Prize-Collecting Steiner Tree: A 1.79 Approximation

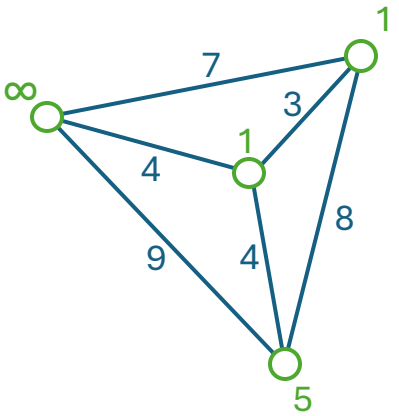
Ali Ahmadi, Iman Gholami, MohammadTaghi Hajiaghayi,
Peyman Jabbarzade, Mohammad Mahdavi

University of Maryland

STOC 2024

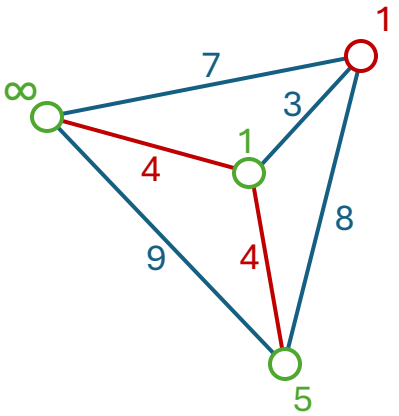
Problem Definition
Prize Collecting Steiner Tree (PCST)

- Given a weighted graph with a root with penalties assigned to vertices
- Find a tree T including root
  - Minimizing total weight of T + penalties for excluded vertices
- Rooted and Unrooted versions are equivalent
  - π(root) = ∞
- NP-Hard



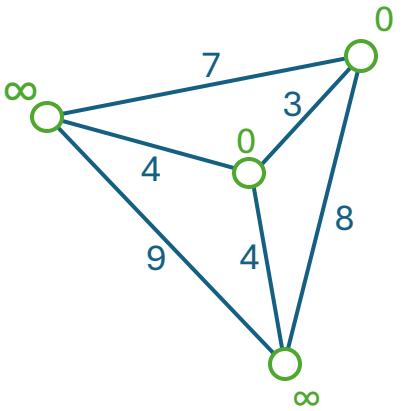
Problem Definition
Prize Collecting Steiner Tree (PCST)

- Given a weighted graph with a root with penalties assigned to vertices
- Find a tree T including root
  - Minimizing total weight of T + penalties for excluded vertices
- Rooted and Unrooted versions are equivalent
  - π(root) = ∞
- NP-Hard



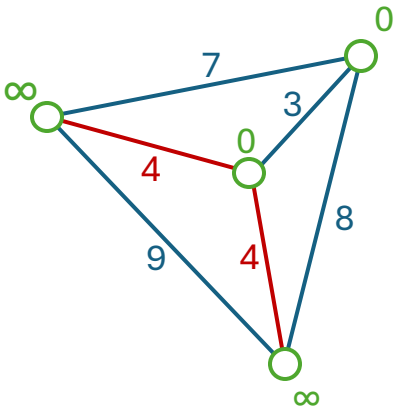
Problem Definition
Steiner Tree

- Given a weighted graph with a set of nodes called terminals
- Find a tree T spanning all terminals
  - Minimizing total weight of T
- Specification of PCST
  - $\pi(v) = \infty$ for all terminals and $\pi(v) = 0$ for other vertices
- A generalization of MST
- NP-Hard



Problem Definition
Steiner Tree

- Given a weighted graph with a set of nodes called terminals
- Find a tree T spanning all terminals
  - Minimizing total weight of T
- Specification of PCST
  - $\pi(v) = \infty$ for all terminals and $\pi(v) = 0$ for other vertices
- A generalization of MST
- NP-Hard



Previous Work
Steiner tree

- Trivial 2-approximation [KMB, Acta Informatica 1981]
  - MST on terminals when metric
- Sub 2 approximations:
  - 11/6≈1.833 [Zel, Algorithmica 1993]
  - ln(4)+ε≈1.39 [BGRS, STOC'10]
    - We use this as a black box

Previous Work
PCST

- 2-Approximation [GW, SIAM J. Comput. 1995]
  - Primal-Dual Approach
  - A crucial part of our work
- 1.9672-Approximation [ABHK, SIAM J. Comput.'11]
- Our contribution: 1.7994-Approximation

