Bottleneck Steiner Tree with Bounded Number of Steiner Vertices

A. Karim Abu-Affash* Paz Carmi† Matthew J. Katz‡

Abstract

Given a complete graph $G = (V, E)$, where each vertex is labeled either terminal or Steiner, a distance function $d : E \rightarrow \mathbb{R}^+$, and a positive integer k, we study the problem of finding a Steiner tree T spanning all terminals and at most k Steiner vertices, such that the length of the longest edge is minimized. We first show that this problem is NP-hard and cannot be approximated within a factor $2 - \varepsilon$, for any $\varepsilon > 0$, unless $P = NP$. Then, we present a polynomial-time 2-approximation algorithm for this problem.

1 Introduction

Given an arbitrary weighted graph $G = (V, E)$ with a distinguished subset $R \subseteq V$ of vertices, a *Steiner tree* is an acyclic subgraph of G spanning all vertices of R. The vertices of R are usually referred to as *terminals* and the vertices of $V \setminus R$ as *Steiner* vertices. The *Steiner tree* (ST) problem is to find a Steiner tree T such that the total length of the edges of T is minimized. This problem has been shown to be NP-complete [4, 10], even in the Euclidean or rectilinear version [11]. Arora [3] gave a PTAS for the Euclidean and rectilinear versions of the ST problem. For arbitrary weighted graphs, many approximation algorithms have been proposed [6, 7, 12, 15, 17, 18].

The *bottleneck Steiner tree* (BST) problem is to find a Steiner tree T such that the bottleneck (i.e., the length of the longest edge) of T is minimized. Unlike the ST problem, the BST problem can be solved exactly in polynomial time [19]. Both the ST and BST problems have many important applications in VLSI design, transportation and other networks, and computational biology [8, 9, 13, 14].

The *k-Bottleneck Steiner Tree* (k-BST) problem is a restricted version of the BST problem, in which there is a limit on the number of Steiner vertices that may be used in the constructed tree. More precisely, given a graph $G = (V, E)$ and a subset $R \subseteq V$ of terminals, a distance function $d : E \rightarrow \mathbb{R}^+$, and a positive integer k, one has to find a Steiner tree T with at most k Steiner vertices such that the bottleneck of T is minimized.

A geometric version of the k-BST problem has been studied in [20]. In this version, we are given a set P of n terminals in the plane and an integer $k > 0$, and we are asked to place at most k Steiner points, such that the obtained Steiner tree has bottleneck as small as possible. Wang and Du [20] showed that the problem is NP-hard to approximate within a factor of $\sqrt{2}$. The best known approximation ratio is 1.866 [21]. Bae et al. [5] presented an $\mathcal{O}(n \log n)$-time algorithm for the problem for $k = 1$ and an $\mathcal{O}(n^2)$-time algorithm for $k = 2$. Li et al. [16] presented a $(\sqrt{2} + \varepsilon)$-approximation algorithm with inapproximability within $\sqrt{2}$ for a special case of the problem where edges between two Steiner points are not allowed.

Recently, Abu-Affash [1] studied the k-BST problem with the additional requirement that all terminals in the computed Steiner tree must be leaves. He presented a hardness result for the problem, as well as a polynomial-time approximation algorithm with performance ratio 4. In [2], the authors considered the following related problem. Given a set P of n points in the plane and two points $s, t \in P$, locate k Steiner points, so as to minimize the bottleneck of a bottleneck path between s and t. They showed how to solve this problem optimally in time $\mathcal{O}(n \log^2 n)$.

In this paper, we show that the k-BST problem is NP-hard and that it cannot be approximated to within a factor of $2 - \varepsilon$. We also present a polynomial-time 2-approximation algorithm for the problem.

