CS4234
Optimiz(s)ation Algorithms

L3b – Steiner-Tree

still-DRAFT (since 2017) VISUALIZATION

(to be improved by 1 FYP student in AY 23/24):

https://visualgo.net/en/steinertree

PS: This lecture will run in two parts:

On Week 03 (up to the *preview* of MST-based approximation algorithm)

On Week 04 (the full details of this 2-approximation algorithm)

Motivation

Imagine you were given a map containing a set of cities, and were asked to develop a plan for connecting these cities with roads. Building a road costs 1 000 000 SGD per kilometer, and you want to minimize the length of the highways. Perhaps the map looks like this:

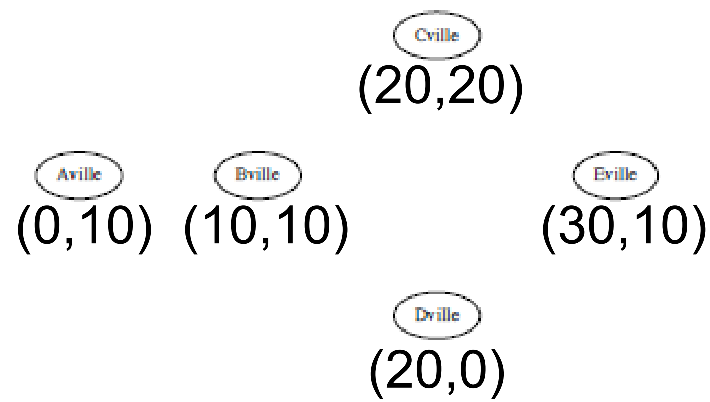
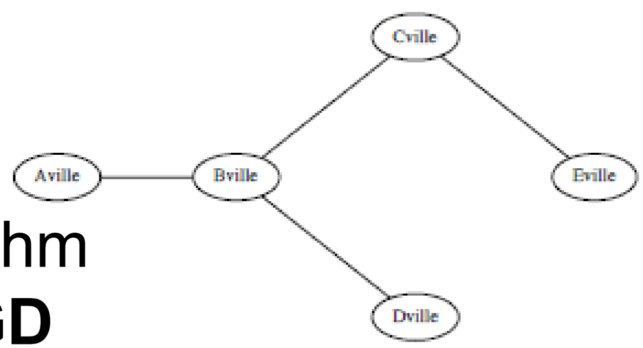
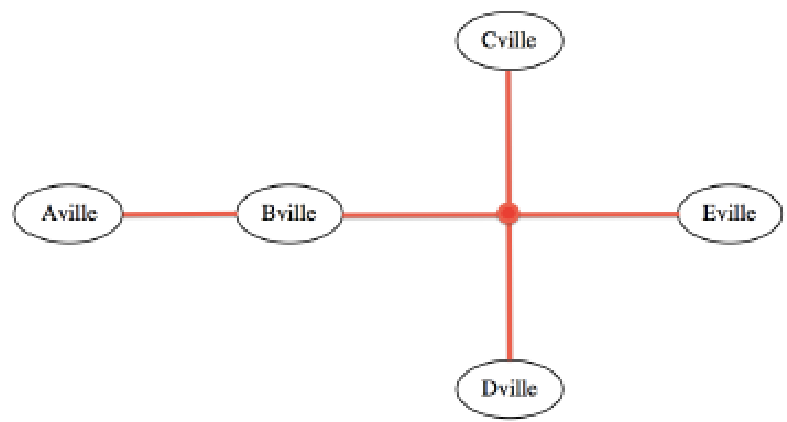


Figure 1: How do you connect the cities with a road network as cheaply as possible?

MST?
Run Prim's or Kruskal's algorithm
Cost: **~52 M SGD**



O($\mathbf{V}^2$ log $\mathbf{V}$) solution
Note that we are dealing with $\mathbf{K_V}$



But what if we can add new points to help us?
Cost: **50M SGD**

EUCLIDEAN-STEINER-TREE

- Given a set **R** of **n** distinct points in the Euclidean (2-dimensional) plane
- Find a(n additional, possibly empty) set of points **S** and a spanning tree **T = (R ∪ S, E)** such that that the weight of the tree is minimized
- The weight of the tree is defined as: $\sum_{(u,v)\in E} |u-v|$

  where **|u – v|** is the Euclidean distance from **u** to **v**
- The resulting tree is called a **Euclidean Steiner Tree** and the points in **S** are called **Steiner points**

