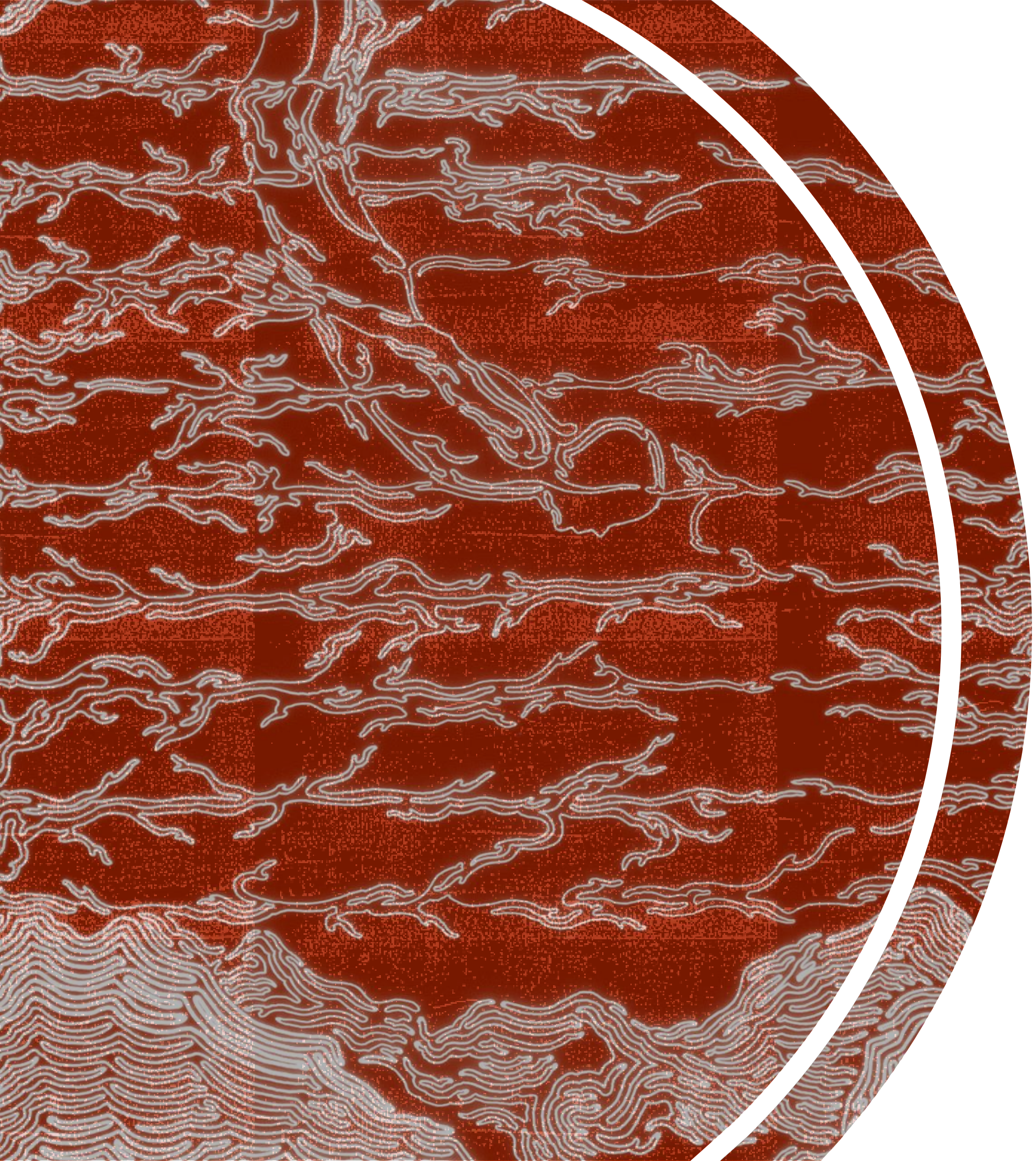


INTRODUCTION TO COMPUTATIONAL TOPOLOGY

HSIEN-CHIH CHANG

LECTURE 9, OCTOBER 12, 2021

Administrivia

- Homework 3 is out, due 10/25 (Mon)

- Optional Final Project:
  - Project proposal is due 10/18 (Mon)
  - Presentation during finals week (likely to be 11/23 (Tue))
  - Project report due 11/29 (Mon)

Minimum Cut in Planar Graphs

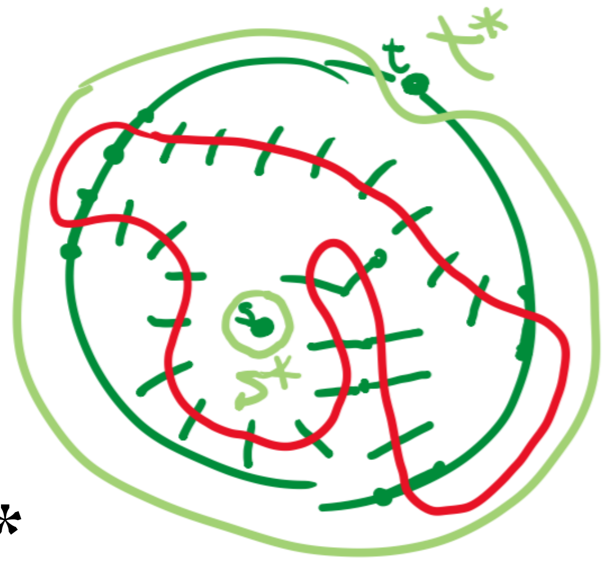
Minimum Cut in a Graph

- Given undirected graph G with positive edge-weights and two vertices s and t, find a minimum-weight edge cut separating s and t

Minimum Cut in Planar Graph

- Given undirected planar graph G with positive edge-weights and two vertices s and t, find a minimum-weight edge cut separating s and t

{edge cuts} ⇔ {circuit = union of cycles}



min (s,t)-cut ⇔ minimum cycle separating s* and t*

Find a Min Homotopic Cycle!

