

Lecture 2

In which we show the equivalence of metric and general Steiner tree, give a 2-approximate algorithm for both problems, and begin to talk about TSP

1 Approximating the Metric Steiner Tree Problem

In the previous lecture we defined the Metric Steiner Tree Problem as follows: the input is a set $X = R \cup S$ of points, where R is a set of *required* points and S is a set of *optional* points, and a symmetric distance function $d : X \times X \rightarrow \mathbb{R}_{\geq 0}$ that associates a non-negative distance $d(x, y) = d(y, x) \geq 0$ to every pair of points. We restrict the problem to the case in which d satisfies the *triangle inequality*, that is,

$$\forall x, y, z \in X. d(x, z) \leq d(x, y) + d(y, z)$$

In such a case, d is called a (semi-)metric, hence the name of the problem.

The goal is to find a tree $T = (V, E)$, where V is any set $R \subseteq V \subseteq X$ of points that includes all of the required points, and possibly some of the optional points, such that the *cost*

$$\text{cost}_d(T) := \sum_{(u,v) \in E} d(u, v)$$

of the tree is minimized.

In the previous lecture we suggested the following algorithm: completely disregard the optional vertices in S , and find a minimum *spanning* tree of the weighted graph that has vertex set R and edge weights defined by $d(\cdot, \cdot)$.

Now we prove that this algorithm is 2-approximate, that is, it finds a solution whose cost is at most twice the optimal cost.

To do so, we prove the following.

Lemma 1 *Let $(X = R \cup S, d)$ be an instance of the metric Steiner tree problem, and $T = (V, E)$ be a Steiner tree with $R \subseteq V \subseteq X$.*

Then there is a tree $T' = (R, E')$ which spans the vertices in R and only the vertices in R such that

$$\text{cost}_d(T') \leq 2 \cdot \text{cost}_d(T)$$

In particular, applying the Lemma to the optimal Steiner tree we see that there is a spanning tree of R whose cost is at most twice the cost of the optimal Steiner tree. This also means that the minimal spanning tree of R also has cost at most twice the cost of the optimal Steiner tree.

PROOF: [Of Lemma 1] Consider a DFS traversal of T , that is, a sequence

$$x_0, x_1, x_2, \dots, x_m = x_0$$

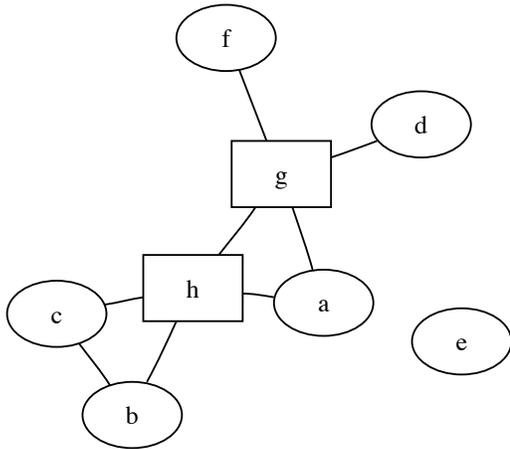
listing the vertices of T in the order in which they are considered during a DFS, including each time we return to a vertex at the end of each recursive call. The sequence describes a cycle over the elements of V whose total length $\sum_{i=0}^m d(x_i, x_{i+1})$ is precisely $2 \cdot \text{cost}_d(T)$, because the cycle uses each edge of the tree precisely twice.

Let now y_0, y_1, \dots, y_k be the sequence obtained from x_0, \dots, x_m by removing the vertices in S and keeping only the first occurrence of each vertex in R .

