Approximation Algorithms

- 'Efficient Algorithms' which give answers 'close' to desired ones.

efficient - polynomial running time

- Why? (1) NP-Hard : VERY hard to find P-time algorithm
  NP-Hard optimization.

(2) Sometimes even 'easy' problems do not have 'good enough' ~~solution~~ algorithm.

Eg: EDIT DISTANCE $O(mn)$
Till now no sub quadratic ($< n^2$) algo.
So compromise on the algo for fast solution ← Approx algo.

Why taught by sir?
1. Explain interesting design + analysis techniques beyond ADA.
2. Research :- Fun Math

Evaluation

Drills : 10% submit (email) by next day
HW : 40% (only one which is coding)
↳ groups of 2 or individual
Exams : 50% long exam - half day / entire day
(25% + 25%)

Resources - posted on GC.
Lecture Notes

Optimization Problem

Π is an optimization problem

Π : Set of instances : $\mathcal{I}$

$\$$: Set of "feasible" solutions

$c : \$ \rightarrow \mathbb{R}$ [cost function]

$A : \mathcal{I} \rightarrow S$ which minimizes/ $c(S)$, $S \in \$$
maximize

A ← algo, S ← solution

opt $(\Pi, \mathcal{I})$: min / max feasible solution for $\mathcal{I}$ on Π.

Approximation Algorithms

For a minimization problem Π
A is an α-approximation if

$$\max_{\mathcal{I}} \frac{c(A(\Pi, \mathcal{I}))}{c(\text{opt}(\Pi, \mathcal{I}))} = \alpha \qquad [\alpha \geq 1]$$

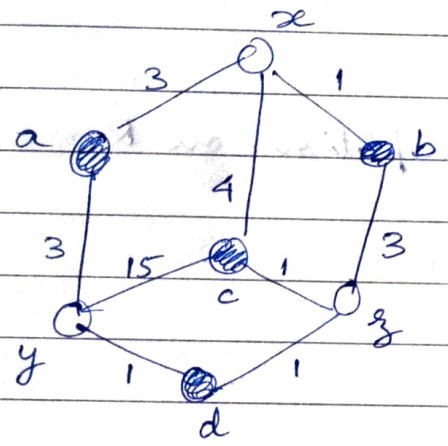
$\alpha = 1$ → exact algo

$\alpha \nearrow$ → strictly approximation algo

For a maximization problem Π,
A is an α-approximation if,

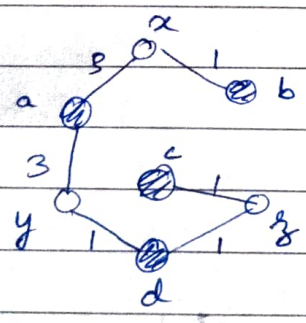
$$\min_{I} \frac{c(A(\pi, I))}{c(opt(\pi, I))} = \alpha \qquad [\alpha \leq 1]$$

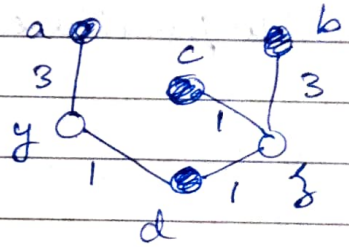
Example — Steiner Tree



$G = (V, E)$;
$l : E \longrightarrow \mathbb{R}_{\geq 0}$
$R \subseteq V$ [Terminals]
shaded.
Non-mandatory nodes ← Steiner vertices.
Minimum tree that spans all the terminals.
[NP-Hard]

Feasible solution





Ex 1: Find an example graph such that for any constant α cost of MST+Prune $> \alpha \cdot c(opt(G))$.

[i.e. MST+Prune is not a constant factor approximation]

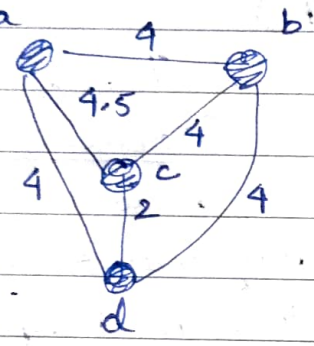
MST+Prune ⇒

terminals ↓ ↓ new cost function

Given G, create H (R, w) is a complete graph.