Observations

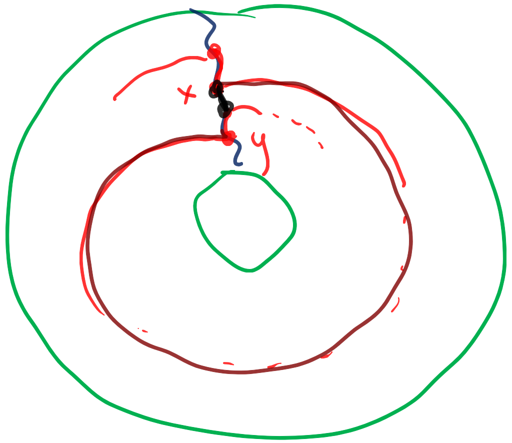
- Shortest cycle C must pass through any path π from s^* to t^*
- Cycle C intersects π at one segment if π is shortest

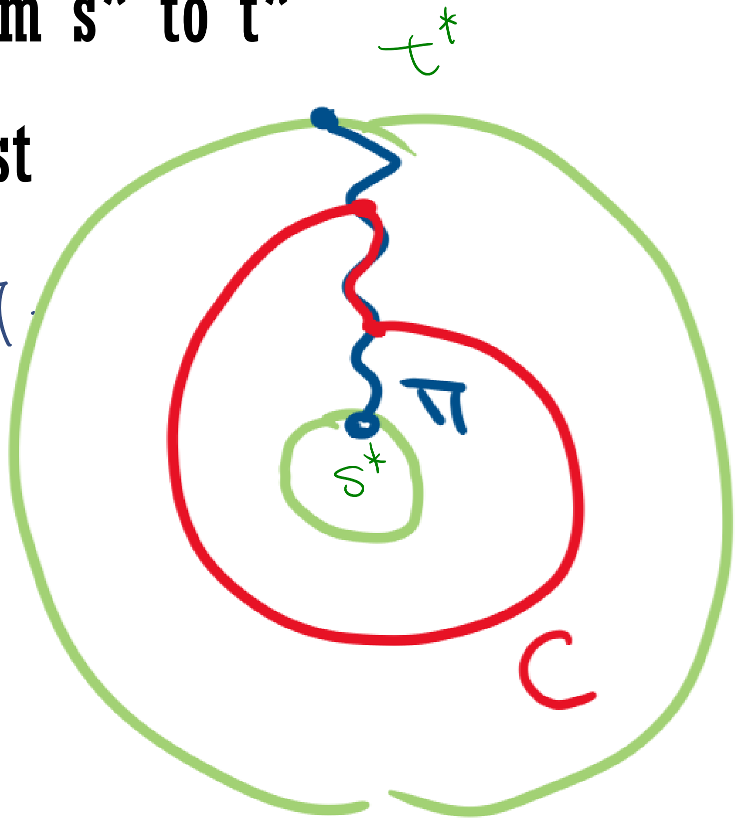
Claim: If π is shortest, then $C \cap \pi$ at a segment.

pf. $C \cap \pi$ at ≥ 2 segments.

Replace $C[x..y]$

with $\pi[x..y]$



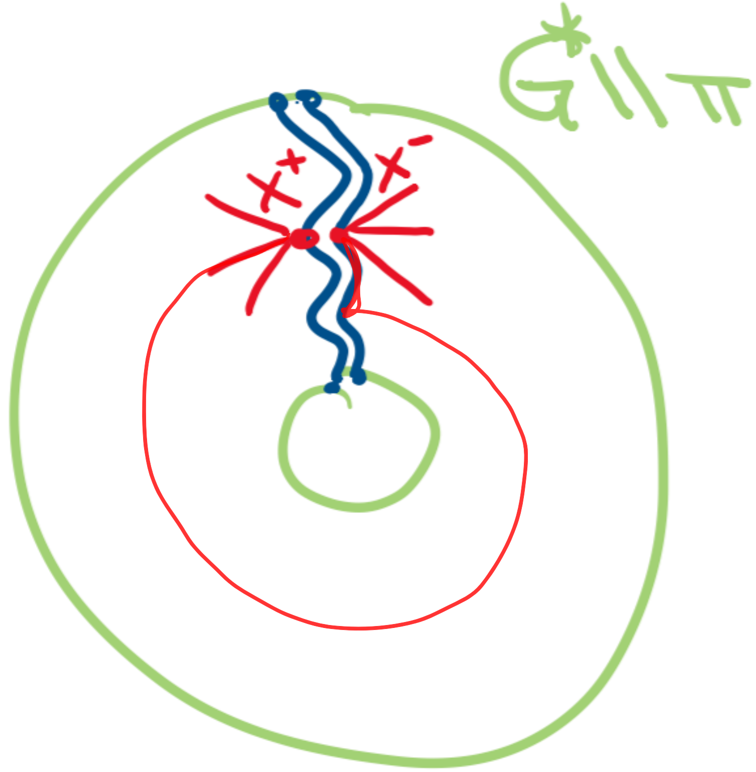


Naïve Algorithm

MinCut(G, s, t):

Find shortest path π from $s^* \rightsquigarrow t^*$
Cut open G^* along π.
for each vertex x on π:
    find shortest path $x^+ \rightsquigarrow x^-$

