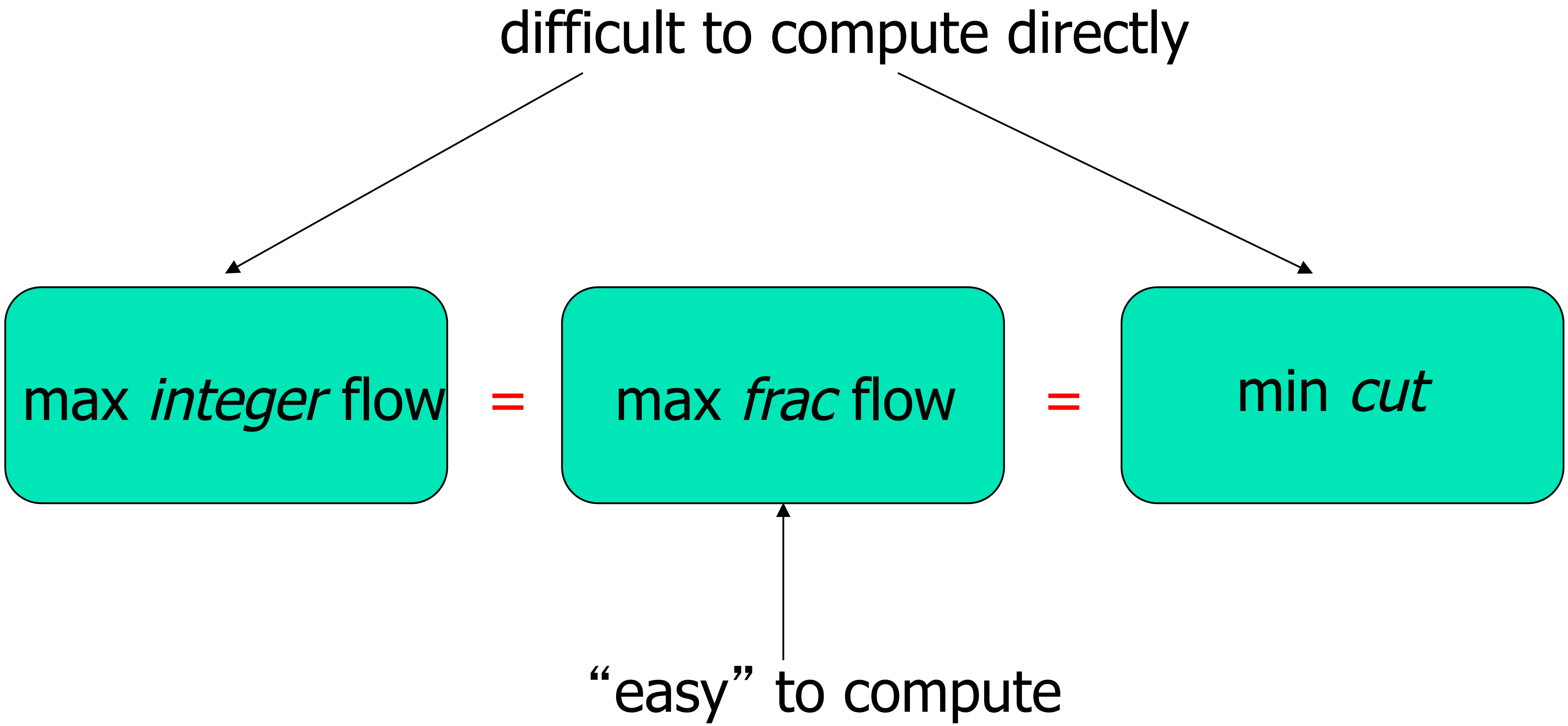


Max-Flow Min-Cut Theorem

[Ford-Fulkerson]

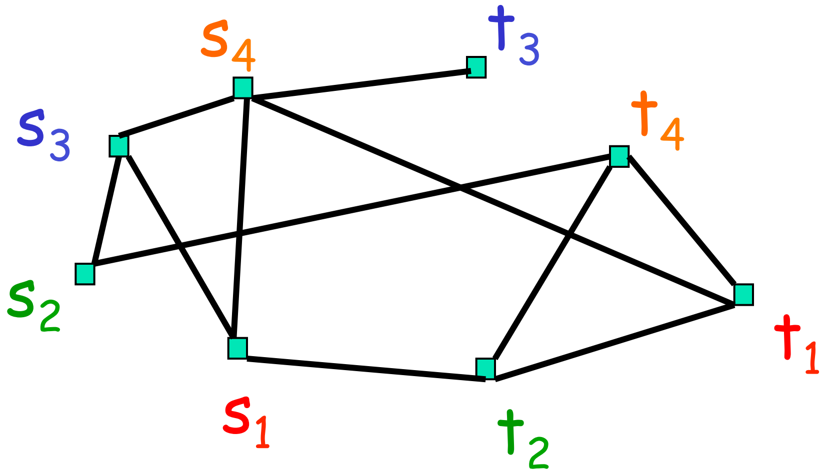
Theorem: The maximum s-t flow in an edge-capacitated graph G=(V,E) is equal to minimum s-t cut. If capacities are integer valued then max fractional flow is equal to max integer flow.

Computational view



Multi-commodity Setting

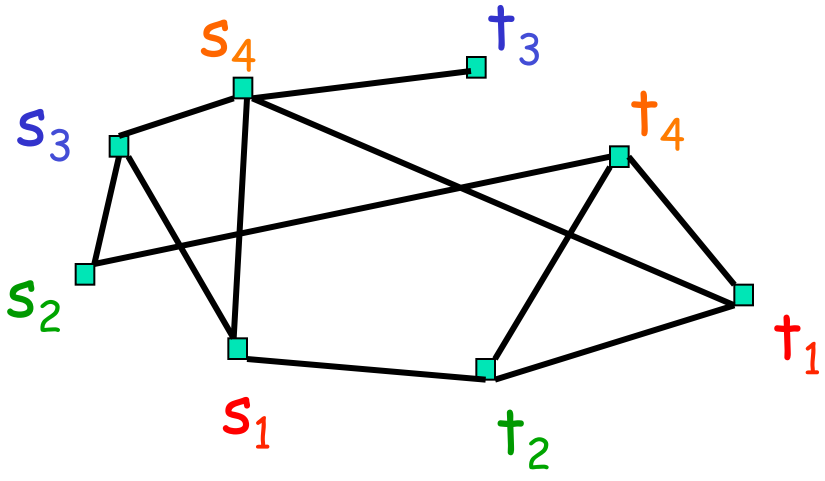
Several pairs: s_1t_1, s_2t_2,…, s_kt_k



Multi-commodity Setting

Several pairs: s_1t_1, s_2t_2,…, s_kt_k

Can *all* pairs be connected via edge-disjoint paths?