2 Hardness Result

Given a complete graph $G = (V, E)$ with a distinguished subset $R \subseteq V$ of terminals, a distance function $d : E \rightarrow \mathbb{R}^+$, and a positive integer k, the goal in the k-BST problem is to find a Steiner tree with at most k Steiner vertices and bottleneck as small as possible. In this section we prove a lower bound on the approximation

*Department of Computer Science, Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel, abuaffas@cs.bgu.ac.il. Partially supported by the Lynn and William Frankel Center for Computer Sciences, by the Robert H. Arnow Center for Bedouin Studies and Development, by a fellowship for outstanding doctoral students from the Planning & Budgeting Committee of the Israel Council for Higher Education, and by a scholarship for advanced studies from the Israel Ministry of Science and Technology.

†Department of Computer Science, Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel, carmip@cs.bgu.ac.il. Partially supported by a grant from the German-Israeli Foundation, the Lynn and William Frankel Center for Computer Sciences

‡Department of Computer Science, Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel, matya@cs.bgu.ac.il. Partially supported by grant 1045/10 from the Israel Science Foundation, and by the Israel Ministry of Industry, Trade and Labor (consortium CORNET).

ratio of polynomial-time approximation algorithms for the problem.

Theorem 1 *It is NP-hard to approximate the k-BST problem within a factor $2 - \varepsilon$, for any $\varepsilon > 0$.*

Proof. We present a reduction from connected vertex cover in planar graphs, which is known to be NP-complete [11].

Connected vertex cover in planar graphs: Given a planar graph $G = (V, E)$ and an integer k, does there exist a vertex cover V^* of G, such that $|V^*| \le k$ and the subgraph of G induced by V^* is connected?

Given a planar graph $G = (V, E)$ and an integer k, we construct a complete graph $G' = (V', E')$ with an appropriate distance function and appropriate 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 in G' with at most k' Steiner vertices and bottleneck at most $(2 - \varepsilon)$, for some $\varepsilon > 0$.

Let $V = \{v_1, v_2, \ldots, v_n\}$ and let $E = \{e_1, e_2, \ldots, e_m\}$. For each edge $e = (v_i, v_j) \in E$, we add a vertex $t_{i,j}$ (e.g., at the middle of e) and connect it to both v_i and v_j. Let $R = \{t_{i,j} : (v_i, v_j) \in E\}$ and let $E'_1 = \{(v_i, t_{i,j}), (t_{i,j}, v_j) : (v_i, v_j) \in E\}$. We set $V' = V \cup R$, where V is the set of Steiner vertices and R is the set of terminals; see Figure 1. Let $G' = (V', E')$ be the complete graph over V'. For each edge $e \in E'$, we assign length $d(e) = 1$, if $e \in E'_1$, and $d(e) = 2$, otherwise. Finally, we set $k' = k$.

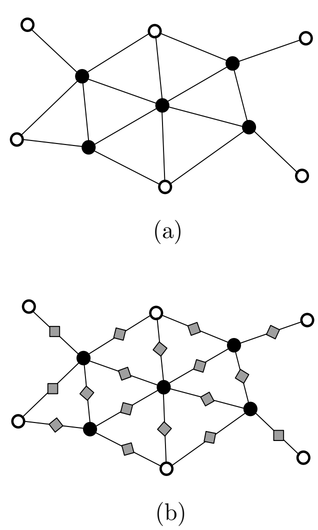

Figure 1: (a) A planar graph $G = (V, E)$, and (b) the vertices of G': circles indicate Steiner vertices and grey squares indicate terminals.

Now, we prove the correctness of the reduction. Clearly, if G has a connected vertex cover V^* with $|V^*| \le k$, then, by selecting the Steiner vertices of V' corresponding to the vertices in V^*, we can construct a Steiner tree T with at most $k' = k$ Steiner vertices, such that the length of each edge in T is exactly 1.