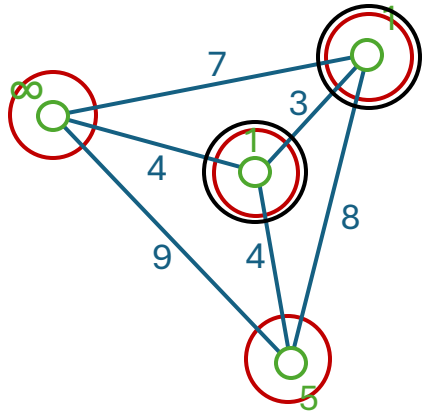
Goemans Williamson Algorithm

2-approximation algorithm

Coloring interpretation

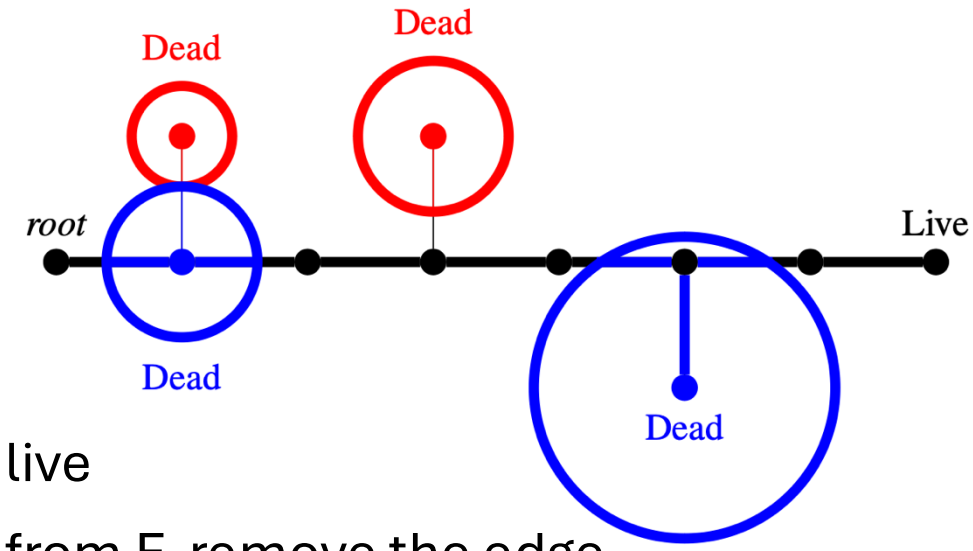
Basics

- Edges are curves with length=weight
- We maintain a forest F
  - Empty at the beginning
- Each vertex has a unique color
  - With coloring capacity π(v)
- Active sets
  - Connected components of F with color remained
- Active sets color their adjacent edges



GW Algorithm

- Active sets color their adjacent edges
- An edge is fully colored
  - Add the edge to F
  - Merge 2 connected components
- Finish when there is only one active set
- Vertices run out of color are dead, otherwise live
- Prune phase: If a dead set cuts a single edge from F, remove the edge
- Return component of root



Intuition on Approximation Guarantee

- X: total coloring moment for all active sets
- Optimal solution is at least X
  - For each moment of coloring
    - Either color at least one edge of OPT, or
    - We pay a proportional penalty
- Our solution is at most 2X
  - For each moment of coloring
    - Either color 2 edges of our solution on average, or
    - We pay a proportional penalty

Modify Penalties

- Use parameter $1 <= \beta <= 2$:
- Set coloring capacities $\pi(v) / \beta$
- Still 2-approximation
- Colors run out earlier
  - Fewer vertices connected to root
- Why modify penalties?

This is Why

- X: total coloring moment for all active sets
- Optimal solution is at least X
  - For each moment of coloring
    - Either color at least one edge of OPT, or (coefficient 1)
    - We pay a proportional penalty (coefficient 1)
- Our solution is at most 2X
  - For each moment of coloring
    - Either color 2 edges of our solution on average, or (coefficient 2)
    - We pay a proportional penalty (coefficient 1)

Our Recursive Algorithm

1.7994-approximation algorithm

Recursive Algorithm

- Run GW Algorithm with parameter β (GW solution)
- Pay penalties for all dead vertices
  - Remove those penalties from the instance
- Run SteinerTree on live vertices (ST solution)
- If we removed penalty in this phase
- Recursively run the algorithm on the modified instance (IT solution)

- Return the best solution among all obtained

Why Pay Dead Vertices Penalties?

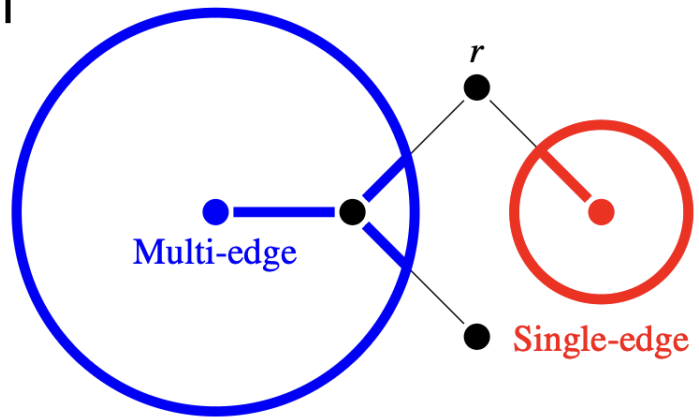
- Some dead vertices are connected to root
- Dead vertices have used all of their color capacity
- Connecting them costs too much
- Let’s pay their penalty

Why SteinerTree?

- If most vertices are connected in OPT and GW
- Let's connect them more optimal
  - Cost $1.39c(T_{OPT})$ to connect vertices of T_{OPT}
- Issue: still we may connect more vertices
  - Connect them to T_{OPT} the way that GW do
  - At most pay 1.39 times the cost of GW for them

Why Recursive Solution?

- Let R be the recursive instance
- If our algorithm is α-approximation
  - Using induction, the recursive solution is $\alpha \cdot cost(OPT_R) + \pi(dead\ vertices)$
- Recap about coloring of connected vertices
  - Active sets color at least one edge of OPT
  - They color on average two edges of GW
- If most of the times active sets color exactly one edge of OPT
  - $cost(OPT_R)$ is too small
    - We can remove many edges from T_{OPT} to have a valid solution for instance R
- Otherwise
  - GW is a good solution



Analysis Overview

- Classify vertices based on
  - Whether connected to root in OPT
  - Whether is a dead vertex in GW
- We bound every solution as $\alpha \cdot cost(OPT) + extra$
- Take a weighted average on different solutions
  - Find α, β, and weights such that extra value become at most zero
    - $\alpha = 1.7994$, $\beta = 1.252$
  - Then the minimum is α-approximate solution

Thanks