An Approximation Algorithm for the Euclidean Bottleneck Steiner Tree Problem

May 4, 2014

Abstract

Given two sets of points in the plane, $P$ of $n$ terminals and $S$ of $m$ Steiner points, a Steiner tree of $P$ is a tree spanning all points of $P$ and some (or none or all) points of $S$. A Steiner tree with length of longest edge minimized is called a bottleneck Steiner tree. In this paper, we study the Euclidean bottleneck Steiner tree problem: given two sets, $P$ and $S$, and a positive integer $k \leq m$, find a bottleneck Steiner tree of $P$ with at most $k$ Steiner points. The problem has application in the design of wireless communication networks.

We first show that the problem is NP-hard and cannot be approximated within factor $\sqrt{2}$, unless $P = NP$. Then, we present a polynomial-time approximation algorithm with performance ratio 2.

1 Introduction

Consider a wireless communication network with $n$ stations, each station has a limited power so that it can only communicate with stations within a limited range, and suppose that, in order to make the network connected and due to budget limits, we are only allowed to put at most $k$ new stations in given potential locations in this network. Clearly, we would like to select locations such that distance between stations as small as possible. This application motivates the following problem:

**The Bottleneck Steiner Tree ($k$-BST) problem.** Given two sets in the plane, $P$ of terminal points and $S$ of Steiner points, and a positive integer $k$, one is asked to find Steiner tree $T$ of $P$ with at most $k$ Steiner points, such that the bottleneck (i.e., length of the longest edge) of $T$ is minimized.

In the classical Steiner tree (ST) problem, the goal is to find a Steiner tree $T$ such the total length of edges of $T$ is minimized. This problem has been shown to be NP-complete [7] and many approximation algorithms have been proposed [2, 3, 9, 13]. In the bottleneck Steiner tree (BST) problem, the goal is to find a Steiner tree $T$ such the bottleneck of $T$ is minimized. Sarrafzadeh and Wong [14] showed that this problem can be solved in polynomial time.
A version of the $k$-BST, where $S$ is the whole plane $\mathbb{R}^2$, has been studied extensively in the last decade. In [15], this version was shown to be NP-hard to approximate within ratio $\sqrt{2}$. The best known upper bound on approximation ratio is 1.866 [16]. Bae et al. [1] presented an $O(n \log n)$ time algorithm to the problem for $k = 1$ and an $O(n^2)$ time algorithm for $k = 2$. Li et al. [12] presented a $(\sqrt{2} + \epsilon)$-approximation algorithm with inapproximability within $\sqrt{2}$ for a special case of the problem where there should be no edge connecting any two Steiner points in the optimal solution. These versions have many important applications in VLSI design, network communication and computational biology [4,6,10,11].

We are not aware of any previous work studying our version. However, in this paper, we show that the $k$-BST problem is NP-hard and we present a polynomial-time algorithm with constant factor approximation ratio for the problem.

2 Hardness Result

Given a set $P$ of $n$ terminals, a set $S$ of $m$ Steiner points and an integer $k \leq m$, the goal in the $k$-BST problem is to find a Steiner tree with at most $k$ Steiner points from $S$ and bottleneck as small as possible. In this section we prove hardness of the problem.

Theorem 2.1. The $k$-BST problem is NP-hard. Moreover, the $k$-BST problem cannot be approximated within $\sqrt{2}$ in polynomial time, unless $P = NP$.

Proof: Our proof is based on a reduction from the following problem which is known to be NP-complete due to Garey and Johnson [8].

Connected vertex cover in planar graphs with maximum degree 4. Given a planar graph $G = (V,E)$ with no vertex degree exceeding 4 and an integer $k$, does there exist a vertex cover $V^*$ for $G$ such that $|V^*| \leq k$ and the subgraph of $G$ induced by $V^*$ is connected?

Given a planar graph $G = (V,E)$ with no vertex degree exceeding 4 and an integer $k$, we construct two planar sets $\mathcal{P}$ and $\mathcal{S}$ and compute an integer $k'$, such that $G$ has a connected vertex cover of size at most $k$ if and only if there exists a Steiner tree $T$ with at most $k'$ Steiner points and bottleneck at most 1.

Let $V = \{v_1, v_2, \ldots, v_n\}$ and let $E = \{e_1, e_2, \ldots, e_m\}$. We first embed $G$ into a rectangular grid, with distance 2 between adjacent vertices, as follows. Each vertex $v_i \in V$ corresponds to some grid vertex and each edge $e = (v_i, v_j) \in E$ corresponds to a rectilinear path $p_e$, consisting of some horizontal and vertical elementary grid segments, whose endpoints are the grid vertices corresponding to $v_i$ and $v_j$. In addition, these paths are pairwise disjoints; see Figure 1.