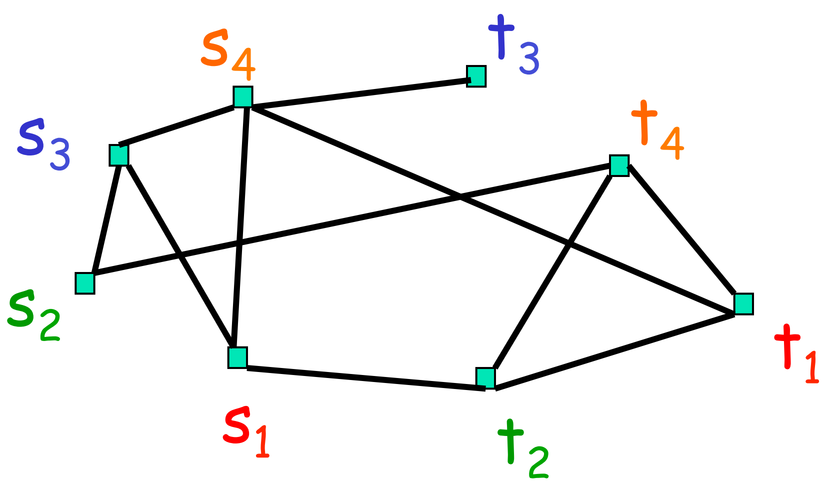


Multi-commodity Setting

Several pairs: s_1t_1, s_2t_2,…, s_kt_k

Can *all* pairs be connected via edge-disjoint paths?

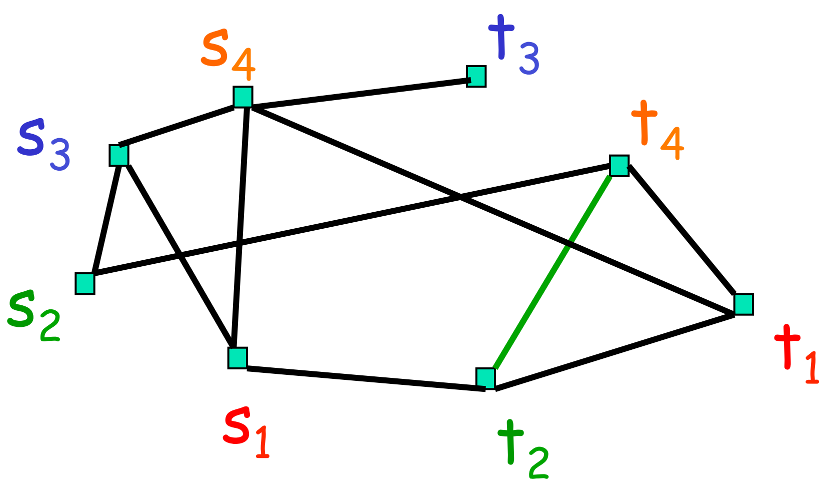
Maximize number of pairs that can be connected



Maximum Edge Disjoint Paths Prob

Input: Graph $G(V,E)$, node pairs s_1t_1, s_2t_2, ..., s_kt_k

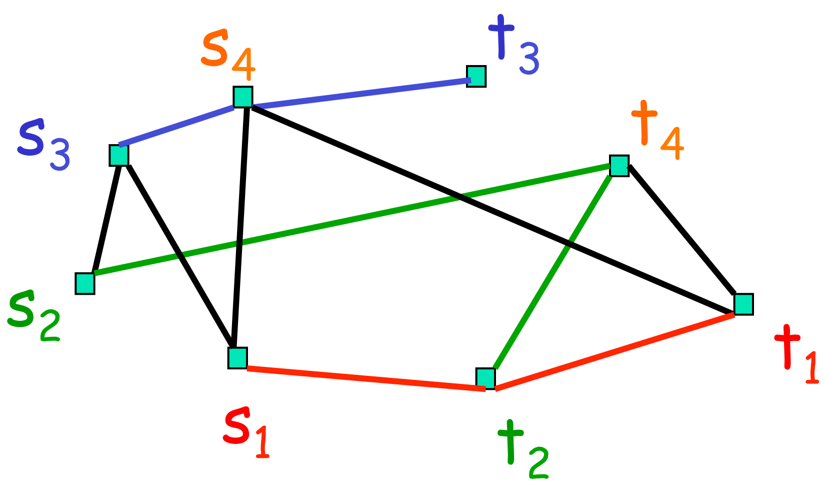
Goal: Route a maximum # of s_i-t_i pairs using *edge-disjoint* paths



Maximum Edge Disjoint Paths Prob

Input: Graph $G(V,E)$, node pairs s_1t_1, s_2t_2, ..., s_kt_k

Goal: Route a maximum # of s_i-t_i pairs using *edge-disjoint* paths



Motivation

Basic problem in combinatorial optimization

Applications to VLSI, network design and routing, resource allocation & related areas

Related to significant theoretical advances

- Graph minor work of Robertson & Seymour
- Randomized rounding of Raghavan-Thompson
- Routing/admission control algorithms

Computational complexity of MEDP

Directed graphs: 2-pair problem is NP-Complete
[Fortune-Hopcroft-Wylie' 80]

Undirected graphs: for any fixed constant k, there is a polynomial time algorithm
[Robertson-Seymour' 88]

NP-hard if k is part of input

Approximation

Is there a good *approximation algorithm*?

- polynomial time algorithm
- for *every* instance I returns a solution of value at least $OPT(I)/\alpha$ where α is approx ratio

How useful is the flow *relaxation*?

- What is its integrality gap?