Return length of min$\{x^+ \rightsquigarrow x^-\}$

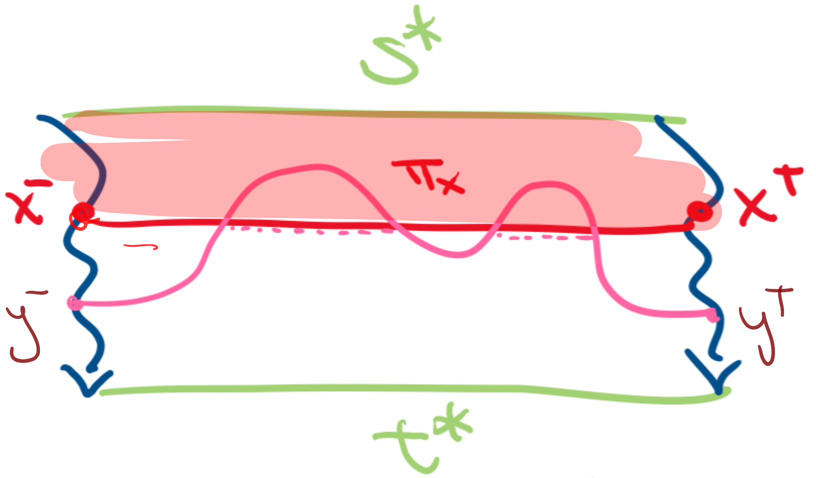


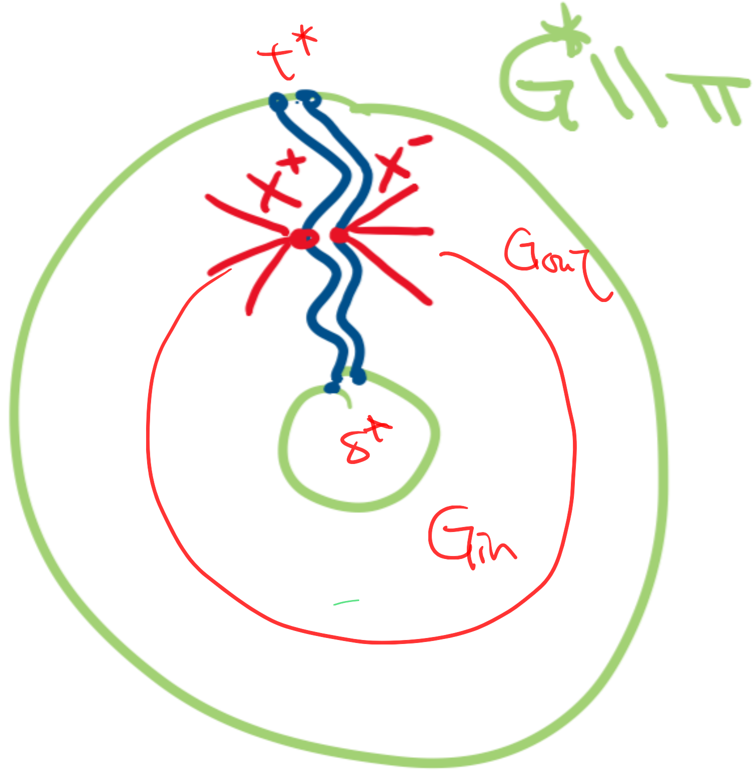
Reif's Algorithm

[Reif 1983]

MinCut(G, s, t):

- Find shortest path π from $s^* \rightsquigarrow t^*$
- Cut open G^* along π.
- for ~~each~~ middle vertex x on π:
  - find shortest path $\pi_x : x^+ \rightsquigarrow x^-$
  - MinCut(G_{in} , s^*, π_x^*)
  - MinCut(G_{out} , π_x^*, t^*)
- Return length of min$\{x^+ \rightsquigarrow x^-\}$





Improved Algorithms for Min Cut and Max Flow in Undirected Planar Graphs

Giuseppe F. Italiano [*]
Dipartimento di Informatica, Sistemi e Produzione
University Rome "Tor Vergata"
Rome, Italy
italiano@disp.uniroma2.it

Yahav Nussbaum [†]
The Blavatnik School of Computer Science
Tel Aviv University
Tel Aviv, Israel
yahav.nussbaum@cs.tau.ac.il

Piotr Sankowski [‡]
Institute of Informatics
University of Warsaw
Warsaw, Poland
sank@mimuw.edu.pl

Christian Wulff-Nilsen [§]
School of Computer Science
Carleton University
Ottawa, Canada
koolooz@diku.dk

ABSTRACT

We study the min *st*-cut and max *st*-flow problems in planar graphs, both in static and in dynamic settings. First, we present an algorithm that given an undirected planar graph and two vertices s and t computes a min st-cut in $O(n \log\log n)$ time. Second, we show how to achieve the same bound for the problem of computing a max *st*-flow

Categories and Subject Descriptors

G.2.2 [**Graph Theory**]: Graph algorithms

General Terms

Algorithms,Theory

Faster Planar Min-cut [Italiano-Nussbaum-Sankowski-Wulff-Nilsen 2011]

Planar min-cut can be computed in O(n loglog n) time

High-Level Ideas

$$T(n,\pi) \le \underbrace{O(n)}_{O(n\log\log n)} + \sum_{i=1}^{f(n)} T\left(n_i, \frac{\pi}{f(n)}\right)$$

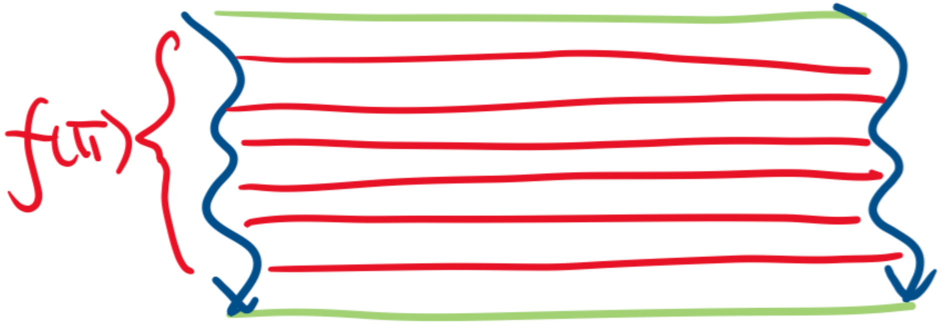
$$T(n,\pi) = O(n \cdot F^*(\pi)) \qquad F(\pi) := \frac{\pi}{f(n)}$$

