ELSEVIER

Information Processing Letters 81 (2002) 151–156

Information Processing Letters

www.elsevier.com/locate/ipl

An approximation algorithm for a bottleneck k-Steiner tree problem in the Euclidean plane

Lusheng Wang *, Zimao Li

Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong

Received 1 December 2000; received in revised form 27 March 2001
Communicated by F.Y.L. Chin

Abstract

We study a *bottleneck Steiner tree problem*: given a set $P = \{p_1, p_2, \ldots, p_n\}$ of n terminals in the Euclidean plane and a positive integer k, find a Steiner tree with at most k Steiner points such that the length of the longest edges in the tree is minimized. The problem has applications in the design of wireless communication networks. We give a ratio-1.866 approximation algorithm for the problem. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Algorithmical approximation; Algorithms; Steiner trees

1. Introduction

Given a set of terminals in the plane, a *Steiner tree* is an acyclic network interconnecting the terminals. Every vertex in a Steiner tree other than terminals is called a *Steiner point*. In this paper, we study a *bottleneck Steiner tree problem* that is defined as follows: given a set $P = \{p_1, p_2, \ldots, p_n\}$ of n terminals and a positive integer k, we want to find a Steiner tree with at most k Steiner points such that the length of the longest edges in the tree is minimized. Contrary to the classic Steiner tree problem, degree-2 Steiner points are allowed in the bottleneck Steiner tree problem. Here we want to minimize the length of the longest edges in the tree. The problem has applications in the design of wireless communication networks [7]. Due to budget limit, we could put at most $n + k$ stations in the plane, n of which must be located at given points. We would like to have the length of the longest edge in the tree to be minimized. There are some other related variations and applications, e.g., the *minimum number of Steiner points and bounded edge-length*, see [1–4,6] for details.

The problem is NP-hard. In [7], it is shown that unless P = NP, the problem cannot be approximated in polynomial time within performance ratios 2 and $\sqrt{2}$ in the rectilinear plane and the Euclidean plane, respectively. Moreover, a ratio-2 approximation algorithm was given for both the rectilinear plane and the Euclidean plane [7]. For the rectilinear plane, the performance ratio is the best possible. For the Euclidean plane, the gap between the lower bound $\sqrt{2}$ and the upper bound 2 is still big.

In this paper, we give an approximation algorithm with performance ratio 1.866 for the Euclidean plane.

* Corresponding author.
E-mail addresses: lwang@cs.cityu.edu.hk (L. Wang), lizm@cs.cityu.edu.hk (Z. Li).

PII: S0020-0190(01)00209-5

2. The performance ratio

A *full component* of a Steiner tree is a subtree in which each terminal is a leaf. A Steiner tree for n terminals is a *k-restricted Steiner tree* if each full component spans at most k terminals. We will show that there exists a 3-restricted Steiner tree containing the same number of Steiner points as in an optimal solution such that the length of the longest edge is at most 1.866 times the optimum. Without loss of generality, we assume that the length of the longest edge is 1.

Let a and b be two points in the plane. ab denotes the segment from a to b and $|ab|$ denotes the length of the segment ab.

Theorem 1. *There exists a* 3*-restricted Steiner tree of k Steiner points such that the length of the longest edge in the tree is at most* 1.866 *times the optimum.*

Proof. Let T be an optimal tree. We assume that T is rooted. (We can arbitrarily select a Steiner point as the root of T.) We will modify T bottom up into a 3-restricted Steiner tree without increasing the number of Steiner points such that the length of the longest edge is at most 1.866 times the optimum. Without loss of generality, we assume that T is a full component, i.e., every internal node in T is a Steiner point and every leaf in T is a terminal.

We organize the nodes in T level by level (ignoring degree-2 Steiner points). Level 1 is the lowest level. Level i is the level above level $i-1$. Let v be a node at level 3 that has some grandchildren. Let v' be a child of v. If v' is a Steiner point, we can assume that the degree of v' is 3, i.e., v' has two children that are terminals. Otherwise, suppose that v' has 3 or more children that are terminals, say, a, b, and c. Assume that a, b, c are clockwise around v'. Then the three angles $\angle av'b$, $\angle bv'c$ and $\angle cv'a$ form 360°. Thus, at least one of the three angles $\angle av'b$, $\angle bv'c$ and $\angle cv'a$ is at most 120°. Without loss of generality, assume that $\angle av'b \leqslant 120°$ and av' is not shorter than bv'. Then $|ab| \leqslant \sqrt{3}\,|av'|$. Let m be the number of degree-2 Steiner points (not including v') on the path from a to v'. Then, we can directly connect a and b with m equally spaced degree-2 Steiner points so that the length of each edge in the segment ab is at most $\sqrt{3} < 1.866$. Thus, the degree of v' is reduced.