Let $V' = \{v'_1, v'_2, \ldots, v'_n\}$ be the set of vertices of the grid corresponding to the vertices of $V$, and let $E' = \{p_{e_1}, p_{e_2}, \ldots, p_{e_m}\}$ be the set of edges (paths) corresponding to the
Figure 1: (a) A planar graph $G = (V, E)$, (b) the embedded graph $G' = (V', E')$, and (c) the produced sets: $S = V'$ consists of solid circles, $P$ consists of empty circles.

edges of $E$. Let $|p_e|$ denote the total length of the grid segments of $p_e$. We place $|p_e| - 1$ terminals on $p_e$, such that the distance between any adjacent points is exactly 1, denote by $t(e)$ these terminals; see Figure 1(c). Finally, we set $P = \bigcup_{e \in E} t(e)$, $S = V'$ and $k' = k$.

Now, we prove the correctness of the reduction. Clearly, if $G$ has a connected vertex cover $V^*$ with $|V^*| \leq k$, then, by selecting the Steiner points from $S$ that are corresponded to the nodes in $V^*$, we can construct a Steiner tree $T$ of $P$ with at most $k'$ Steiner points, such that the length of each edge in $T$ is exactly 1.

Conversely, suppose that there exists a Steiner tree $T$ of $P$ with at most $k'$ Steiner points and bottleneck at most 1. Let $V^*$ be the subset of points of $V' = S$ that belong to $T$. By the construction of $P$, every two terminals that belong to different edges in $E'$ have distance greater than 1. Thus, we deduce that each terminal in $p_e \in E'$ is connected to either another terminal in $p_e$ or a Steiner point from $V^*$. Moreover, since $T$ is connected, each edge in $E'$ contains at least one terminal that is connected to at least one Steiner point from $V^*$. This implies that the set of nodes in $G$ corresponding to the points in $V^*$ is a connected vertex cover of $G$, and its size is at most $k = k'$.

Finally, it is not hard to see that if there exists a polynomial-time algorithm that computes a Steiner tree with at most $k$ Steiner point and bottleneck at most $(\sqrt{2} - \varepsilon)$, then we can show that $G$ has a connected vertex cover of size at most $k$.

3 2-Approximation Algorithm

In this section, we develop a polynomial-time approximation algorithm for computing a Steiner tree with at most $k$ Steiner points ($k$-ST for short) such that its bottleneck is at most 2 times the bottleneck of an optimal (minimum-bottleneck) $k$-ST.

Let $G = (V, E)$ be the complete graph over $V = P \cup S$. We assume, without loss of generality, that $E = \{e_1, e_2, \ldots, e_l\}$ such that $|e_1| \leq |e_2| \leq \ldots \leq |e_l|$. It is not hard to
see that the bottleneck of an optimal $k$-ST is a length of an edge from $E$. For an edge $e_i \in E$, let $G_i = (V, E_i)$ be the graph with $E_i = \{ e_j \in E : |e_j| \leq |e_i| \}$. The idea behind our algorithm is to devise a procedure that, for a given edge $e_i \in E$, does one of the following:

(i) It constructs a $k$-ST of $P$ in $G$ with bottleneck at most $2 \times |e_i|$.

(ii) It returns the information that $G_i$ does not contain any $k$-ST of $P$.

For two points $p, q \in P$, let $\delta_i(p, q)$ be a shortest Steiner path between $p$ and $q$ in $G_i$, i.e., a path connecting $p$ and $q$ with minimum number of Steiner points in $G_i$. Let $G_P = (P, E_P)$ be the complete graph over $P$. For each edge $(p, q)$ in $E_P$, we assign a weight $w(p, q)$ equal to the number of Steiner points in $\delta_i(p, q)$. Let $T$ be a minimum spanning tree of $G_P$ under $w$, and let $C(T) = \sum_{e \in T} \lfloor w(e)/2 \rfloor$. The following observation follows from Lemma 3 in [15].

Observation 3.1. For any spanning tree $T'$ of $G_P$, $C(T) \leq C(T')$.

Lemma 3.2. If $G_i$ contains a $k$-ST of $P$, then $C(T) \leq k$.

**Proof:** Let $T^*$ be a $k$-ST of $P$ in $G_i$. A Steiner tree is full if all terminals are leaves. We decompose $T^*$ into a union of full trees. For each full tree $T_j^*$ of $T^*$, we will construct a spanning tree $T'_j$ of the terminals of $T_j^*$ in $G_P$, such that the union of these trees is a spanning tree $T'$ of $P$ in $G_P$ with $C(T') \leq k$.

We arbitrary select a Steiner point as the root of $T_j^*$; see Figure 2(a). The construction of $T'_j$ is bottom-up by an iterative process. In each iteration, we select the deepest leaf $p$ in the rooted tree, which is a terminal, and we connect it to its nearest terminal $q$ by an edge of weight equal to the number of Steiner points between them. Let $s$ be the first common parent of $p$ and $q$. We then remove the Steiner points between $p$ and $s$ (in the last iteration, we may remove all of the remaining points).