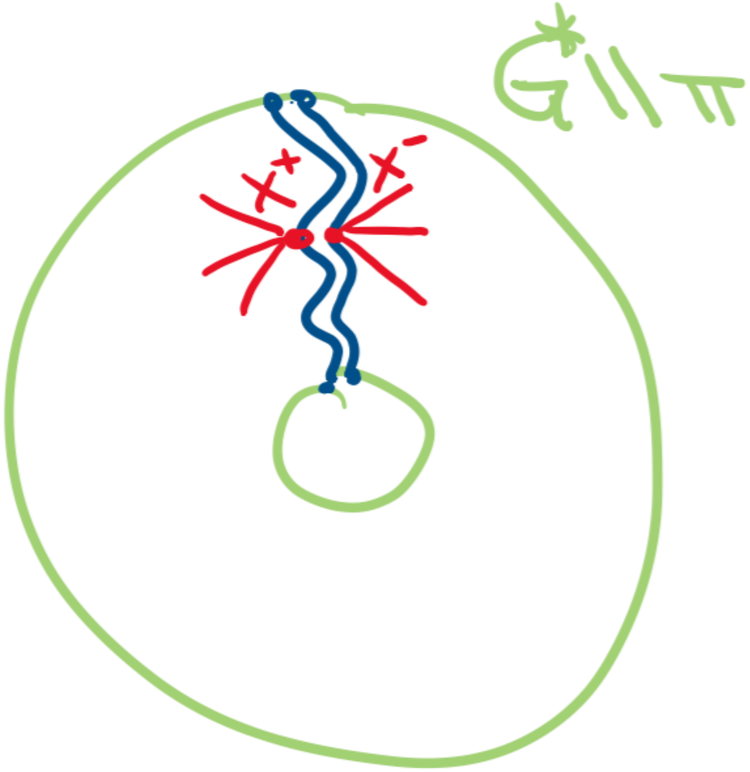
$$F^*(n) := \#\text{times } F^{(d)}(n) \le C$$

$$F(n) = n-2, \quad F^*(n) = n/2$$

$$F(n) = n/2, \quad F^*(n) = \log_2 n$$

$$F(n) = \log n, \quad F^*(n) = \log^* n$$





Toolbox to be Built

- Multiple-source shortest paths [Klein 2005] [Cabello-Chambers-Erickson 2013]
- Cycle separator decomposition/r-division [Frederickson 1989] [Klein-Mozes-Sommer 2012]
- Monge heap/dense distance graph [Aggarwal-Klawe-Moran-Shor-Wilber 1987]
- FR-Dijkstra [Fakcharoenphol-Rao 2001]

- Monge emulator [Chang-Ophelders 2020] [Chang-Krauthgamer-Tan 2022]

Intermission

Philosophical Question:

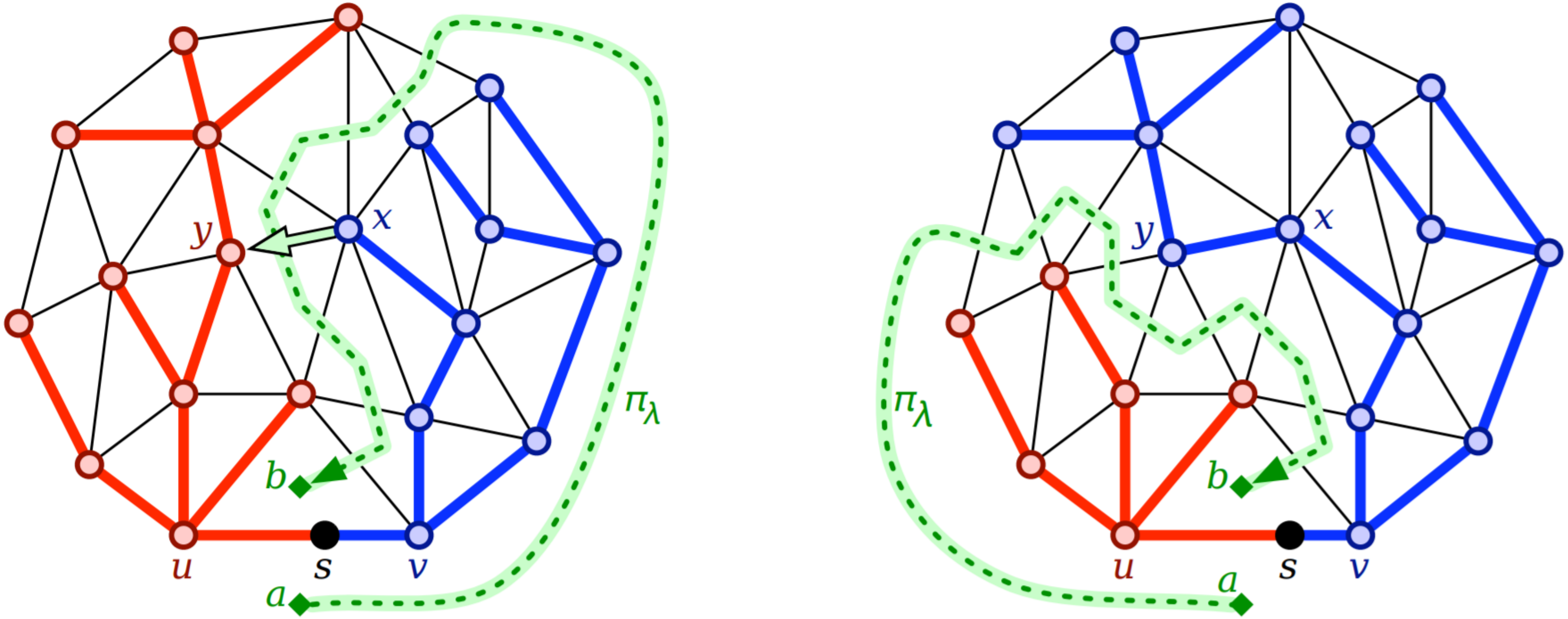
Why do we care about shaving logs?