In the rest part of the proof, we assume that the degree of v' is 3. We consider two cases.

Case 1: Every edge below v in T has length less than or equal to 1. (See Fig. 1(a).) We consider the case where v has 4 grandchildren. The case where v has 3 grandchildren is easier and is left to the interested readers.

We first consider the case where the degree of v is 3. In this case, we assume that $\angle bv'c > 120°$ and $\angle dfe > 120°$. (Otherwise, we can directly connect b and c, and d and e, respectively. The lengths of edges bc and de are at most $\sqrt{3}$. Thus, the degree of v' becomes 2 and we can continue the modification process with less than n terminals.) $\angle bv'c > 120°$ implies that one of $\angle vv'c$ and $\angle vv'b$ is at most 120°. Similarly, $\angle dfe > 120°$ implies that one of $\angle vfd$ and $\angle vfe$ is at most 120°. Without loss of generality, we assume that $\angle vv'c < 120°$ and $\angle vfd < 120°$. Thus

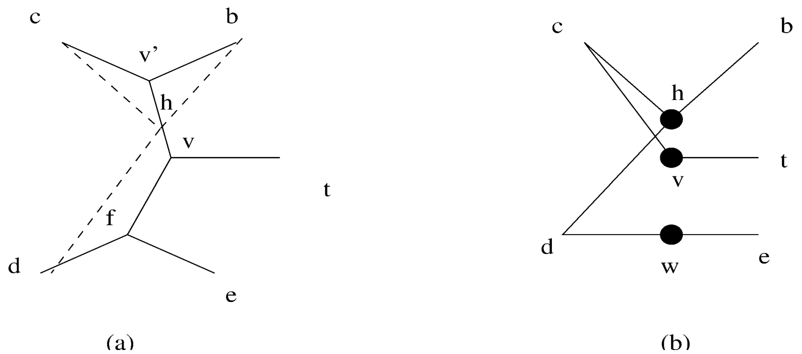

Fig. 1. (a) The original tree T. (b) The modified tree.

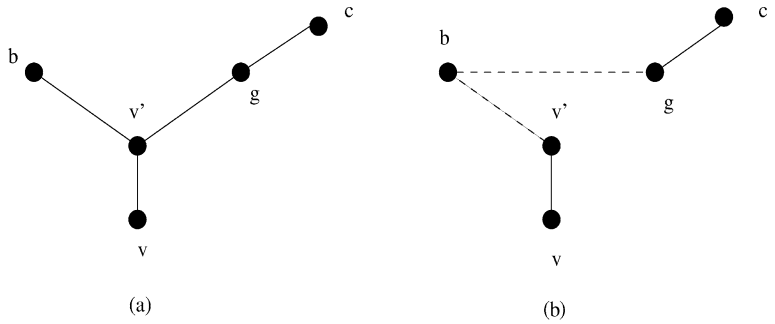

Fig. 2. Case 2.

$|cv| \leqslant \sqrt{3}$ and $|dv| \leqslant \sqrt{3}$. Therefore, we can find a node h on edge $v'v$ such that $|hb| \leqslant 1.866$, $|hc| \leqslant 1.866$ and $|hd| \leqslant 1.866$. (Such a node can be easily found. For example, set $|hv| = (2-\sqrt{3})/2 = 0.134$.) Therefore, we can use two Steiner points h and w to connect the four terminals b, c, d, and e. Moreover, we use the Steiner point v to directly connect terminal c and the node t, where t is the parent of v in T. (See Fig. 1(b).) Note that the length of every new edge ch, hb, hd, vc, dw, and we is at most 1.866, and the length of vt is at most 1. Hence, we can continue the modification process with $n-3$ terminals in $(P \cup \{v\}) - \{b, c, d, e\}$. (Nodes b, c, d e are not included, whereas v is treated as a new terminal.) (See Fig. 1(b).) Note that h connects three terminals b, c and d and they form a full component that spans 3 terminals.