![Figure 2: (a) The rooted tree $T_j^*$, and (b) the construction of $T'_j$.](image)

In the example in Figure 2(b), we first select the terminal $a$, which is the deepest one, we connect it to the terminal $b$ by an edge of weight 3 and we remove the points $s_1$ and
Next, we select the terminal \( s_2 \), we connect it to the terminal \( c \) by an edge of weight 2 and we remove the point \( s_3 \). In the last iteration, we select the terminal \( b \), we connect it to the terminal \( c \) by an edge of weight 3 and we remove all of the remaining points.

Notice that, since, in each iteration, we select the deepest terminal, we add an edge \((p,q)\), of weight \( w(p,q) \), and we remove at least \( \lfloor w(p,q)/2 \rfloor \) Steiner points from \( T_j^* \). This implies that \( C(T'_j) = \sum_{e \in T'_j} \lfloor w(e)/2 \rfloor \leq k_j \), where \( k_j \) is the number of Steiner points in \( T_j^* \). Moreover, the union \( T' \) of the trees \( T_j^* \) is a spanning tree of \( G' \) and has \( C(T') \leq k \). Thus, since \( T \) is a minimum spanning tree of \( G' \), by Observation 3.1, we have \( C(T) \leq C(T') \leq k \).

We now describe our approximation algorithm. We traverse the edges of \( E \) in the sorted order and, for each edge \( e_i \in E \), we construct a minimum spanning tree \( T \) of \( G_P = (P,E_P) \) and check whether \( C(T) \leq k \). If so, we construct a \( k \)-ST of \( P \), otherwise, we move to the next edge \( e_{i+1} \).

**Algorithm 1 EBST**

\[
\begin{align*}
1: & \quad C(T) \leftarrow \infty \\
2: & \quad G_P = (P,E_P) \leftarrow \text{the complete graph over } P \\
3: & \quad i \leftarrow 0 \\
4: & \quad \textbf{while } C(T) > k \textbf{ do} \\
5: & \quad \quad i \leftarrow i + 1 \\
6: & \quad \quad \textbf{for each edge } (p,q) \in E_P \textbf{ do} \\
7: & \quad \quad \quad w(p,q) \leftarrow \text{the number of Steiner points in } \delta_i(p,q) \\
8: & \quad \quad \quad \textbf{construct a minimum spanning tree } T \textbf{ of } G_P \textbf{ under } w \\
9: & \quad \quad \quad C(T) \leftarrow \sum_{e \in T} \lfloor w(e)/2 \rfloor \\
10: & \quad \quad \textbf{Construct-}k\text{-ST}(T,G_i) \\
\end{align*}
\]

The construction of a \( k \)-ST is done as follows. For each edge \( e = (p,q) \in T \), we select \( \lfloor w(e)/2 \rfloor \) Steiner points on any shortest Steiner path between \( p \) and \( q \) in \( G_i \), such that, the path from \( p \) to \( q \) that passes through these points has a bottleneck at most \( 2|e_i| \), and we connect these points to form a path; see Figure 3. Clearly, the obtained Steiner tree contains at most \( k \) Steiner points and its bottleneck is at most \( 2|e_i| \).

**Lemma 3.3.** The algorithm above constructs a \( k \)-ST of \( P \) with bottleneck at most 2 times the bottleneck of an optimal \( k \)-ST.

**Proof:** Let \( e_i \) be the first edge satisfying the condition \( C(T) \leq k \). Thus, by Lemma 3.2, the bottleneck of any \( k \)-ST in \( G \) is at least \( |e_i| \), and, therefore, the constructed \( k \)-ST has a bottleneck at most 2 times the bottleneck of an optimal \( k \)-ST.

**Lemma 3.4.** The algorithm above has a polynomial running time.
Figure 3: The constructed $k$-ST consists of solid circles and dotted lines.

**Proof:** $G_i$ can be constructed in $O((n + m)^2)$ time. In order to construct the graph $G_P$, we can compute in $O((n + m)^3)$ time the shortest Steiner paths between each pair of points in $P$ [5]. Once $G_P$ is constructed, computing a minimum spanning tree of $G_P$ can be done in $O(n^2)$ time, and selecting the relevant Steiner points can be done in $O(k(n + m))$ time.

By combining Lemma 3.3 and Lemma 3.4, we get the following theorem.

**Theorem 3.5.** There exists a polynomial-time approximation algorithm with performance ratio 2 for the $k$-BST problem.

4 Conclusion

In this paper, we studied the problem of finding bottleneck Steiner trees in the Euclidean plane. We proved that the $k$-BST problem in the plane does not admit any approximation algorithm with performance ratio less than $\sqrt{2}$, unless $P = NP$, and that there exists a polynomial-time approximation algorithm with performance ratio 2. It would be interesting to find better approximation algorithm for the $k$-BST problem. Another interesting question is how efficient can one solve the $k$-BST problem for a constant $k > 0$?

References