Multiple-Source Shortest Paths

MSSP Problem Definition

- Given a planar directed graph G with and sources all on the outer-face, and edges weights w: E(G) $\longrightarrow$ R_{+}

- Compute shortest paths between every source s and every vertex x
  - Represented implicitly

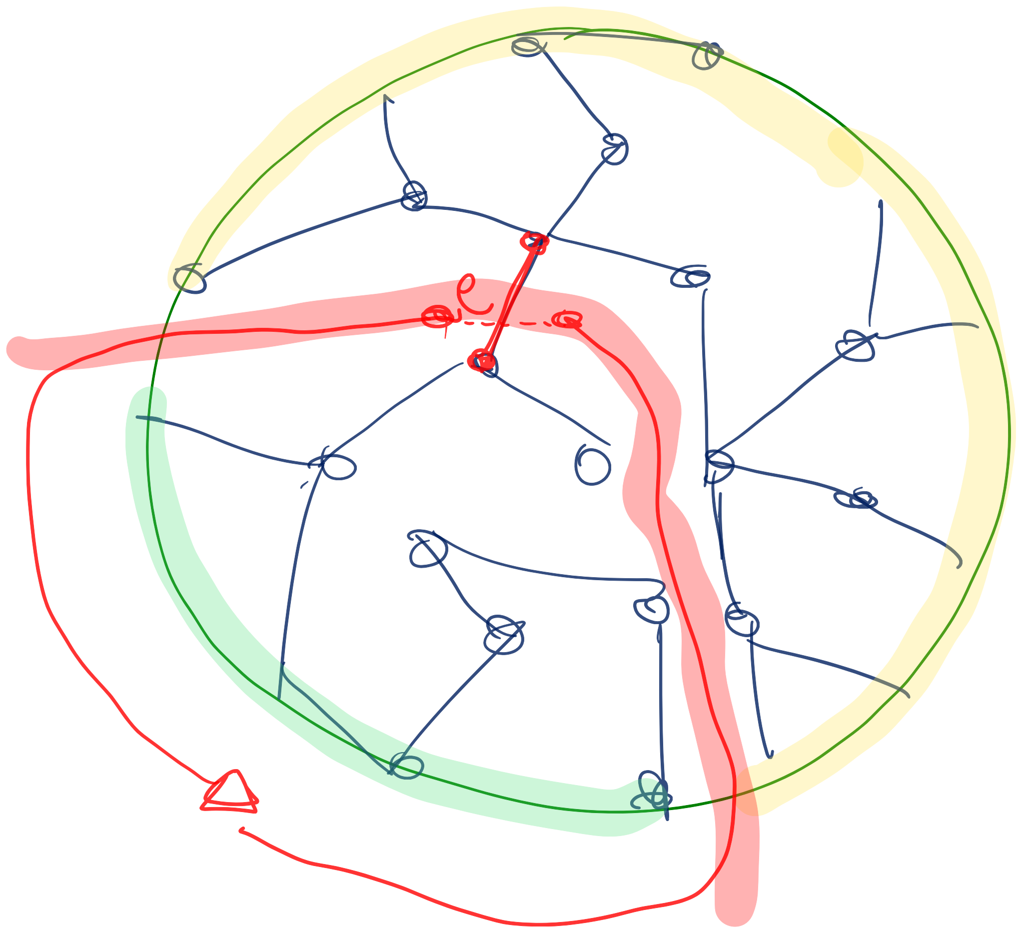


Multiple-Source Shortest Paths

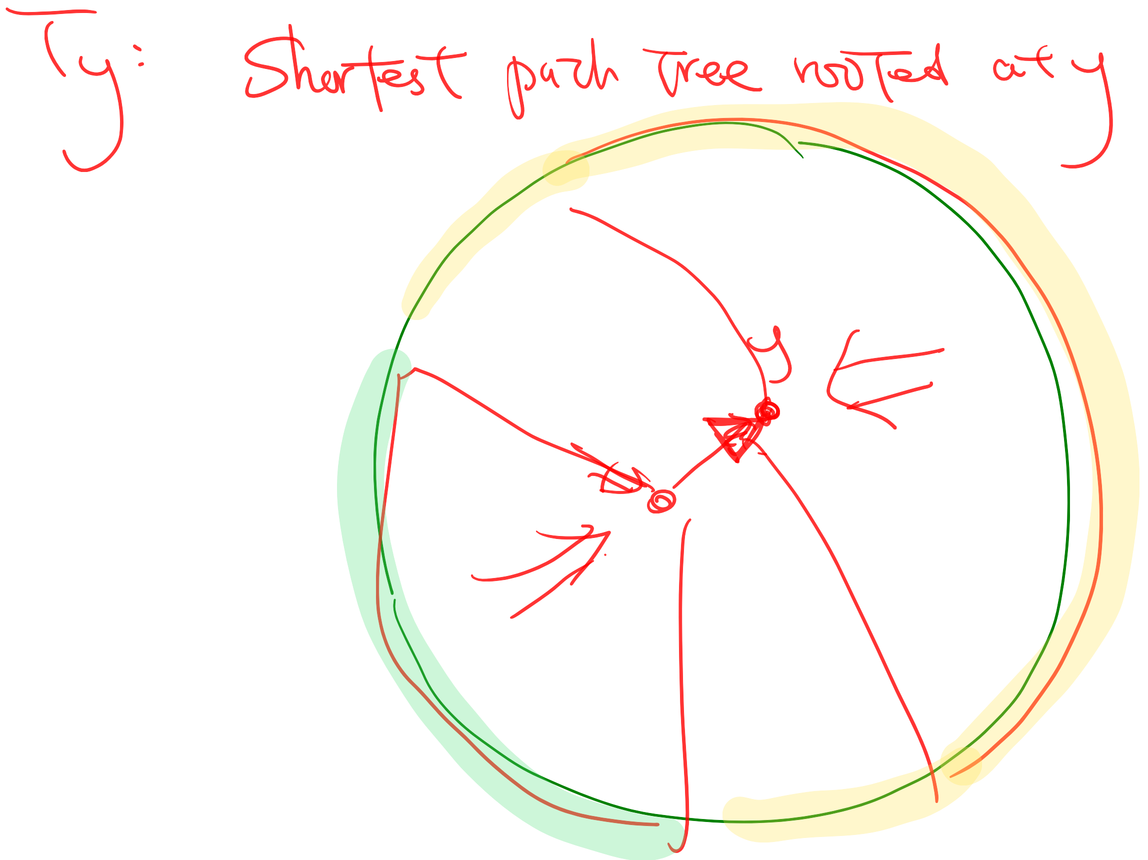
[Klein 2005]
[Cabello-Chambers-Erickson 2013]

