Final Exam

Friday, May 7, 630 – 940 pm (includes 10 mins for submission)

Comprehensive Exam

25% – 25% – 50%
Part I II III

Part I

Chapters 0, 1, 2

Chap 0 → big oh ☆

Chap 1 → basic arithmetic operations (mul / div)
exponentiation
Euclid's GCD
Primality testing
$\boxed{\text{Skip 1.4}}$ ← crypto
Hashing

regular + modular

1 Q

Chap 2 → Divide & Conquer

Selection ← randomized
Sorting ← quicksort

1 Q

Master theorem ☆
$\boxed{\text{Skip 2.6}}$ ← FFT

Part II

Chapters 3, 4, 5

Chap 3 : DFS
↳ pre-post
↳ SCC / CC
↳ Topological sort
↳ DAGs

Connected components

Chap 4 :

BFS
Dijkstra (PQ)
Bellman-Ford (negative)

shortest paths ☆

Dijkstra (PQ)
Bellman-Ford (negative)] shorter paths

Chap 5 : greedy algorithms

MST
→ Prims + PQ
→ Kruskals + union-find ★] spanning trees

Huffman Encoding

Part III

Chapters 6, 7, 8, 9

Chap 6 Dynamic Programming

All pairs shortest paths
Shortest paths in DAGs
TSP ←] graph

longest increasing seq
edit distance] sequence 1Q

Knapsack ← fitting (value) within a budget

Skip 6.7

Chap 7

Skip 7.1

Sec 7.2
7.3 [Flows in Networks (max-flow / min-cut)
Bipartite Matching] 1Q

Skip rest: 7.4 – 7.7

Chap 8

NP / NP-completeness → decision / optimization

polynomial time reduction 1Q

TSP / rudrata → path / cycle / SAT / knapsack
Hamilton

not ask to reduce for ILP / ZOE
3D MATCHING, etc

Chap 9

1Q

Approximation ratio
TSP (9.2.3)] sec 9.2 (skip 9.2.1, 9.2.2)

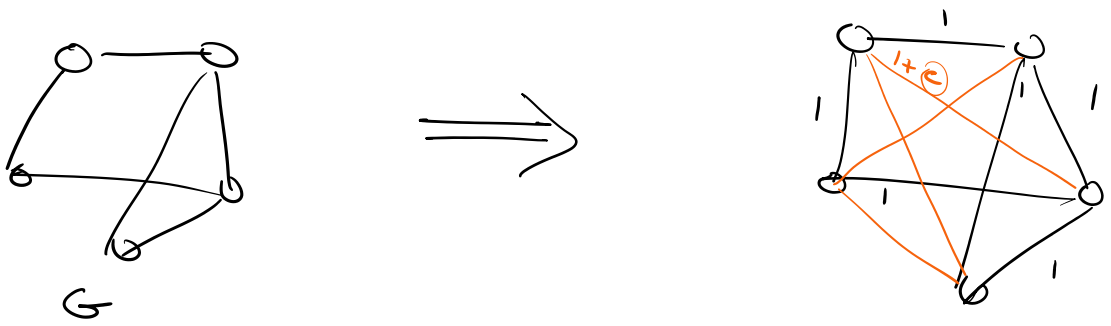
TSP: find a tour that visits each city only once, start-end @ start city given a complete graph G, with weights $d_{ij} \equiv$ distance between i and j

NP-complete

Rudrata cycle $\longrightarrow$ TSP

TSP: Hard to Approximate

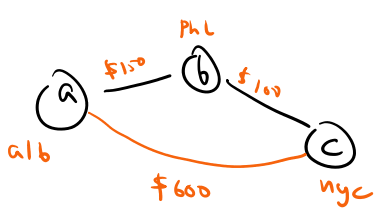
reduction from rudrata cycle, using the approx ratio within the constant c.



G

rudrata

Metric TSP: $\underbrace{\text{TSP problem}}_{G,\ d_{ij}\ \forall i,j \in V}$ + one additional property



d_{ij} are arbitrary

No polytime approximation

d_{ij} are a metric

triangle inequality

$$d_{ik} \le d_{ij} + d_{jk}$$

$$d_{ac} \le d_{ab} + d_{bc}$$

approximate with ratio 2.

approximate with ratio 2.