This is NP-hard (proof omitted)

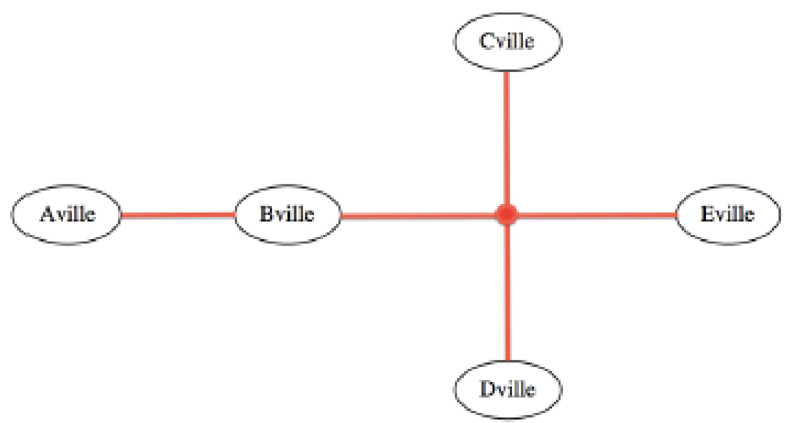
But there are some known properties of any optimal Euclidean Steiner Tree:

- Each Steiner point in an optimal solution has **degree 3**
- The three lines entering a Steiner point form **120 degree angles**, in an optimal solution
- An optimal solution has at most **n-2** Steiner points

So is this Steiner Tree (**50M**) optimal?

If no, can you come up with a better one?

Review recording to see the solution



Flipped Classroom Challenge

Now we switch to Lecture 3b and T2: The EUCLIDEAN-STEINER-TREE problem. Based on your understanding on the structure of an optimal EUCLIDEAN-STEINER-TREE solution, try your best to derive a manual solution for this instance below with 8 points on 2D Euclidean space. The actual coordinates of the 8 points are ignored so that you can concentrate on the *structure* of the optimal Euclidean Steiner Tree. Answer that is an Euclidean Steiner Tree but not the minimum Euclidean Steiner Tree will be given some partial marks if it is not worse than 2-times the optimal answer.