Current knowledge

- If P $\neq$ NP, problem is hard to approximate to within polynomial factors in *directed graphs*
- In undirected graphs, problem is quite open
  - upper bound - $O(n^{1/2})$ [C-Khanna-Shepherd' 06]
  - lower bound - $\Omega(\log^{1/2-\varepsilon} n)$ [Andrews etal' 06]
- Main approach is via flow relaxation

Current knowledge

- If P $\neq$ NP, problem is hard to approximate to within polynomial factors in *directed graphs*
- In undirected graphs, problem is quite open
  - upper bound - $O(n^{1/2})$ [C-Khanna-Shepherd' 06]
  - lower bound - $\Omega(\log^{1/2-\varepsilon} n)$ [Andrews etal' 06]
- Main approach is via flow relaxation

Rest of talk: focus on *undirected* graphs

Flow relaxation

For each pair $s_i t_i$ allow fractional flow $x_i \in [0,1]$

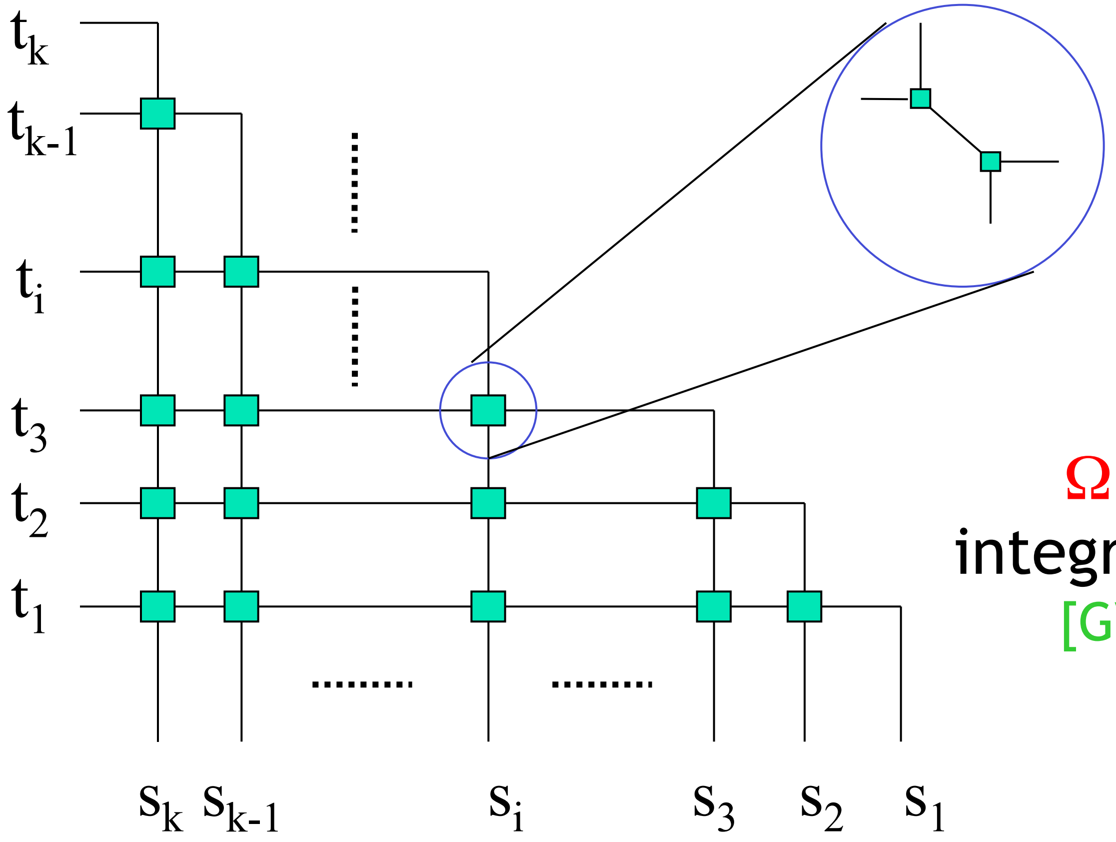
Flow for each pair can use *multiple* paths

Total flow for all pairs on each edge e is ≤ 1

Total fractional flow = $\sum_i x_i$

Relaxation can be solved in polynomial time using linear programming (faster approximate methods also known)

Example



$\Omega(n^{1/2})$
integrality gap
[GVY 93]

max integer flow = 1, max fractional flow = k/2

Overcoming integrality gap

Two approaches:

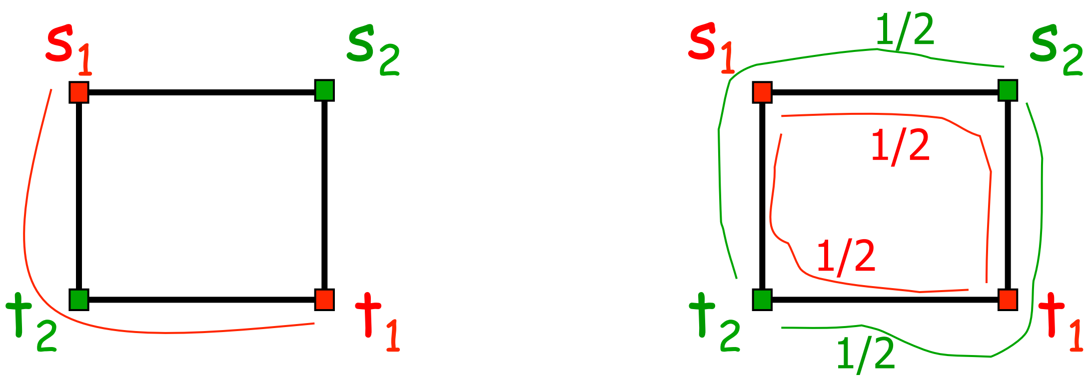
- Allow some small congestion c
  - up to c paths can use an edge

Overcoming integrality gap

Two approaches:

- Allow some small congestion c
  - up to c paths can use an edge
  - c=2 is known as half-integer flow path problem
- All-or-nothing flow problem
  - $s_i t_i$ is routed if *one unit of flow* is sent for it (can use multiple paths) [C.-Mydlarz-Shepherd' 03]

Example



Prior work on approximation

Greedy algorithms or randomized rounding of flow

- polynomial approximation ratios in general graphs. $O(n^{1/c})$ with congestion c
- better bounds in various special graphs: trees, rings, grids, graphs with high expansion

No techniques to take advantage of relaxations: congestion or all-or-nothing flow

New framework

[C-Khanna-Shepherd]

New framework to understand flow relaxation

Framework allows near-optimal approximation algorithms for planar graphs and several other results

Flow based relaxation is much better than it appears

New connections, insights, and questions

Some results

OPT: optimum value of the flow relaxation

Theorem: In *planar graphs*

- can route $\Omega(OPT/\log n)$ pairs with $c=2$ for both edge and node disjoint problems
- can route $\Omega(OPT)$ pairs with $c=4$

Theorem: In any graph $\Omega(OPT/\log^2 n)$ pairs can be routed in all-or-nothing flow problem.