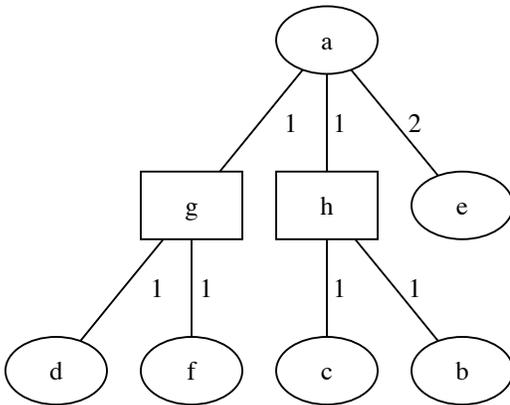
Then y_0, \dots, y_k is a path that includes all the vertices of R , and no other vertex, and its cost $\sum_{i=0}^k d(y_i, y_{i+1})$ is at most the cost of the cycle $x_0, x_1, x_2, \dots, x_m$ (here we are using the triangle inequality), and so it is at most $2 \cdot \text{cost}_d(T)$.

But now note that y_0, \dots, y_k , being a path, is also a tree, and so we can take T' to be tree (R, E') where E' is the edge set $\{(y_i, y_{i+1})\}_{i=0, \dots, k}$. \square

For example, if we have an instance in which $R = \{a, b, c, d, e, f\}$, $S = \{g, h\}$, and the distance function $d(\cdot, \cdot)$ assigns distance 1 to the points connected by an edge in the graph below, and distance 2 otherwise



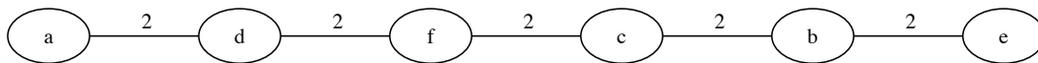
Then the following is a Steiner tree for our input whose cost is 8:



We use the argument in the proof of the Lemma to show that there is a Spanning tree of R of cost at most 16. (In fact we will do better.)

The order in which we visit vertices in a DFS of T is $a \rightarrow g \rightarrow d \rightarrow g \rightarrow f \rightarrow g \rightarrow a \rightarrow h \rightarrow c \rightarrow h \rightarrow b \rightarrow h \rightarrow a \rightarrow e \rightarrow a$. If we consider it as a loop that starts at a and goes back to a after touching all vertices, some vertices more than once, then the loop has cost 16, because it uses every edge exactly twice.

Now we note that if we take the DFS traversal, and we skip all the optional vertices and all the vertices previously visited, we obtain an order in which to visit all the required vertices, and no other vertex. In the example the order is $a \rightarrow d \rightarrow f \rightarrow c \rightarrow b \rightarrow e$.



Because this path was obtained by “shortcutting” a path of cost at most twice the cost of T , and because we have the triangle inequality, the path that we find has also cost at most twice that of T . In our example, the cost is just 10. Since a path is, in particular, a tree, we have found a spanning tree of R whose cost is at most twice the cost of T .

The factor of 2 in the lemma cannot be improved, because there are instances of the Metric Steiner Tree problem in which the cost of the minimum spanning tree of R is, in fact, arbitrarily close to twice the cost of the minimum steiner tree.

Consider an instance in which $S = \{v_0\}$, $R = \{v_1, \dots, v_n\}$, $d(v_0, v_i) = 1$ for $i = 1, \dots, n$, and $d(v_i, v_j) = 2$ for all $1 \leq i < j \leq n$. That is, consider an instance in which the required points are all at distance two from each other, but they are all at distance one from the unique optional point. Then the minimum Steiner tree has v_0 as a root and the nodes v_1, \dots, v_n as leaves, and it has cost n , but the minimum spanning tree of R has cost $2n - 2$, because it is a tree with n nodes and $n - 1$ edges, and each edge is of cost 2.

2 Metric versus General Steiner Tree

The General Steiner Tree problem is like the Metric Steiner Tree problem, but we allow arbitrary distance functions.

In this case, it is not true any more that a minimum spanning tree of R gives a good approximation: consider the case in which $R = \{a, b\}$, $S = \{c\}$, $d(a, b) = 10^{100}$, $d(a, c) = 1$ and $d(b, c) = 1$. Then the minimum spanning tree of R has cost 10^{100} while the minimum Steiner tree has cost 2.