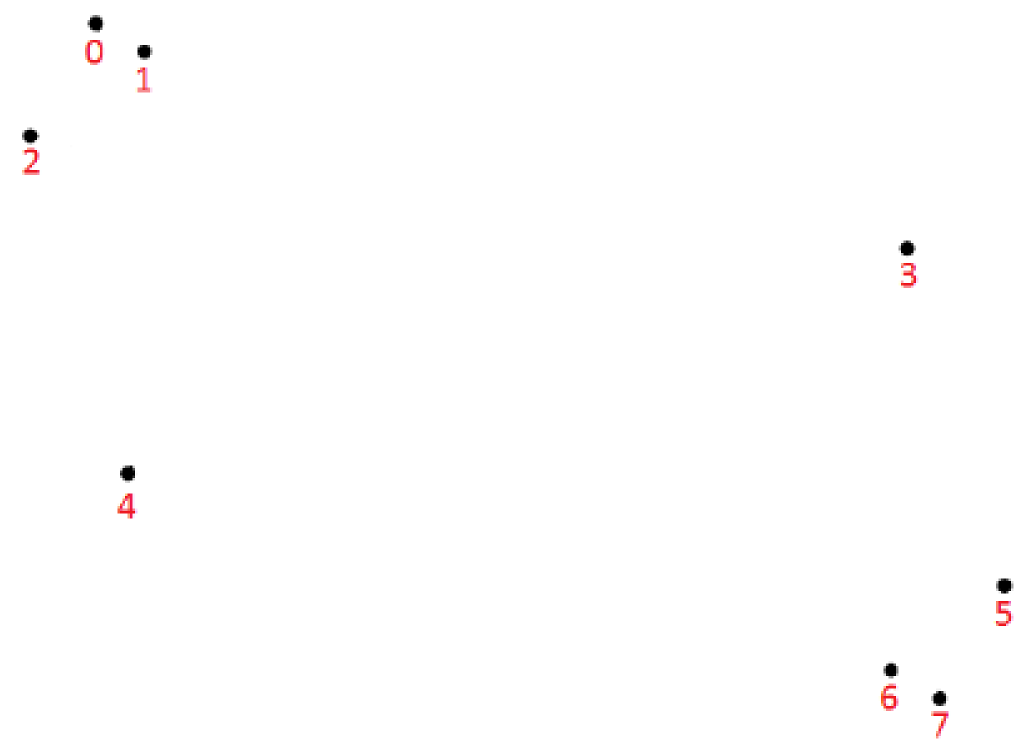


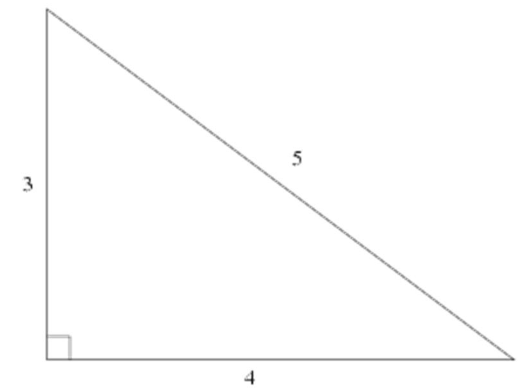
Figure 5: Instance for Part 5, draw your best Euclidean Steiner Tree of these 8 points.

`Metric-Steiner-Tree` (1)

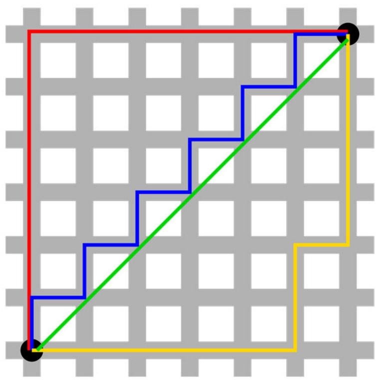
Let's define a metric function, e.g., Euclidean distance

Definition 3 *We say that function* $d : V \times V \to \mathbb{R}$ *is a metric if it satisfies the following properties:*

- Non-negativity: *For all* $u, v \in V$, $d(u, v) \geq 0$.
- Identity: *For all* $u \in V$, $d(u, u) = 0$.
- Symmetric: *For all* $u, v \in V$, $d(u, v) = d(v, u)$.
- Triangle inequality: *For all* $u, v, w \in V$, $d(u, v) + d(v, w) \geq d(u, w)$.



There are several functions other than Euclidean distance that are metric, e.g., the Manhattan/taxicab/rectilinear distance



Metric-Steiner-Tree (2)

Unlike in Euclidean case where the additional Steiner points can be anywhere on the 2D plane, we are also given the set of possible Steiner vertices **S**

Definition 4 *Assume we are given:*

- *A set of required vertices R,*
- *A set of Steiner vertices S,*
- *A distance function $d : (R \cup S) \times (R \cup S) \rightarrow \mathbb{R}$ that is a distance metric on the points in R and S.*

The METRIC-STEINER-TREE *problem is to find a subset $S' \subset S$ of the Steiner vertices and a spanning tree $T = (R \cup S', E)$ of minimum weight. The weight of the tree $T = (R \cup S', E)$ is defined to be:*

$$\sum_{(u,v) \in E} d(u,v) \, .$$

Think: Does this make **Metric**-Steiner-Tree $\boxed{\text{easier}}$ or harder than the **Euclidean**-Steiner-Tree?

General-Steiner-Tree

We can generalize this even further

At this point, we can generalize even further to the case where d is not a distance metric. Instead, assume that we are simply given an arbitrary graph with edge weights, where some of the vertices are required vertices and some of the vertices are Steiner vertices.

Definition 5 *Assume we are given:*

- *a graph* $G = (V, E)$,
- *edge weights* $w : E \to \mathbb{R}$,
- *a set of* ***required*** *vertices* $R \subseteq V$,
- *a set of* ***Steiner*** *vertices* $S \subseteq V$.

Assume that $V = R \cup S$. *The* **General-Steiner-Tree** *problem is to find a subset* $S' \subset S$ *of the Steiner vertices and a spanning tree* $T = (R \cup S', E)$ *of minimum weight. The* ***weight*** *of the tree* $T = (R \cup S', E)$ *is defined to be:*

$$\sum_{(u,v) \in E} d(u, v) .$$

Steiner-Tree (the 3 variants)

All NP-hard (proof by book, Garey & Johson, 1979)...