Flows, Cuts, and Integer Flows

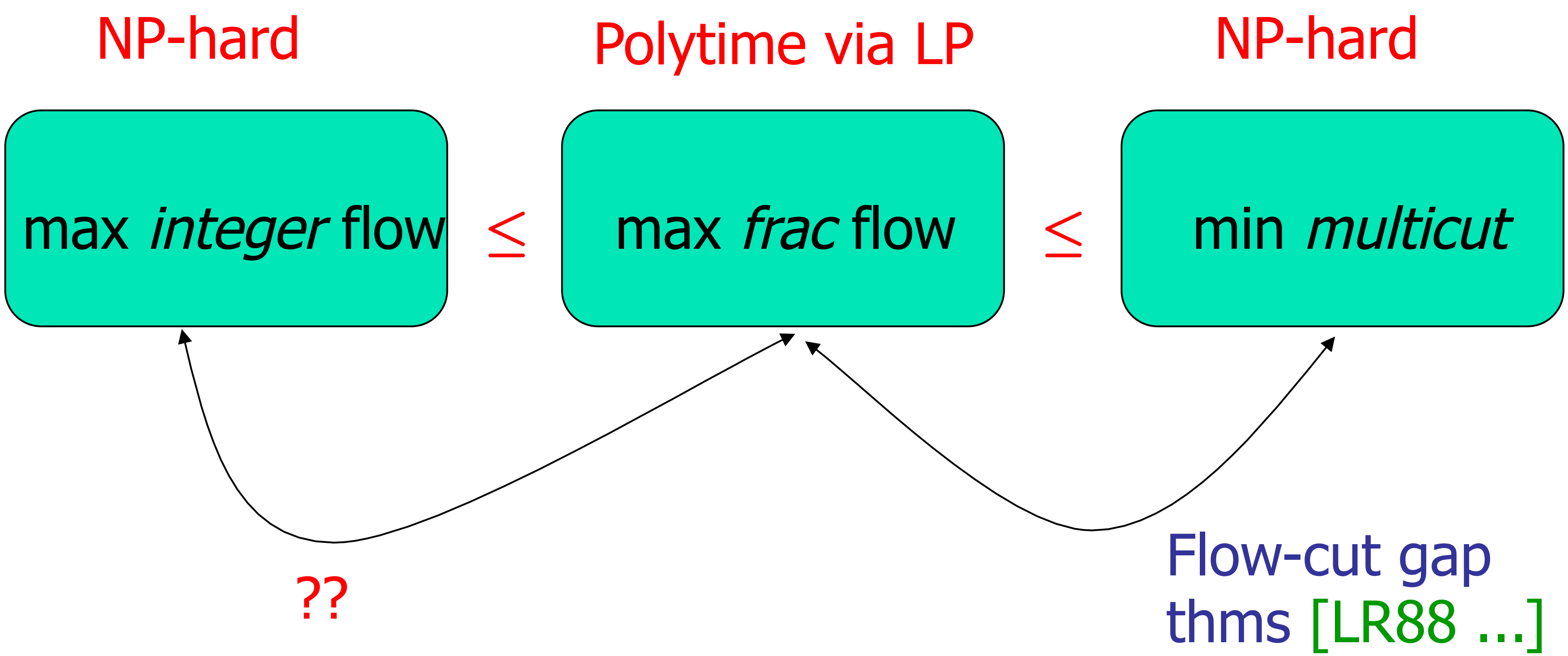
Multicommodity: several pairs

NP-hard | Polytime via LP | NP-hard

max *integer* flow $\leq$ max *frac* flow $\leq$ min *multicut*

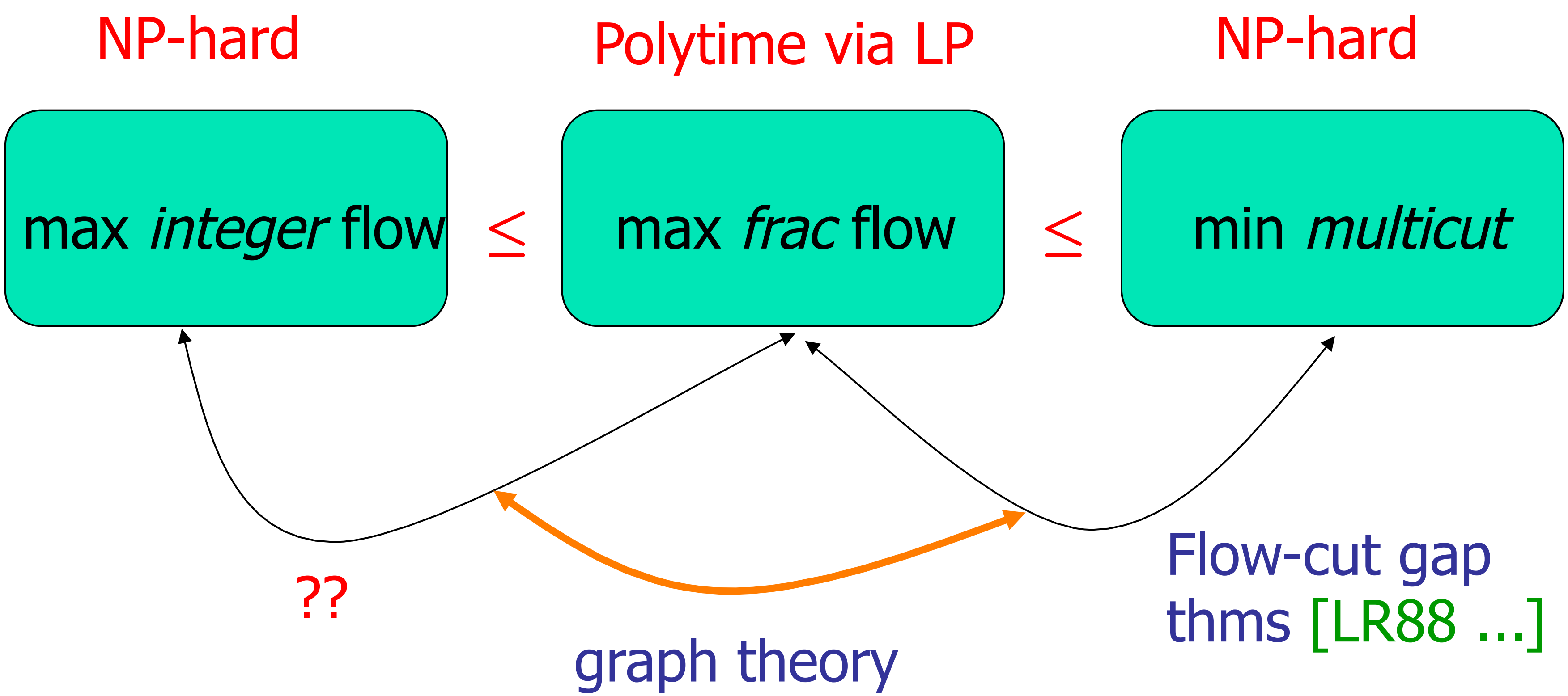
Flows, Cuts, and Integer Flows

Multicommodity: several pairs



Flows, Cuts, and Integer Flows

Multicommodity: several pairs



Part II: Details

New algorithms for routing

1. Compute maximum fractional flow

2. Use fractional flow solution to decompose input instance into a collection of well-linked instances.

3. Well-linked instances have nice properties – exploit them to route

Some simplifications

Input: undir graph $G=(V,E)$ and pairs $s_1t_1,\ldots, s_kt_k$

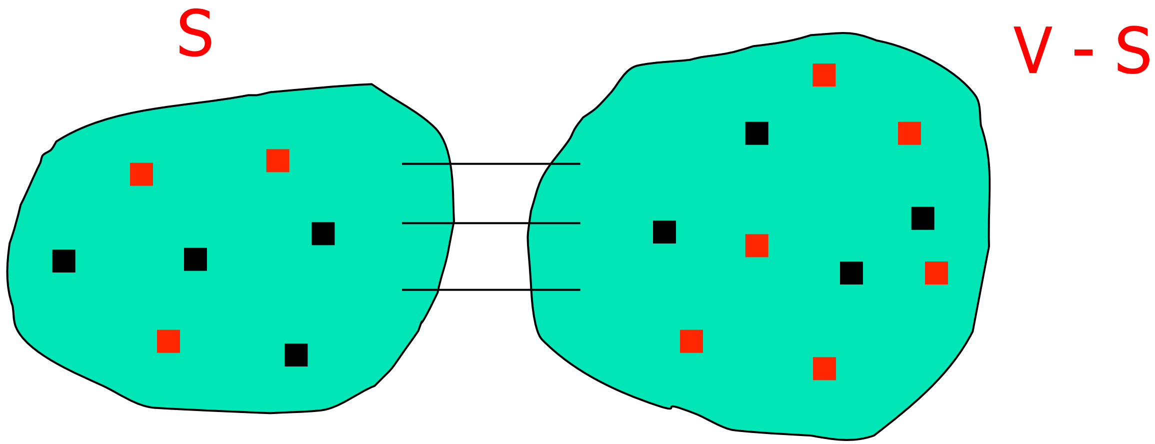
$X = \{s_1, t_1, s_2, t_2, \ldots, s_k, t_k\}$ -- *terminals*

Assumption: wlog each terminal in only one pair

Instance: (G, X, M) where M is matching on X

Well-linked Set

Subset X is *well-linked* in G if for *every* partition (S,V-S) , # of edges cut is at least # of X vertices in smaller side