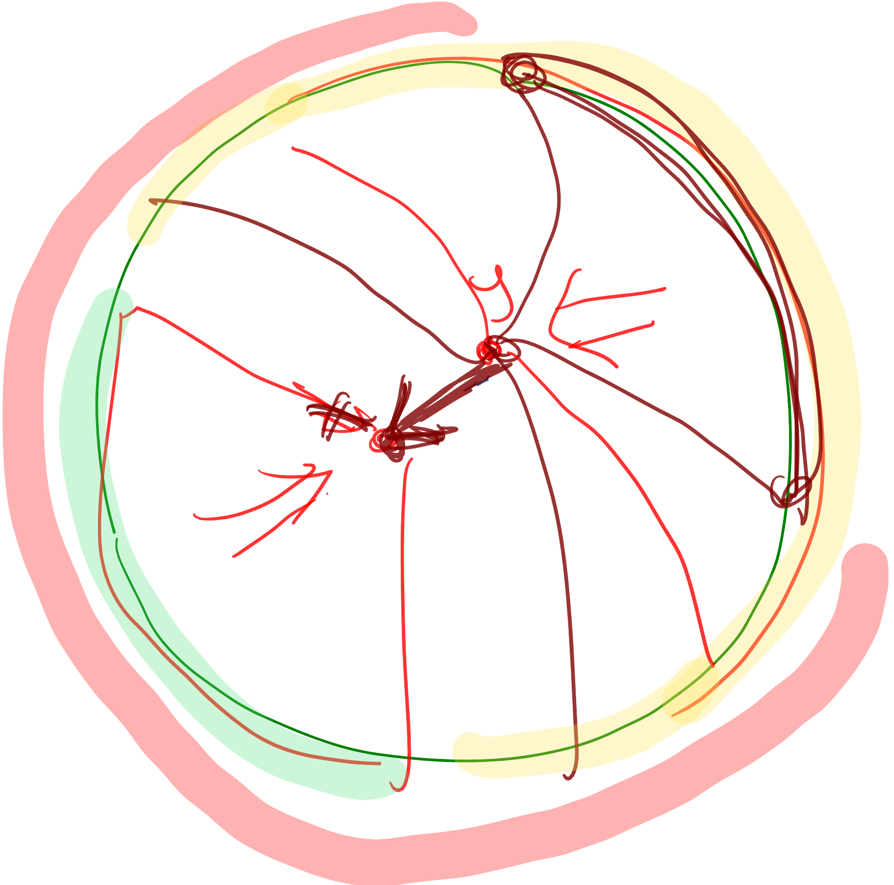
MSSP problem can be solved in O(n log n) time,
such that each distance can be queried in O(log n) time

Disk-Tree Lemma. For any spanning tree T and tree-edge e, the boundary vertices in components of T-e are consecutive.



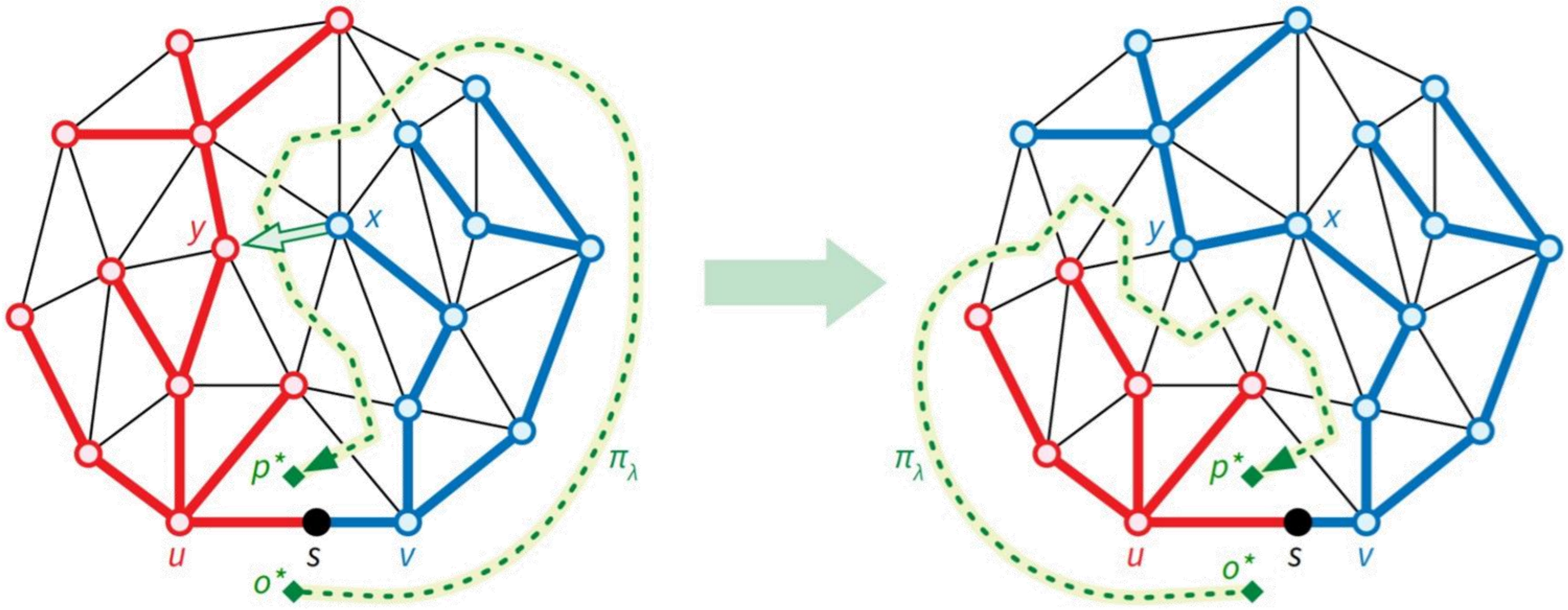
COROLLARY. Let $T_0, \ldots, T_{k-1}$ be shortest path trees in sequence. Then any edge $x \rightarrow y$ belongs to a consecutive interval of shortest-path trees: $T_i, \ldots, T_{i+j \bmod k}$.