Conversely, suppose that there exists a Steiner tree T in G' with at most k' Steiner vertices and bottleneck at most $2 - \varepsilon$. Let V^* be the subset of vertices of V that belong to T. By the construction, any two terminals are connected in E' by an edge of length 2. Thus, we deduce that each terminal is connected in T to a Steiner vertex in V^*. Since T is connected and each edge in E corresponds to one terminal in V', we conclude that V^* is a connected vertex cover of G, and its size is at most $k = k'$. □

3 2-Approximation Algorithm

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

Let $G = (V, E)$ be the complete graph with n vertices, let $R \subseteq V$ be the set of terminals, and let $d : E \to \mathbb{R}^+$ be a distance function. Let $e_1, e_2, \ldots, e_m$, where $m = \binom{n}{2}$, be the edges of G sorted by length, that is, $d(e_1) \le d(e_2) \le \cdots \le d(e_m)$. Clearly, the bottleneck of an optimal k-ST is the length of an edge in E. For an edge $e_i \in E$, let $G_i = (V, E_i)$ be the graph obtained from G by deleting all edges of length greater than $d(e_i)$, that is, $E_i = \{e_j \in E : d(e_j) \le d(e_i)\}$.

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 either constructs a k-ST in G with bottleneck at most $2d(e_i)$, or

(ii) it returns the information that G_i does not contain a k-ST.

Let $e_i \in E$. For two terminals $p, q \in R$, let $\delta_i(p, q)$ be a path between p and q in G_i with minimum number of Steiner vertices. Let $G_R = (R, E_R)$ be the complete graph over R. For each edge $(p, q) \in E_R$, we assign a weight $w(p, q)$ equal to the number of Steiner vertices in $\delta_i(p, q)$. Let T be a minimum spanning tree of G_R under w, and put $C(T) = \sum_{e \in T} \lfloor w(e)/2 \rfloor$. The following observation follows from Lemma 3 in [20].

Observation 1 *For any spanning tree T' of G_R, $C(T) \le C(T')$.*

Lemma 2 *If G_i contains a k-ST, then $C(T) \le k$.*

Proof. Let T^* be a k-ST in G_i. A Steiner tree is *full* if all its terminals are leaves. It is well known that every Steiner tree can be decomposed into a collection of full Steiner trees, by splitting each of the non-leaf terminals.

We begin by decomposing T^* into a collection of full Steiner trees. Next, for each full Steiner tree T_j^* in the collection, we construct in G_R a spanning tree T_j' of the terminals of T_j^*, such that the union of these trees is a spanning tree T' of G_R and $C(T') \leq k$. Finally, by Observation 1, we conclude that $C(T) \leq k$.

We now describe how to construct T_j' from T_j^*. Arbitrarily select one of the Steiner vertices as the root of T_j^*; see Figure 2(a). The construction of T_j' is done by an iterative process applied to T_j^*. In each iteration, we select a deepest terminal p, among the terminals of the current rooted tree that have not yet been processed. From p we move up the tree until we reach a Steiner vertex s that has terminal descendants other than p. Let q, $q \neq p$, be a terminal descendant of s that is closest to s. We connect p to q by an edge of weight equal to the number of Steiner vertices between p and q in T_j^*, and remove the Steiner vertices between p and s (not including s). After processing all terminals but one, we remove all remaining Steiner vertices.

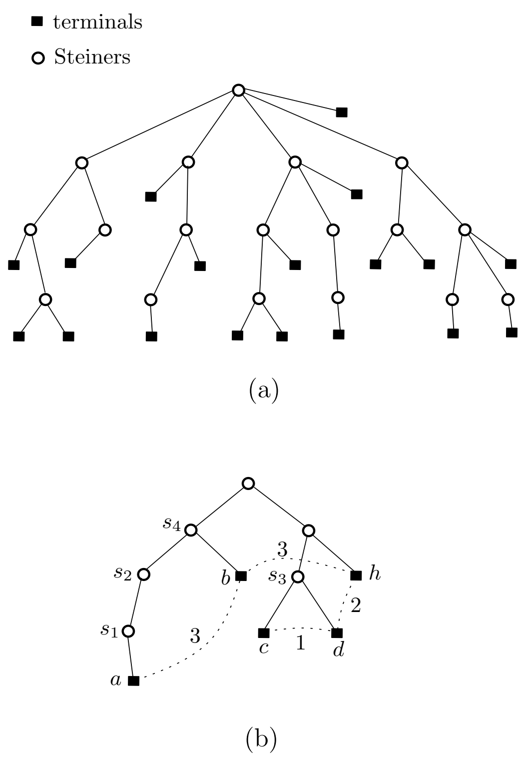

Figure 2: (a) The rooted tree T_j^*, and (b) the construction of T_j'.