- Euclidean: The points are in Euclidean plane
- Metric: We have a distance metric
- General: On arbitrary graph

General-ST is a generalization of Metric-ST

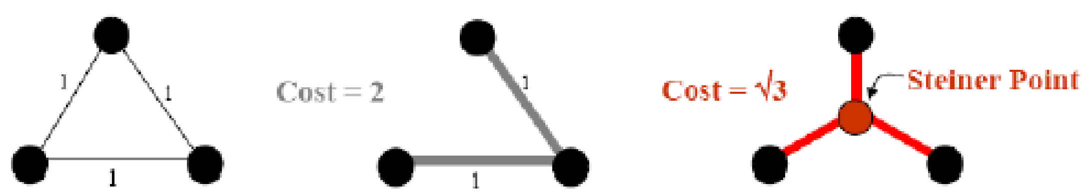
Metric-ST is not simply a generalization of Euclidean-ST as Euclidean-ST allows any points in the plane to be a Steiner point

`Steiner-Tree` Complete Search Solution

- https://visualgo.net/en/steinertree
- Now with some form of e-Lecture slides (in 2023), but anyway
  - Draw all the **R**equired vertices first, label as [0, 1, …, **s**-1]
  - Then draw all the **S**teiner vertices [**s**, **s**+1, …, **n**-1]
  - Click "Exact" and enter the value of **s** accordingly
  - VisuAlgo will show one possible way to solve General-ST by trying all $\mathbf{2^{|n-s|}}$ possible subsets of **S**teiner vertices, include them and their associated weighted edges along with the **R**equired vertices, and then run MST algorithm on them (each in O(E log V) = O($\mathbf{n}^2$ log **n**))
  - Time complexity: O($2^{|\mathbf{n}-\mathbf{s}|}$ * $\mathbf{n}^2$ log **n**), very slow for big TC
- Review recording for a sample run

Steiner-Tree Approximations

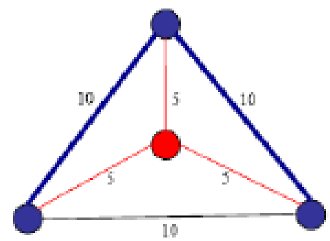
- Steiner-Tree is known to have a close relationship with the (easier, e.g., in P) Min-Spanning-Tree problem (hence I put MST in PS1-Prerequisites)
- So… what if we just ignore all Steiner points/vertices **S** and just find the MST on the required vertices **R** only?



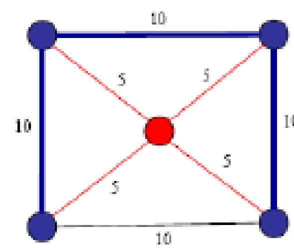
Ratio: $2/(\sqrt{3})$ or ~1.15

Widening Approximation Ratio?

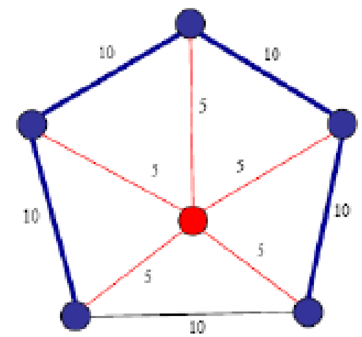
We can increase the size of this cycle graphs C_n



MST: 20
Steiner Tree: 15
Ratio: 1.33



MST: 30
Steiner Tree: 20
Ratio: 1.5



MST: 40
Steiner Tree: 25
Ratio: 1.8

PS: weights are non-metric (but almost metric)

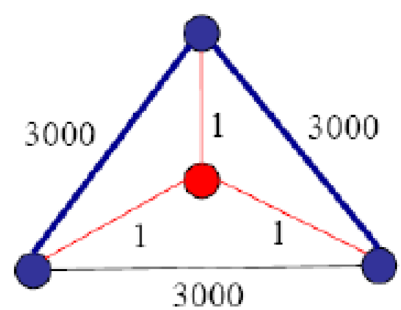
Warning: The Steiner Tree may not be the optimal one, e.g., can you draw an even better Steiner Tree for C_4 and C_5 above?