Parametric Shortest Paths

- Shortest path tree **pivots** as one moves the source



- $d_\lambda(x)$: distance from s to x under w_λ
- $slack_\lambda(x{\rightarrow}y) = d_\lambda(x) + w(x{\rightarrow}y) - d_\lambda(y)$

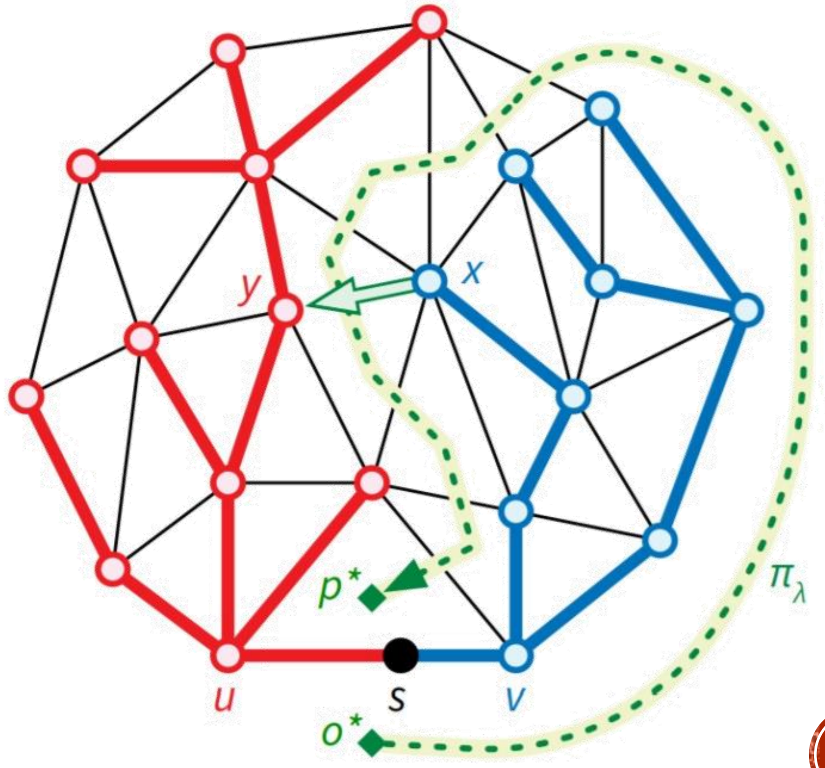
Observation. Any shortest-path tree has

- non-negative slack on all darts
- zero slack on tree darts
- positive slack on non-tree darts

Parametric Shortest Paths

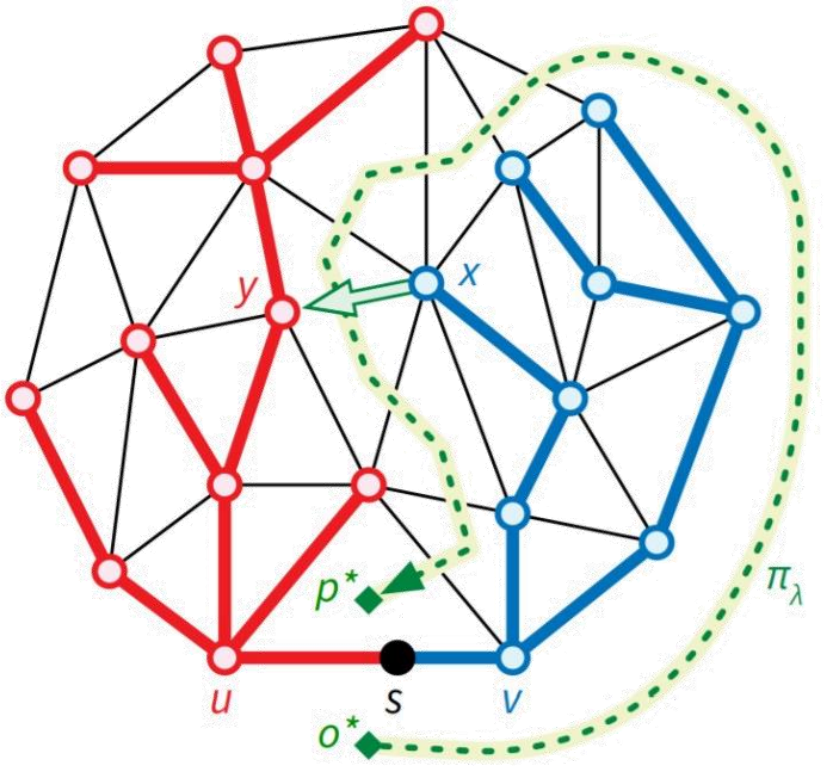
- A vertex x is
  - red if $d_\lambda(x)$ goes up as λ goes up
  - blue if $d_\lambda(x)$ goes down as λ goes up
- Dart $x \to y$ is active if
  - $\text{slack}_\lambda(x \to y)$ goes down as λ goes up

$= d_\lambda(x) + w(x \to y) - d_\lambda(y)$



Red-Blue Lemma. For any λ:

- All vertices behind u are red
- All vertices in front of v are blue
- x→y active if x blue and y red