In the example in Figure 2(b), we first select terminal a, which is the deepest one, connect it to terminal b by an edge of weight 3, and remove the vertices s_1 and s_2. Next, we select terminal c, connect it to terminal d by an edge of weight 1, and do not remove any Steiner vertex. Next, we select terminal d, connect it to terminal h by an edge of weight 2, and remove the vertex s_3. In the last iteration, we select terminal b, connect it to terminal h by an edge of weight 3 and remove the vertex s_4. We can now remove all of the remaining Steiner vertices.

Clearly, the union T' of the trees T_j' is a spanning tree of G_R. We show below that $C(T') \leq k$. Notice that in each iteration during the construction of T_j', if the weight of the added edge e is $w(e)$, then we remove at least $\lfloor w(e)/2 \rfloor$ Steiner vertices 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 vertices in T_j^*, and, therefore, $C(T') = \sum_j C(T_j') \leq k$. □

We now present our 2-approximation algorithm. We consider the edges of E, one by one, by non-decreasing length. For each edge $e_i \in E$, we construct a minimum spanning tree T of $G_R = (R, E_R)$, using the weight function w induced by G_i, and check whether $C(T) \leq k$. If so, we construct a k-ST in G with bottleneck at most $2d(e_i)$, otherwise, we proceed to the next edge e_{i+1}.

Algorithm 1 $BST(G = (V, E), R, k)$

1: Let $e_1, e_2, \ldots, e_m$ be the edges of E sorted by non-decreasing length
2: $G_R = (R, E_R) \leftarrow$ the complete graph over R
3: $C(T) \leftarrow \infty$
4: $i \leftarrow 0$
5: **while** $C(T) > k$ **do**
6: $\quad i \leftarrow i + 1$
7: $\quad$ construct the graph G_i
8: $\quad$ **for** each edge $(p, q) \in E_R$ **do**
9: $\quad\quad w(p, q) \leftarrow$ the number of Steiner vertices in $\delta_i(p, q)$
10: $\quad$ construct a minimum spanning tree T of G_R under w
11: $\quad C(T) \leftarrow \sum_{e \in T} \lfloor w(e)/2 \rfloor$
12: *Construct-k-ST*(T, G_i)

The construction of a k-ST (line 12 in the algorithm above) is done as follows. For each edge $e = (p, q) \in T$, we select at most $\lfloor w(e)/2 \rfloor$ Steiner vertices from the path $\delta_i(p, q)$, to obtain a path connecting between p and q with at most this number of Steiner vertices and bottleneck at most $2d(e_i)$; see Figure 3. Clearly, the obtained Steiner tree contains at most k Steiner vertices and its bottleneck is at most $2d(e_i)$.

Lemma 3 *The algorithm above constructs a k-ST in G with bottleneck at most twice the bottleneck of an optimal k-ST.*

Proof. Let e_i be the first edge satisfying the condition $C(T) \leq k$. Then, by Lemma 2, the bottleneck of any k-ST in G is at least $d(e_i)$, and, therefore, the constructed k-ST has a bottleneck at most twice the bottleneck of an optimal k-ST. □

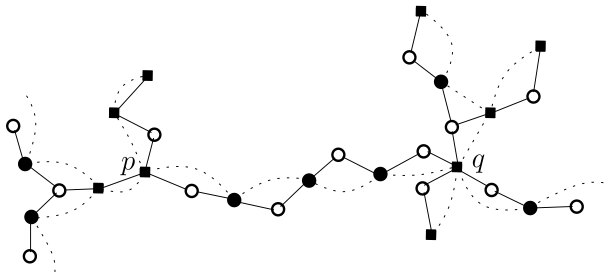

Figure 3: The constructed k-ST consists of the squares, solid circles and dotted edges.