We can show, however, that our 2-approximation algorithm for Metric Steiner Tree can be turned, with some care, into a 2-approximation algorithm for General Steiner Tree.

Lemma 2 *For every $c \geq 1$, if there is a polynomial time c -approximate algorithm for Metric Steiner Tree, then there is a polynomial time c -approximate algorithm for General Steiner Tree.*

PROOF: Suppose that we have a polynomial-time c -approximate algorithm A for Metric Steiner Tree and that we are given in input an instance $(X = R \cup S, d)$ of General Steiner Tree. We show how to find, in polynomial time, a c -approximate solution for (X, d) .

For every two points $x, y \in X$, let $d'(x, y)$ be the length of a shortest path from x to y in the weighted graph of vertex set X of weights $d(\cdot, \cdot)$. Note that $d'(\cdot, \cdot)$ is a distance function that satisfies the triangle inequality, because for every three points x, y, z it

must be the case that the length of the shortest path from x to z cannot be any more than the length of the shortest path from x to y plus the length of the shortest path from y to z .

This means that (X, d') is an instance of Metric Steiner Tree, and we can apply algorithm A to it, and find a tree $T' = (V', E')$ of cost

$$\text{cost}_{d'}(T') \leq c \cdot \text{opt}(X, d')$$

Now notice that, for every pair of points, $d'(x, y) \leq d(x, y)$, and so if T^* is the optimal tree of our original input (X, d) we have

$$\text{opt}(X, d') \leq \text{cost}_{d'}(T^*) \leq \text{cost}_d(T^*) = \text{opt}(X, d)$$

So putting all together we have

$$\text{cost}_{d'}(T') \leq c \cdot \text{opt}(X, d)$$

Now, from T' , construct a graph $G = (V, E)$ by replacing each edge (x, y) by the shortest path from x to y according to $d(\cdot)$. By our construction we have

$$\text{cost}_d(G) = \sum_{(x,y) \in E} d(x, y) \leq \sum_{(x,y) \in E'} d'(x, y) = \text{cost}_{d'}(T')$$

Note also that G is a connected graph.

The reason why we have an inequality instead of an equality is that certain edges of G might belong to more than one shortest path, so they are counted only once on the left-hand side.

Finally, take a minimum spanning tree T of G according to the weights $d(\cdot, \cdot)$. Now T is a valid Steiner tree, and we have

$$\text{cost}_d(T) \leq \text{cost}_d(G) \leq c \cdot \text{opt}(X, d)$$

□

3 The Traveling Salesman Problem

In the Traveling Salesman Problem (abbreviated TSP) we are given a set of points X and a symmetric distance function $d : X \times X \rightarrow \mathbb{R}_{\geq 0}$. The goal is to find a cycle that reaches all points in X and whose total length is as short as possible.

For example, a shuttle driver that picks up seven people at SFO and needs to take them to their home and then go back to the airport faces a TSP instance in which X includes eight points (the seven home addresses and the airport), and the distance function is the driving time between two places. A DHL van driver who has to make a series of delivery and then go back to the warehouse has a similar problem. Indeed TSP is a basic model for several concrete problems, and it one of the most well studied problems in combinatorial optimization.

We can distinguish four versions of TSP. One distinction is whether we work with the Metric version, in which we restrict ourselves to inputs in which $d(\cdot, \cdot)$ satisfies the triangle inequality, or with the General version in which $d(\cdot, \cdot)$ can be arbitrary. Another distinction is whether we want to cycle to reach every point *at least* once, or whether we are only looking for solutions in which every point is reached *exactly* once.

It can be proved that, unless $P = NP$, no approximation is possible for the General TSP without repetitions.

Next time we will show that the other three versions (Metric TSP without repetitions, General TSP with repetitions, and Metric TSP with repetitions), are all equivalent from the point of view of approximation.