This test case shows that the MST approximation is *no better than* a **2(n-1)/n**-approximation of the optimal spanning tree (the Steiner tree)

- As **n** gets large, this is ~**2**-approximation
- We will do the more proper analysis soon

Bad for `General-Steiner-Tree`

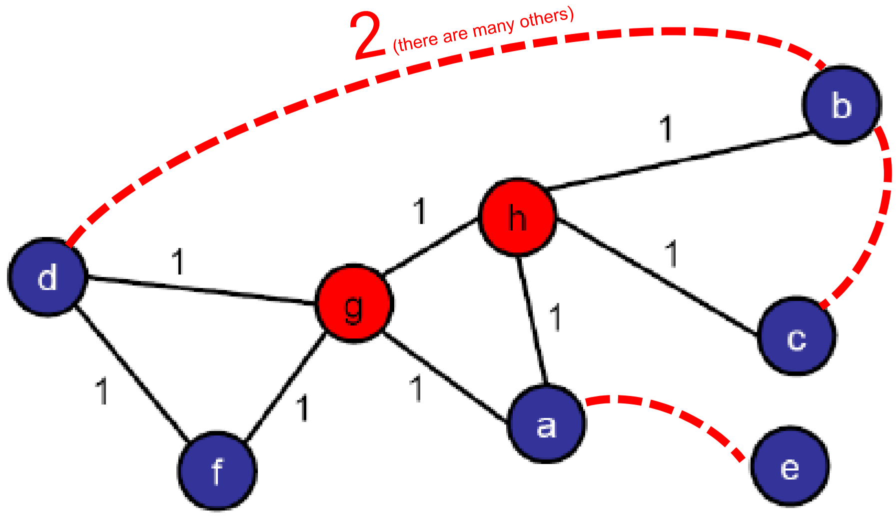
Bad news for general, *non-metric*, Steiner Tree variant… as we can construct similar test case but play with the non-metric weights to have a very bad MST approximation solution



Natural Breakpoint

- Most years, I won’t be able to finish all slides on Week 03
- Even though this is a recorded lecture, I plan to keep the flow to be the same as in 2022 edition, i.e., I stop here and continue after I return from IOI 2023 on Week 04 (the early part of Week 04 lecture will also act as a refresher of MSC and Steiner-Tree topics)

Focus on `Metric-ST` First (1)



PS: weights are metric, but not drawn in a proper 2 dimension

Figure 4: Example for showing that a minimum spanning tree is a 2-approximation of the optimal Steiner tree, if the distances are a metric. Assume that all edges *not* drawn have distance 2. Verify that these distances satisfy the triangle inquality.

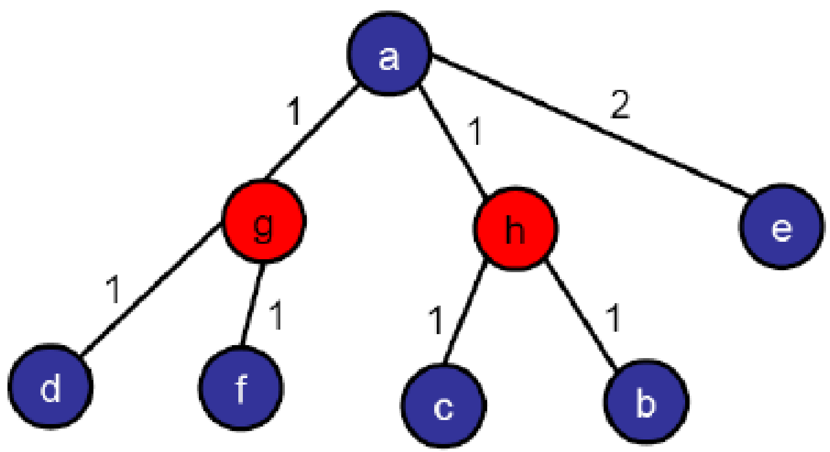


Figure 5: Here we have drawn the optimal Steiner tree T of graph shown in Figure 4.