Metric TSP: Approximation Algo

given G, d_{ij}, that satisfy triangle inequality

$A(G)$:

1) Compute the MST for G

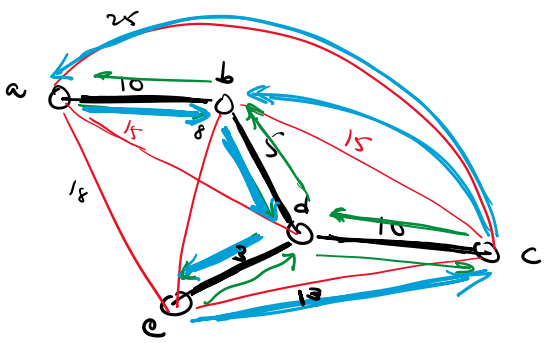
tree

↓

path (+ tour)

$$\boxed{\alpha = \max_{G} \frac{A(G)}{\text{Opt TSP}} \leq 2}$$

G:



MST : edges in black

Start city: a

$$\boxed{\alpha = \frac{A(I)}{\text{opt TSP}} \leq \frac{A(I)}{MST}}$$

$\underbrace{MST}_{\text{least of all possible spanning trees}} \leq$ Opt TSP. $\sum_{i=0}^{n-2} d_{i,i+1}$

(v_0) $\underbrace{a - v_1 - v_2 \cdots v_{n-1}}_{\text{path (except for cycle completion)}}$

a path is also a tree!

2) Create a double walk:

$a \to b \to d \to e \to d \to c \to d \to b \to a$

Cost: $2 \cdot MST$

3) transfor. into a tour: $\boxed{a \to b \to d \to e \to c \to a}$ $\leq 2\,MST$

$d_{ec} \leq d_{ed} + d_{dc}$ ← triangle inequality!

$A(I) \leq 2\,MST$

$$A(I) \leq 2\,MST$$

$$\alpha \leq \frac{A(I)}{MST} \leq \frac{2\,MST}{MST} = 2$$

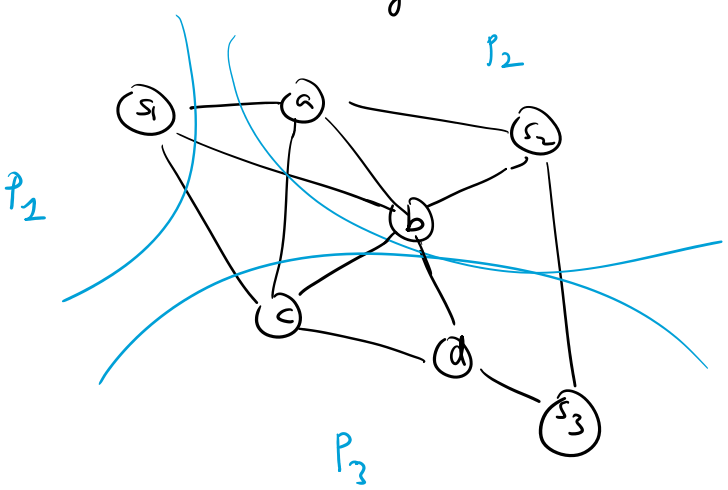
In polynomial time we can approximate Metric TSP

Q9.7 given G, given distinct vertices $s_1, s_2, s_3, \ldots, s_k$ partition G into k parts, such that each s_i belongs to a different part, minimize the cut edges between the diff parts

Multiway minimum cut

Cost of cut

9



$k = 3$

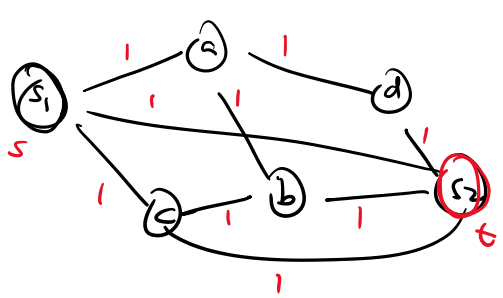
Cut : # of edges that cross from one part to another