If the degree of v is greater than 3, we can handle it similarly except that more than one full component that spans 3 terminals will be obtained.

Case 2: Some edges have length greater than 1. Let v be a level 3 node that has some grandchildren and v' be v's child. From previous discussion, we can assume that v''s degree is 3. Let b and c be the two children of v'.

Consider the edges $v'b$ and $v'c$ in Fig. 2(a). Without loss of generality, assume that $|v'b| \leqslant |v'c|$, $|v'b| = m$ and m is an integer. $v'c$ can be divided into two segments, $v'g$ and gc, where g is a Steiner point in T such that $v'g$ contains m edges, and cg contains x edges (x degree-2 Steiner points including g). Note that, the length of each edge on $v'g$ and gc is at most 1. Moreover, the total number of degree-2 Steiner points (not including v' and g) in $v'g$ and bv' is $2(m-1)$. Then, we can use $\lceil 1.0713m \rceil - 1$ equally spaced degree-2 Steiner points (i.e., $\lceil 1.0713m \rceil$ edges) to directly connect bg. Conjunction with gc, we connect b and c directly using $\lceil 1.0713m \rceil - 1 + x$ equally spaced degree-2 Steiner points. See Fig. 2(b). From triangle inequality, the length of bg is at most $2m$. Thus, the length of each edge on bg is at most

$$\frac{2m}{\lceil 1.0713m \rceil} \leqslant 1.866. \tag{1}$$

After that, we can use $2(m-1) - (\lceil 1.0713m \rceil - 1) = \lfloor 0.9287m \rfloor - 1$ (remaining number of) equally spaced degree-2 Steiner points to directly connect b and v'. Thus, bv' has $\lfloor 0.9287m \rfloor$ edges. The length of each edge on bv' is at most

$$\frac{m}{\lfloor 0.9287m \rfloor} \leqslant \frac{m}{0.9287m - 1} \leqslant 1.68 \quad \text{if } m \geqslant 3. \tag{2}$$

(1.68 is obtained by substituting m with 3. If $m > 3$, the ratio is smaller than 1.68.) Note that v' becomes a degree-2 node in the new tree. See Fig. 2(b). Thus, we can continue the modification process with $n-1$ terminals in $P \cup \{v'\} - \{b, c\}$, i.e., b and c are not included, whereas v' is treated as a new terminal.

Therefore, when $m \geqslant 3$, the theorem is true. When $m \leqslant 2$, with careful verifications, we can see that the theorem also holds. The cases are illustrated in Fig. 3.

In both (a) and (b), if $|bv'| < |cv'|$ we can directly connect b and c using the same number of equally spaced degree-2 Steiner points in cv'. The length of edges in bc is at most $\frac{5}{3}$ and $\frac{3}{2}$ for (a) and (b), respectively. After that, the degree of v' becomes 2 in the new tree. Thus, we can continue the modification process with $n-1$ terminals $P \cup \{v'\} - \{b, c\}$.

Next, we consider the case $|bv'| = |cv'|$.

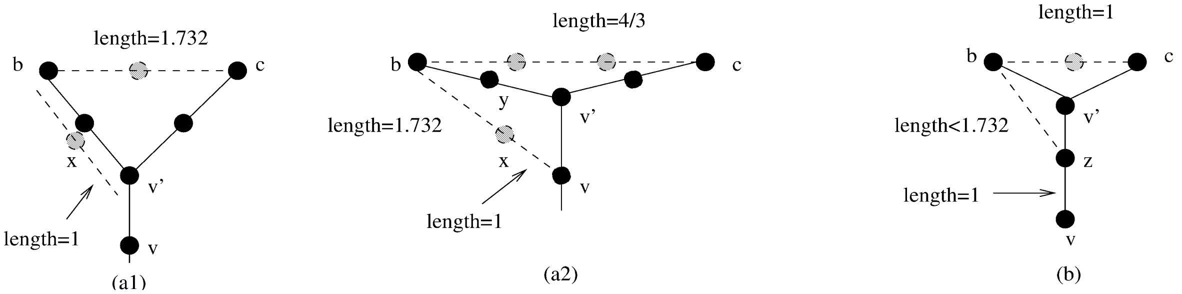

Fig. 3. (a) $m = 2$. (b) $m = 1$.