The following theorem summarizes the main result of this section.

Theorem 4 *There exists a polynomial-time 2-approximation algorithm for the k-BST problem.*

References

[1] A.K. Abu-Affash. On the Euclidean bottleneck full Steiner tree problem. In *Proceedings of the 27th ACM Symposium on Computational Geometry (SoCG '11)*, pages 433–439, 2011.

[2] A.K. Abu-Affash, P. Carmi, M.J. Katz, and M. Segal. The Euclidean bottleneck Steiner path problem. In *Proceedings of the 27th ACM Symposium on Computational Geometry (SoCG '11)*, pages 440–447, 2011.

[3] S. Arora. Polynomial time approximation schemes for Euclidean TSP and other geometric problems. *Journal of the ACM*, 45:735–782, 1998.

[4] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. In *Proceedings of the 33rd Annual Symposium on Foundations of Computer Science (FOCS '92)*, pages 14–23, 1992.

[5] S.W. Bae, C. Lee, and S. Choi. On exact solutions to the Euclidean bottleneck Steiner tree problem. *Information Processing Letters*, 110:672–678, 2010.

[6] P. Berman and V. Ramaiyer. Improved approximation for the Steiner tree problem. *Journal of Algorithms*, 17:381–408, 1994.

[7] A. Borchers and D.Z. Du. The k-Steiner ratio in graphs. *SIAM Journal on Computing*, 26:857–869, 1997.

[8] X. Cheng and D.Z. Du. *Steiner Tree in Industry.* Kluwer Academic Publishers, Dordrecht, Netherlands, 2001.

[9] D.Z. Du, J.M. Smith, and J.H. Rubinstein. *Advances in Steiner Tree.* Kluwer Academic Publishers, Dordrecht, Netherlands, 2000.

[10] M.R. Garey, R.L. Graham, and D.S. Johnson. The complexity of computing Steiner minimal trees. *SIAM Journal of Applied Mathematics*, 32(4):835–859, 1977.

[11] M.R. Garey and D.S. Johnson. The rectilinear Steiner tree problem is NP-complete. *SIAM Journal of Applied Mathematics*, 32(4):826–834, 1977.

[12] S. Hougardy and H.J. Prömel. A 1.598 approximation algorithm for the Steiner problem in graphs. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '00)*, pages 448–453, 1999.

[13] F.K. Hwang, D.S. Richards, and P. Winter. *The Steiner Tree Problem.* Annals of Discrete Mathematics, Amsterdam, 1992.

[14] A.B. Kahng and G. Robins. *On Optimal Interconnection for VLSI.* Kluwer Academic Publishers, Dordrecht, Netherlands, 1995.

[15] M. Karpinski and A. Zelikovsky. New approximation algorithms for the Steiner tree problem. *Journal of Combinatorial Optimization*, 1(1):47–65, 1997.

[16] Z.-M. Li, D.-M. Zhu, and S.-H. Ma. Approximation algorithm for bottleneck Steiner tree problem in the Euclidean plane. *Journal of Computer Science and Technology*, 19(6):791–794, 2004.

[17] H.J. Prömel and A. Steger. A new approximation algorithm for the Steiner tree problem with performance ratio 5/3. *Journal of Algorithms*, 36(1):89–101, 2000.

[18] G. Robbins and A. Zelikovsky. Tighter bounds for graph Steiner tree approximation. *SIAM Journal on Discrete Mathematics*, 19(1):122–134, 2005.

[19] M. Sarrafzadeh and C.K. Wong. Bottleneck Steiner trees in the plane. *IEEE Transactions on Computers*, 41(3):370–374, 1992.

[20] L. Wang and D.-Z. Du. Approximations for a bottleneck Steiner tree problem. *Algorithmica*, 32:554–561, 2002.

[21] L. Wang and Z.-M. Li. Approximation algorithm for a bottleneck k-Steiner tree problem in the Euclidean plane. *Information Processing Letters*, 81:151–156, 2002.