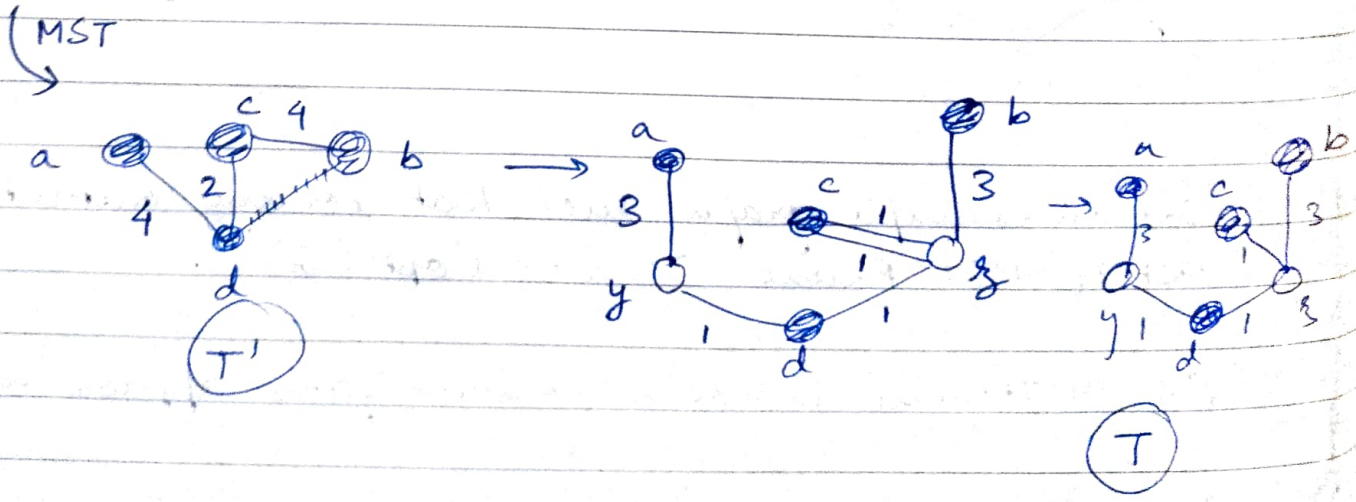
$w(u, v)$ = Shortest path length between u, v in G.

H:



Metric Completion on R.

Alg: Create H (All pair shortest path)
Find MST on H
unfurl the terminal edges
delete parallel edges



Thm: MST-Heuristic is a 2-approx for Steiner Tree problem

Proof

Let T: Tree returned by the algorithm

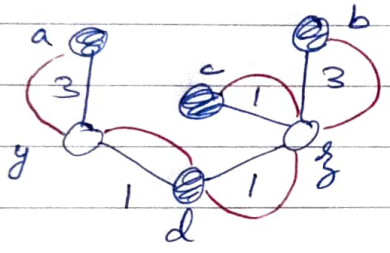
T': intermediate tree

$l(T) \leq w(T')$ [the cost can only go down by removing redundant edges]

Enough to prove:

$w(T') \leq 2\, l(opt(G))$

Analysis :-

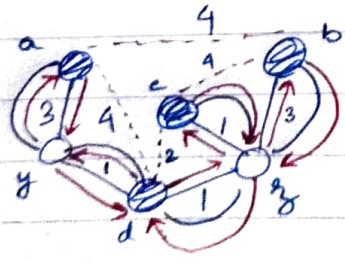


Thought process

- duplicate all edges of optimal solution.
- all vertices will have even degree
- will have Eulerian Walks
  [start and stop at same vertex, each edge traversed once.]

- start a walk from one of the terminals.
  skip Steiner nodes.
  skip used nodes
  end at start node.

MST < subgraph.



Cost of dotted graph H'

$\leq 2 \cdot l(opt(G))$

Also, H' spans all of R and is a subgraph of H.

~~Also~~

~~H' spans all of R~~ ~~(MST T' C dotted graph H')~~

$H' \subset H$ (subgraph) $\therefore w(T') \leq$ cost of dotted graph H' (MST)

Ex2:- There exists a graph s.t.
MST- heuristic $\approx 2 \cdot c(opt(G))$.