for all $S \subset V$ with $|S \cap X| \leq |X|/2$, $|\delta(S)| \geq |S \cap X|$

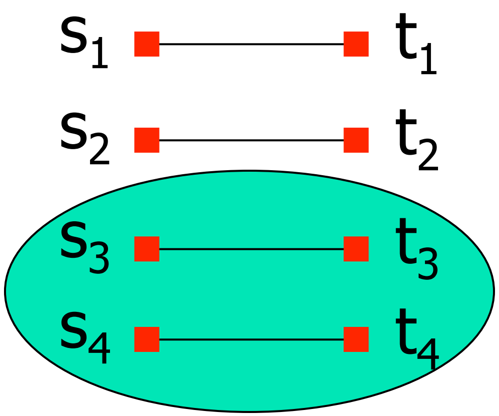
Well-linked instance of EDP

Input instance: (G, X, M)

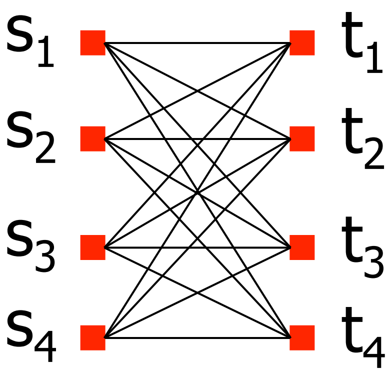
$X = \{s_1, t_1, s_2, t_2, \ldots, s_k, t_k\}$ – terminal set

Instance is well-linked if X is well-linked in G

Examples



Not a well-linked instance



A well-linked instance

New algorithms for routing

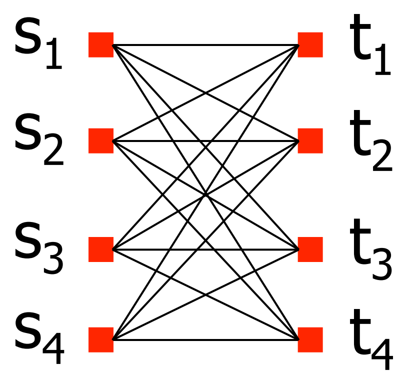
1. Compute maximum fractional flow

2. Use fractional flow solution to decompose input instance into a collection of well-linked instances.

3. Well-linked instances have nice properties – exploit them to route

Advantage of well-linkedness

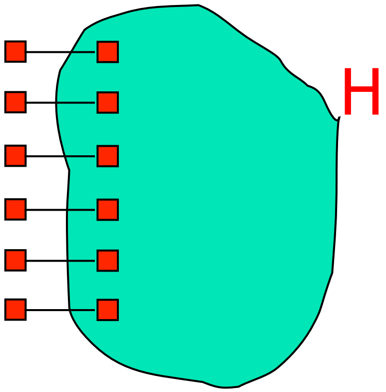
LP value does not depend on input matching M



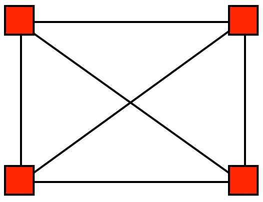
Theorem: If X is well-linked, then for *any* matching on X, LP value is $\Omega(|X|/\log|X|)$. For planar G, LP value is $\Omega(|X|)$