Cost of the optimal Steiner tree T = 8

Focus on Metric-ST First (2)

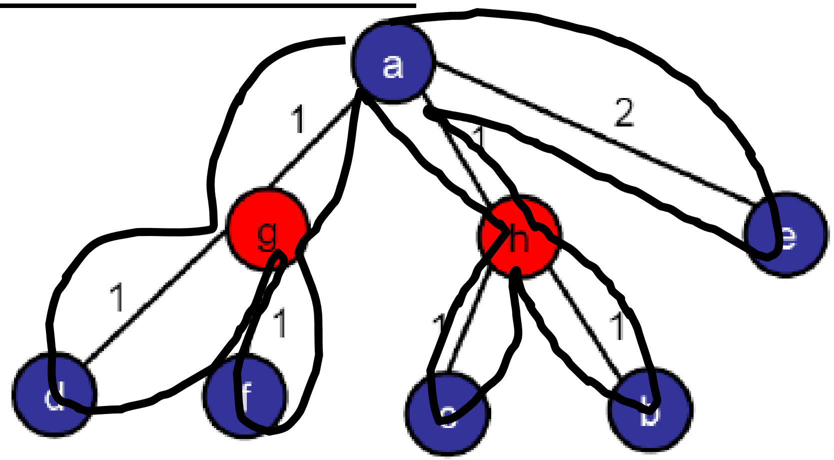


Figure 5: Here we have drawn the optimal Steiner tree T of graph shown in Figure 4.

Cost of DFS on T: 1+1+1+1+1+1+1+1+1+1+1+1+2+2 = **16**

Each edge in **T** is visited **2x** in this cycle **C**, i.e., **cost(C) = 2 x cost(T)**

This is what we want to prove…, i.e., what if we 'ignore' Steiner vertices

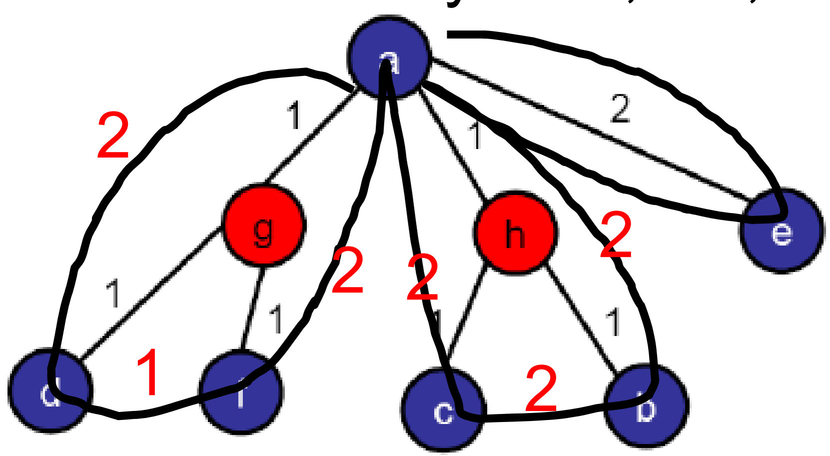


Figure 5: Here we have drawn the optimal Steiner tree T of graph shown in Figure 4.

cost(**C'**) ≤ 2 x cost(**T**)

esp on why
w(d-f) = 1 yet w(c-b) = 2

Cost of **C'**, bypassing all Steiner vertices = **15** (also see Fig 4 for clarity)

Such short-cutting won't increase the cost, because of triangle inequality

Focus on `Metric-ST` First (3)

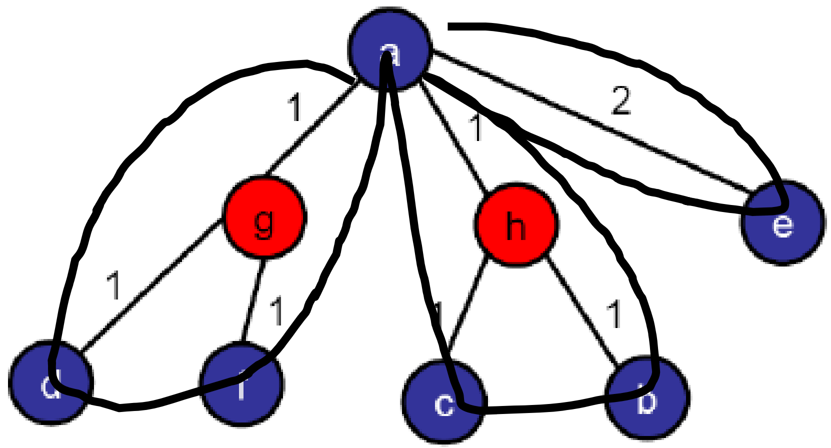


Figure 5: Here we have drawn the optimal Steiner tree T of graph shown in Figure 4.