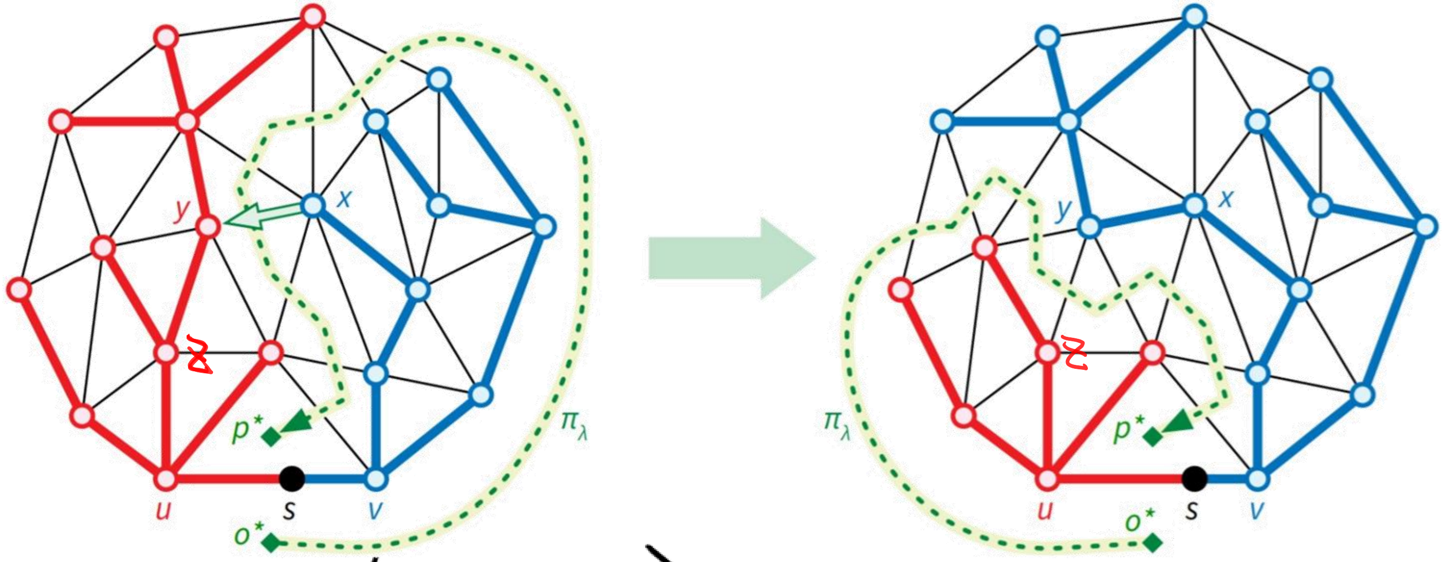
Corollary.

Active darts form dual path π_λ between o* and p*.

Corollary.

The min-slack active dart on π_λ is the next pivot.

MSSP Algorithm



NextPivot(G, T_λ):

$x \to y \leftarrow$ MinPathSlack(o^*, p^*)
$\Delta \leftarrow slack_\lambda(x \to y)/2$

if $\lambda + \Delta / w(u \to v) < 1$:
  Pivot($x \to y$, Δ)
  return $\lambda + \Delta / w(u \to v)$
else
  return 1.

Pivot($x \to y$, Δ):

AddSubtreeDist(Δ, u)
AddSubtreeDist($-\Delta$, v)

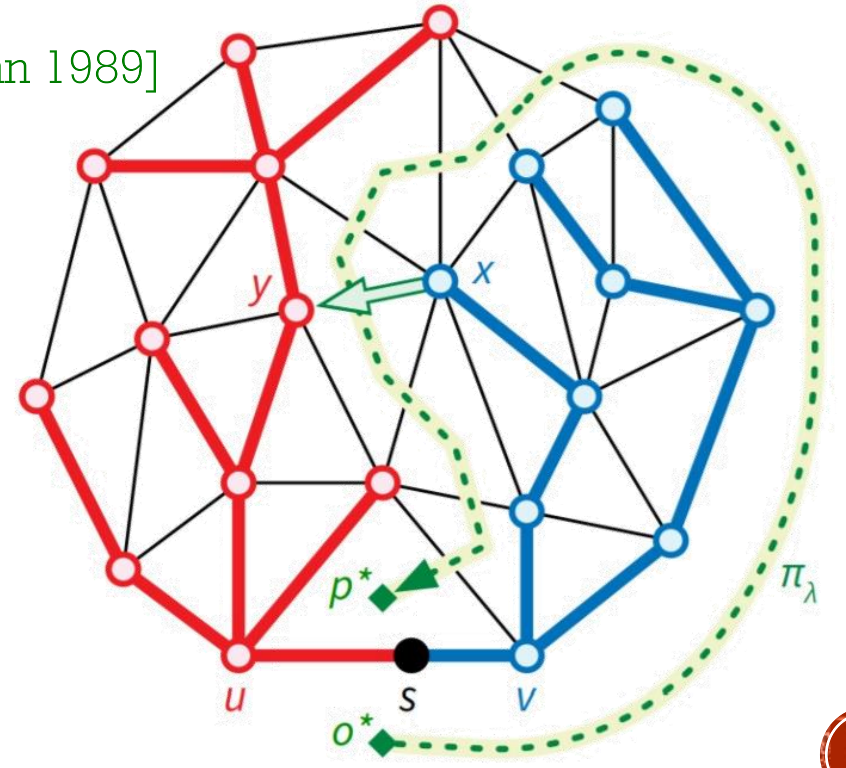
AddPathSlack(-2Δ, o^*, p^*)

$z \leftarrow pred(y)$, $pred(y) \leftarrow x$

Cut(yz), Link(xy)
Cut($(xy)^*$), Link($(z \to y)^*$).

Implementation and Analysis

- Implement tree-cotree using dynamic tree data structure
  - Splay tree into link-cut tree [Sleator-Tarjan 1982-1985]
  - Persistent data structure [Driscoll-Sarnak-Sleator-Tarjan 1989]



- Summary:
  - O(n) pivots (by disk-tree lemma)
  - Correctly identify next pivot (by red-blue lemma)
  - O(log n) amortized update time (by data structure magic)
- Thus O(n log n) time in total

Topology+Data Structure= Fast Algorithm

Next Time:

Two more tools from the toolbox
assemble our faster min-cut algorithm