Crossbars

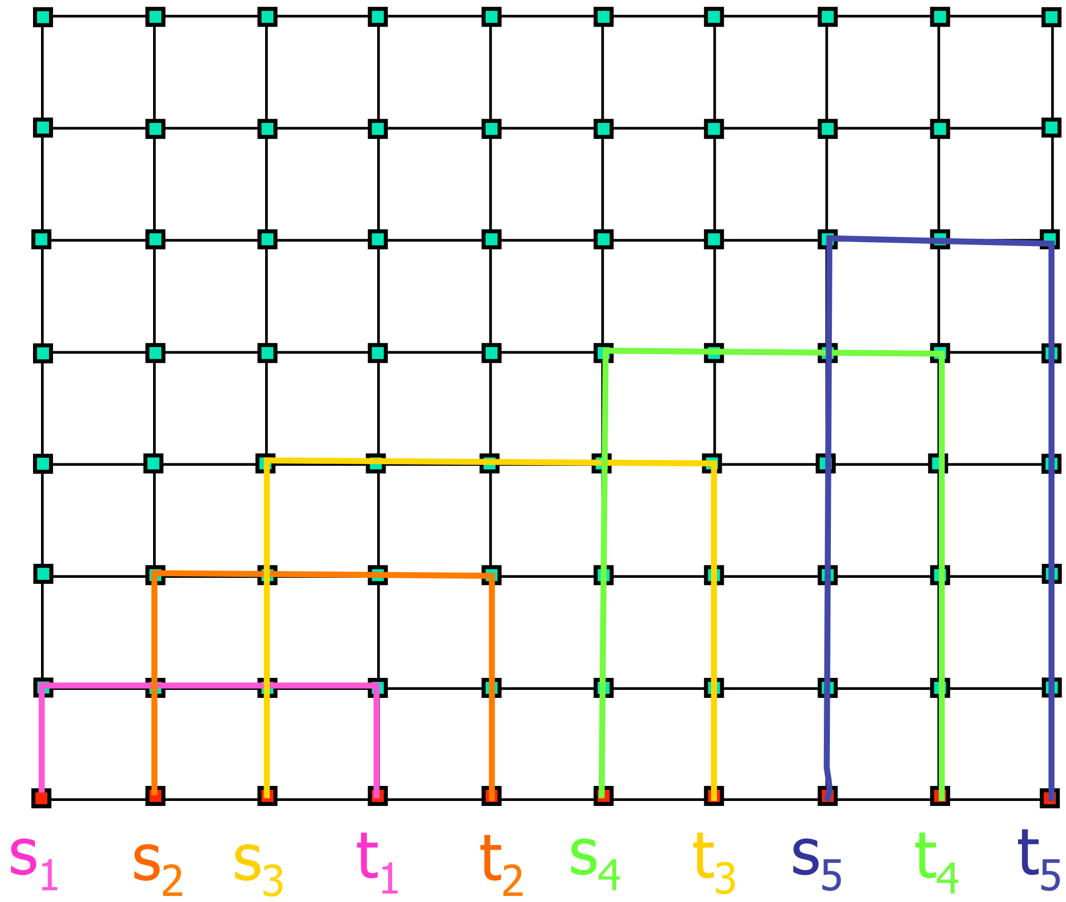
H=(V,E) is a *cross-bar* with respect to an *interface* I $\subseteq$ V if any matching on I can be routed using edge-disjoint paths



Ex: a complete graph is a cross-bar with I=V



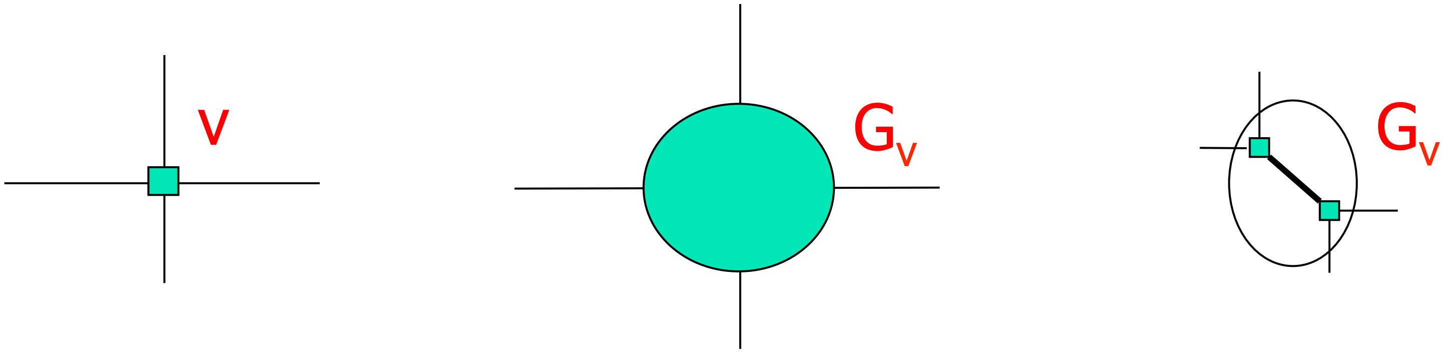
Grids as crossbars



First row is *interface*

Grids in Planar Graphs

Theorem[RST94]: If G is *planar graph* with *treewidth* h, then G has a *grid minor* of size $\Omega(h)$ as a subgraph.