cost(**C'**) ≤ 2 x cost(**T**)

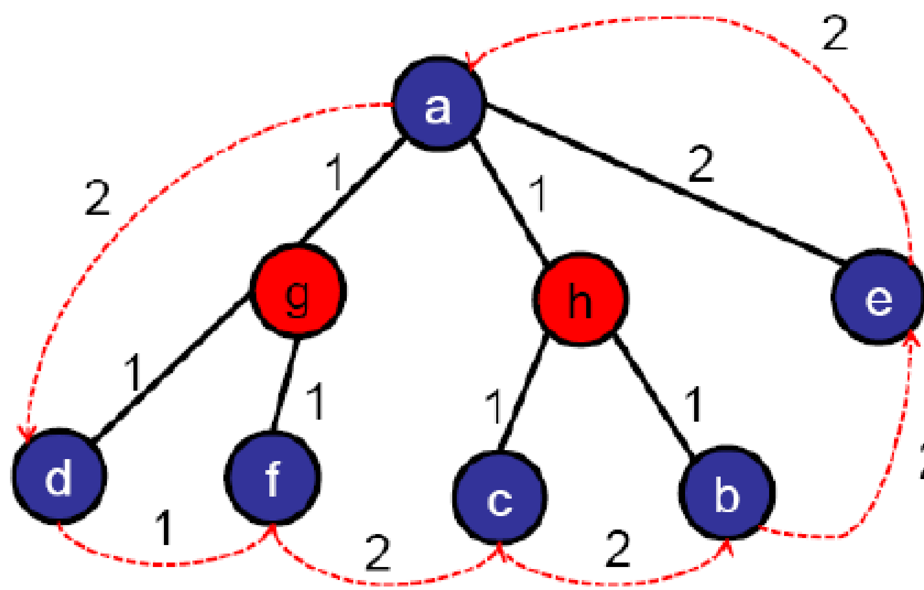
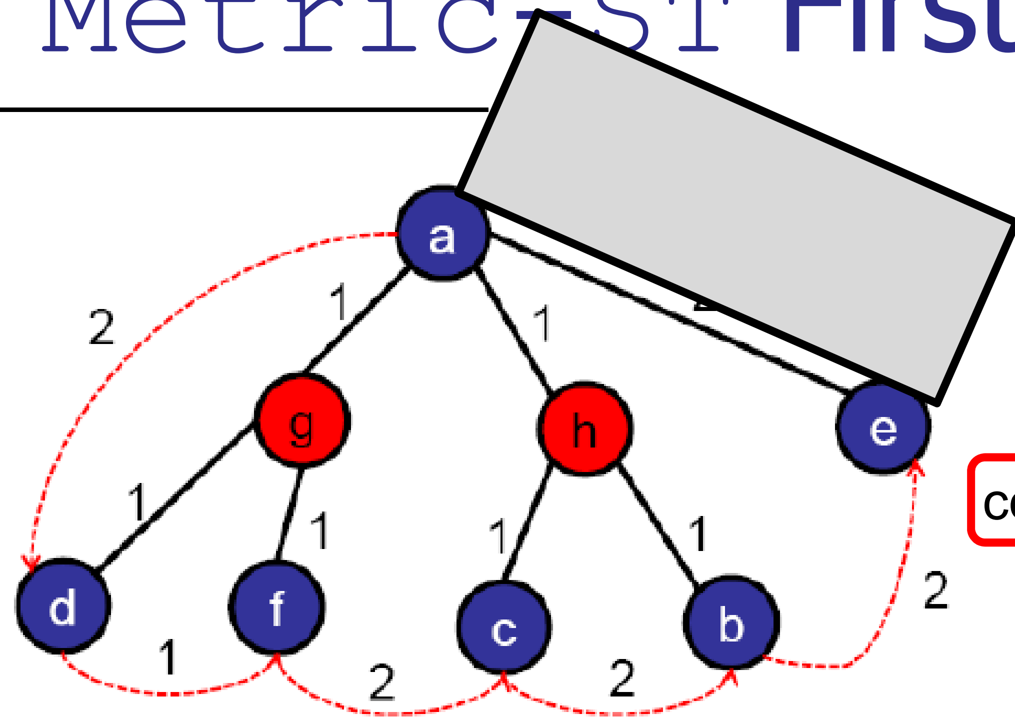


Figure 6: Here we have drawn the cycle C after the Steiner vertices have been short-cut and the repeated vertices have been deleted.

cost(**C''**) ≤ 2 x cost(**T**)

Cost of **C''**, bypassing all Steiner vertices from **C**
and then removing duplicate vertices = 11

Focus on `Metric-ST` First (4)



cost(**T'**) ≤ 2 x cost(**T**)

Figure 6: Here we have drawn the cycle C after the Steiner vertices have been short-cut and the repeated vertices have been deleted.