find min 3-way cut

a) when $k=2$, give a polynomial time solution

$s_1 = s$, $s_2 = t$

source sink



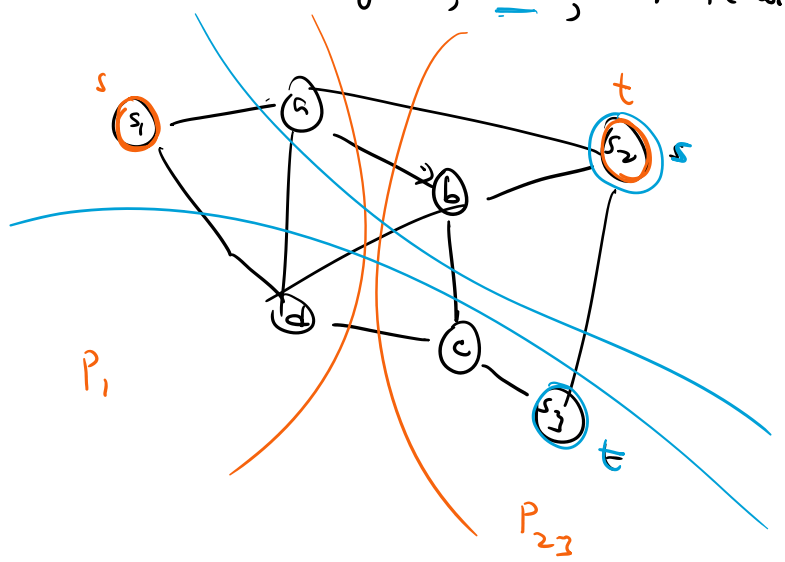
find a max-flow in G from s to t

extract the min cut

b) give an approximation also for $k = 3$, approx ratio ≤ 2

Optimal 3-way cut, O^*, let the corr be E^*

# of edge that cross between parts



Compute the optimal

1) (s_1, s_2) cut (mincut) E_{12} # of cross edgs

2) (s_2, s_3) mincut E_{23} # cross edgs

3) return union of edge in E_{12} and E_{13}

$O = \{O_1 \; O_2 \; O_3\}$

$= \{O_1, [O_2 \; O_3]\}$ $s_1 \; s_2$ cut

$= \{O_2, (O_1 \; O_3]\}$ $s_2 \; s_3$ cut

$|E_{12}| \leq |E|^*$

$|E_{23}| \leq |E|^*$

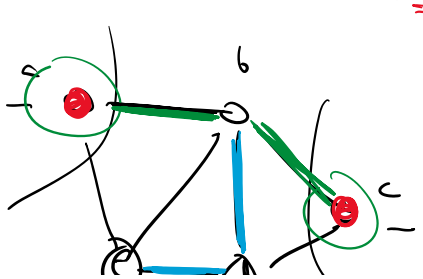
$|E_{12} \cup E_{13}| \leq |E_{12}| + |E_{23}| \leq 2|E^*|$

$$\alpha = \frac{A(I)}{E^*} \leq \frac{2\cancel{E^*}}{\cancel{E^*}} = 2$$

Q 8.13 : Give a polytime solution or show that it is NP-complete

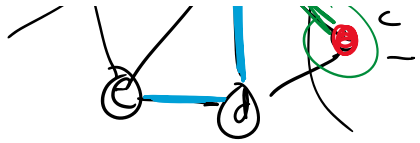
given undirected graph G

a) given a set $L \subseteq V$, find a spanning tree such that its leaves include all the node in L

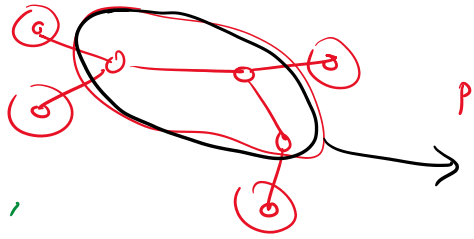


$L = \{a, c\}$

find any spanning tree that



find any spanning tree that include a & c as leaves



periphery / leaves

core

1) delete a & c from G to obtain $G' = V - L$

2) find a spanning tree in G', say T'

3) add back any edge that connects vertices in L to T'

Polynomial time $O(|V| + |E|)$

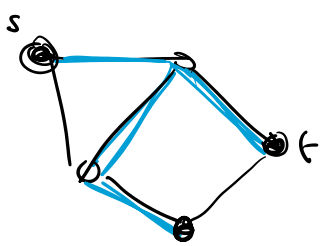
b) find a spanning tree, but the leaves must be exactly L

rudrata problem (s–t) NP-complete

G, let $\boxed{L = \{s, t\}}$

leaves must be exactly s and t

(only 2 leaves are allowed in the spanning tree)



Any tree that has exactly 2 leaves, must be a path

If G has a spanning tree with $L = \{s, t\}$ as the exact set of leaves

then that tree is a path that visits all the vertices exactly once

$\Rightarrow$ rudrata s–t path

$\Rightarrow$ rudrata s-t path

NP-complete!

c) similar to b.

d) find a spanning tree with $\underline{k}$ or fewer edges.

NP-complete

since when $k=2$, this is the same as rudrata path