Grid minor is crossbar with congestion 2

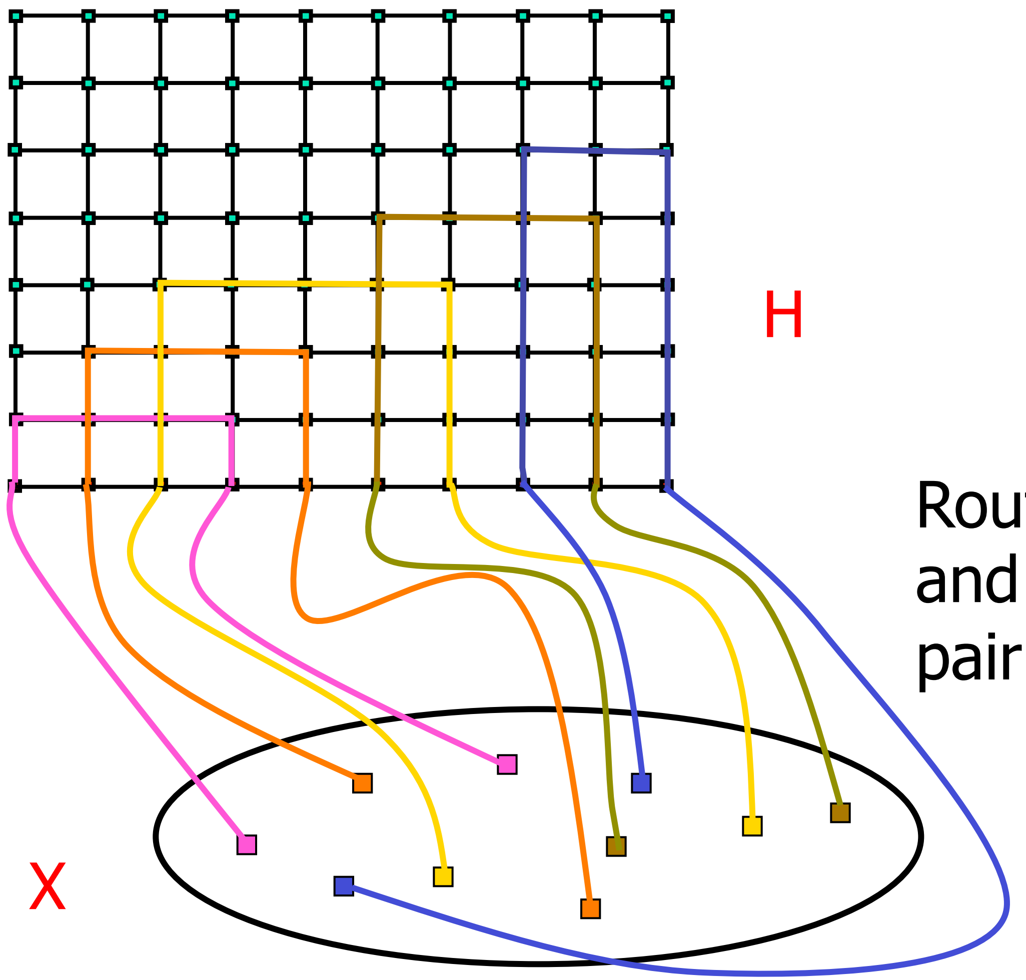
Back to Well-linked sets

Claim: X is *well-linked* implies treewidth = $\Omega(|X|)$

X *well-linked* $\Rightarrow$ G has grid minor H of size $\Omega(|X|)$

Q: how do we route $M = (s_1t_1, \ldots, s_kt_k)$ using H ?

Routing pairs in X using H



Route X to I and use H for pairing up

Several technical issues

- What if X cannot reach H?
- H is smaller than X, so can *pairs* reach H?
- Can X reach H without using edges of H?
- Can H be found in polynomial time?

General Graphs?

Grid-theorem extends to graphs that exclude a fixed minor [RS, DHK' 05]

For general graphs, need to prove following:

Conjecture: If G has treewidth h then it has an *approximate crossbar* of size $\Omega(h/\text{polylog}(n))$

Crossbar $\Leftrightarrow$ LP relaxation is good

Reduction to Well-linked case

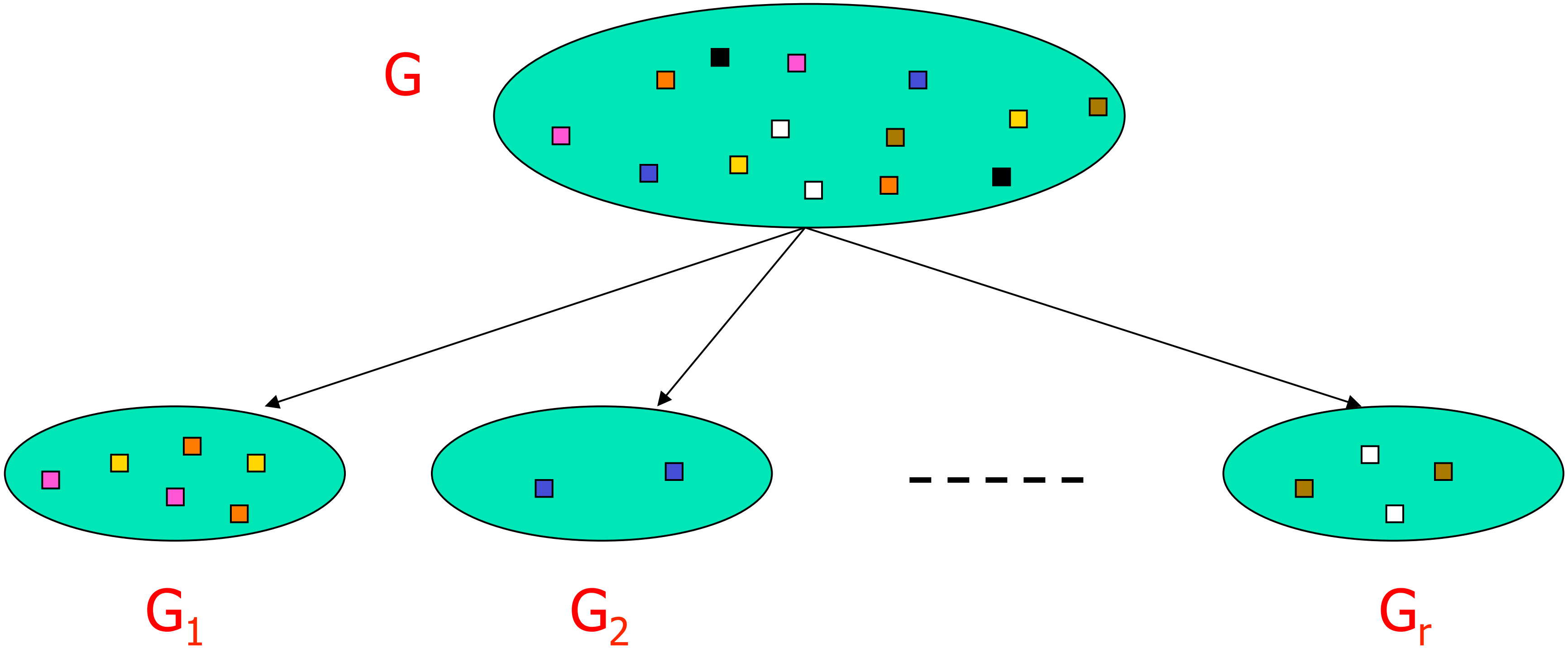
Given G and k pairs $s_1t_{1,}$ $s_2t_{2,}$ ... s_kt_k