Cost of **T'**, bypassing all Steiner vertices from C
and then removing duplicate vertices
and then breaking any one edge in this cycle (e.g., e→a) = 9
This **T'** is an acyclic **spanning tree** of **G**
Let **M** be the **<u>minimum</u>** spanning tree of the required vertices **R** of **G**
So, **cost(M)** **≤ cost(T')**
≤ 2 x cost(T)
Conclusion: **cost(M)** ≤ **2 x cost(T)**, a **2**-approximation ☺

Now how about `General-ST`?

Can we do this?

1. Given an instance of the General-ST problem
2. **Reduce it** to a new Metric-ST instance
3. Solve the new Metric-ST instance using an existing (approx) algorithm that solves the metric version (e.g., the **2**-approximation: MST on **R** vertices only)
4. **Convert the solution** of Metric-ST **back** to a solution for General-ST

Is this a **gap-preserving** reduction, i.e., can we also have 2-approximation on General-ST variant?

Step 2: Metric completion

We can make a non-metric edge weights into a metric one, using All-Pairs Shortest Paths algorithm, e.g., the **O(V³)** Floyd-Warshall (other algorithms exist)

Definition 7 *Given a graph $G = (V, E)$ and non-negative edge weights w, we define the **metric completion** of G to the be the distance function $d : V \times V \to \mathbb{R}$ constructed as follows: For every $u, v \in V$, define $d(u, v)$ to be the distance of the* shortest path *from u to v in G with respect to the weight function w.*

We can use proof by contradiction about <u>shortest path properties</u> to show that such metric completion preserves the triangle inequality

(the 3 other metric properties non-negativity, identity, and symmetric can be easily shown)

Details in the PDF

Step 4: Reconstruction

When we do the metric completion, we also remember the actual shortest paths, e.g.,

- Assume **d(i, j) > d(i, k) + d(k, j)** and thus we shorten **d(i, j)** into **d(i, k) + d(k, j)**,
- In the event this 'virtual' edge **(i, j)** is taken in the Metric-ST variant, we actually take edge **(i, k)** and **(k, j)** in the General-ST variant
- Some of these edges may overlap and create cycles, we have to remove some edges as we want a spanning tree, not cycle(s)
  - Thus the cost can be equal or lower in General ST version

Analysis

Theorem: Given an α-approximation algorithm for finding a Metric Steiner tree, we can find an α-approximation for a General Steiner tree

Proof Assume we have a graph $G^g = (V, E^g)$ with required vertices R, Steiner vertices S, and a non-negative edge weight function w. Let T^m be an α-approximate Steiner tree for (R, S, d), where d is the metric completion of G^g. Let T^g be the spanning tree constructed above by converting the edges in T^m into paths in $G^g = (V, E^g)$ and removing cycles. We will argue that T^g is an α-approximation of the minimum cost spanning tree for G.

First, we have shown that $cost_d(T^m) \leq \alpha \cdot cost_w(OPT^g)$. Second, we have shown that $cost_w(T^g) \leq cost_d(T^m)$. Putting the two pieces together, we conclude that $cost_w(T^g) \leq \alpha \cdot cost_w(OPT^g)$, and hence T^g is an α-approximation for the minimum cost Steiner tree for G with respect to w. $\square$

Full details in the PDF

Summary

- Introducing the Steiner Tree problem
  - Similar to the MST problem, but different and *harder*
- Three variants: Euclidean, Metric, General
  - All NP-hard, focus on Metric and General variants
  - Exponential Complete Search solution for General ST
- Approximation algorithm: MST of **R** vertices only
  - OK for Metric, can be awful for General variant verbatim
  - Proof of **2**-Approximation on Metric variant
  - Converting General variant to Metric variant (metric completion) and then using the same **2**-approximation algorithm (**gap preserved**)