$X = \{s_1,t_1, s_2, t_2, \ldots, s_k, t_k\}$

We know how to solve problem if X is well-linked

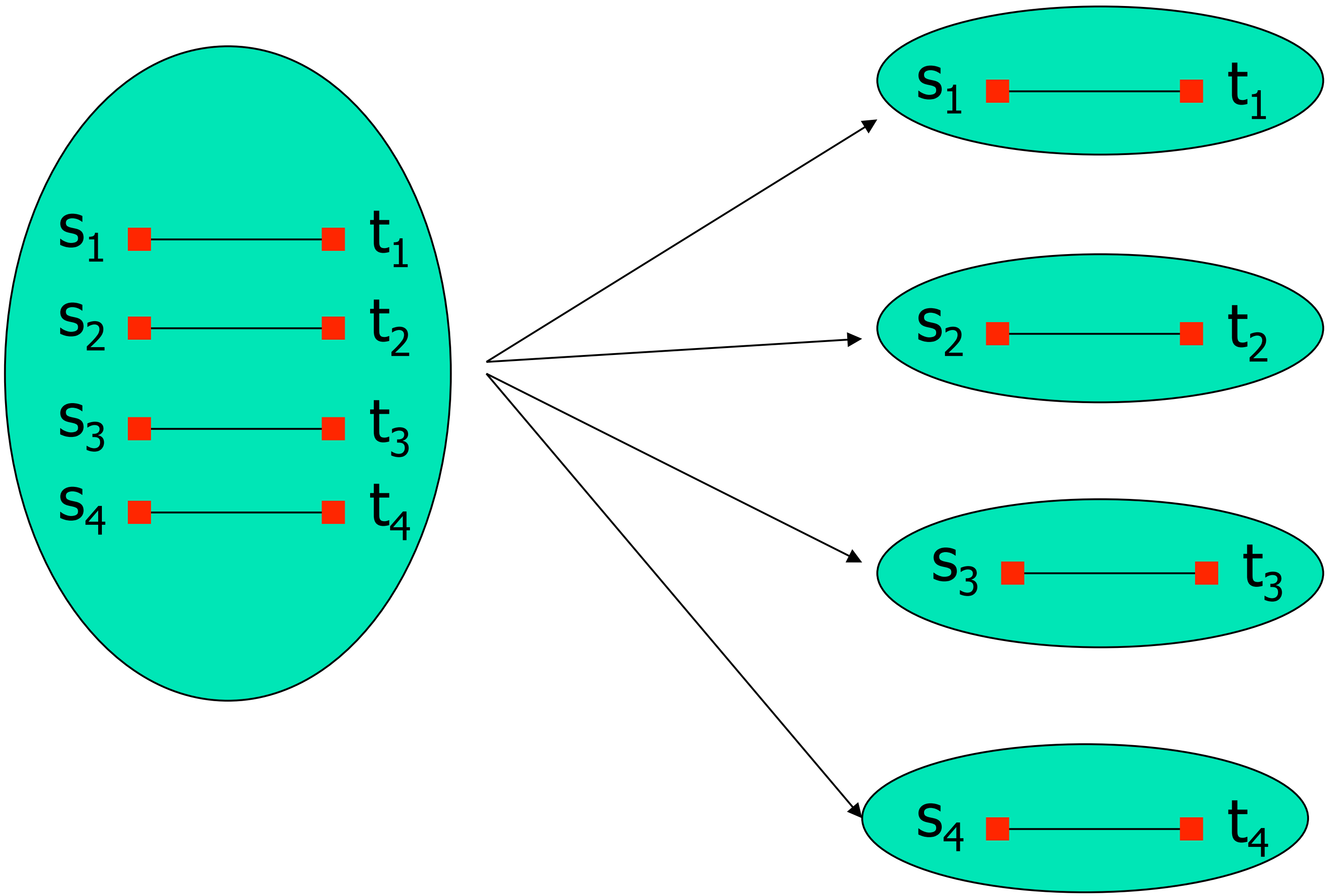
Q: can we reduce general case to well-linked case?

Decomposition



X_i is well-linked in G_i $\sum_i |X_i| \geq OPT/\beta$

Example



Decomposition

$\beta = O(\log k\ \gamma)$ where γ = worst gap between flow and cut

- $\gamma = O(\log k)$ using [Leighton-Rao' 88]
- $\gamma = O(1)$ for planar graphs [Klein-Plotkin-Rao' 93]

Decomposition based on LP solution

Recursive algorithm using *separator* algorithms

Need to work with *approximate* and *weighted* notions of well-linked sets

Decomposition Algorithm

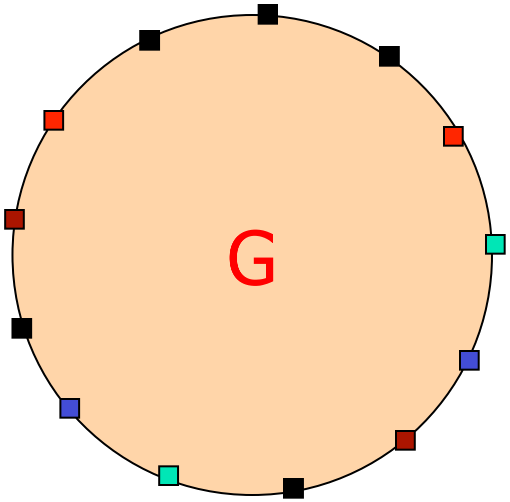
Weighted version of well-linkedness

- each $v \in X$ has a weight
- weight determined by LP solution
- weight of s_i and t_i equal to x_i the flow in LP soln

- X is well-linked *implies* no *sparse cut*
- If sparse cut exists, break the graph into two
- Recurse on each piece
- Final pieces determine the decomposition

OS instance

Planar graph G, all terminals on single (outer) face



Okamura-Seymour Theorem: If all terminals lie on a single face of a planar graph then the cut-condition implies a *half-integral* flow.

Decomposition into OS instances

Given instance (G,X,M) on planar graph G, algorithm to decompose into OS-type instances with only a constant factor loss in value

Contrast to well-linked decomposition that loses a log n factor

Using OS-decomposition and several other ideas, can obtain O(1) approx using c=4

Conclusions

- New approach to disjoint paths and routing problems in undirected graphs
- Interesting connections including new proofs of flow-cut gap results via the “primal” method
- Several open problems
  - Crossbar conjecture: a new question in graph theory
  - Node-disjoint paths in planar graphs - $O(1)$ approx with $c = O(1)$?
  - Congestion minimization in planar graphs. $O(1)$ approximation?

Thanks!