(a) $m = 2$. There are 3 Steiner points including v' in the path $b - v' - c$ in the original tree.

(a1) $\angle bv'c \leqslant 120°$. Then $|bc| \leqslant 2 \times \sqrt{3}$. Thus, we use one degree-2 Steiner point to connect b and c directly. Therefore, we can continue the process with $n - 1$ terminals, i.e., b and c are not included, whereas x is treated as a new terminal. (See Fig. 3(a1).)

(a2) One of $\angle bv'v$ and $\angle cv'v$ is at most 120°. Assume $\angle bv'v \leqslant 120°$. Thus, $|bv| \leqslant 1 + \sqrt{3}$. We use two degree-2 Steiner points to connect b and c and one Steiner point x to connect b and v. The length of the edge (x, v) is at most 1. Note that, v' is no longer in the reconstructed paths. Thus, we can continue the process with $n - 1$ terminals, i.e., b and c are not included, whereas x is treated as a new terminal. (See Fig. 3(a2).)

(b) $m \leqslant 1$. In this case, $|v'v| > 1$. (The case where $|v'v| \leqslant 1$ has been discussed in Case 1.) So, we can assume that a degree-2 Steiner point z exists in the path $v' - v$. If $\angle bv'c \leqslant 120°$, we can directly connect b and c with an edge of length at most $\sqrt{3}$. The modification process can still continue. Otherwise, one of $\angle bv'v$ and $\angle cv'v$ is at most 120°. Thus, one of bz and cz, say, bz, has length at most 1.732. Thus, we directly connect b and z and use one Steiner points to connect b and c. Note that, in the new tree, v' is not included. (See Fig. 3(b).) In this case, we can continue the process with $n - 1$ terminals, i.e., b and c are not included, whereas z is treated as a new terminal. □

3. The algorithm

Now, we will focus on the approximation of an optimal 3-restricted Steiner tree. We transform the computation of an optimal 3-restricted Steiner tree into the *minimum spanning tree problem for* 3*-hypergraphs*. A *hypergraph* $H = (V, F)$ is a generalization of a graph where the edge set F is an arbitrary family of subsets of vertex set V. A *weighted hypergraph* $H = (V, F, w)$ is a hypergraph such that each of its edge e in F has a weight $w(e)$. An r*-hypergraph* $H_r(V, F, w)$ is a weighted hypergraph, each of its edges has cardinality at most r. We consider $r = 3$ here. A *spanning tree* for a 3-hypergraph $H = (V, F)$ is a subgraph T of H that is a tree containing every node in V. A *minimum spanning tree* for a weighted 3-hypergraph $H = (V, F, w)$ is a spanning tree with the smallest weight.

Now we construct a weighted 3-hypergraph $H_3(V, F, w)$ from the set P of terminals. Here V in $H_3(V, F, w)$ is just the set P of given terminals in the plane.

$$F = \{(a, b) \mid a \in P \text{ and } b \in P\} \cup \\ \{(a, b, c) \mid a \in P \text{ and } b \in P \text{ and } c \in P\}.$$

To obtain the weight of each edge in F, we need to know B, the length of the longest edge in an optimal solution for the bottleneck Steiner tree problem. It is hard to find the exact value of B. However, we can find a B' that is at most $(1 + \varepsilon)B$ for any ε as follows:

(1) run the ratio-2 algorithm in [7] to get an upper bound X of B.
(2) Try to use one of $\frac{X}{2}, \frac{X}{2}(1 + \frac{1}{p}), \frac{X}{2}(1 + \frac{2}{p}), \ldots, X$ as B', where p is an integer such that $\frac{1}{p} \leqslant \varepsilon$.

The weight of an edge in F is defined as follows:

(1) The weight for each edge $(a, b) \in F$ is the smallest number of Steiner points that should be

added to the edge such that the length of each resulting edge is at most B', i.e.,

$$w(f) = \left\lceil \frac{|ab|}{B'} \right\rceil - 1.$$

(2) The weight for each edge $(a, b, c) \in F$ is the smallest number of Steiner points used to connect a, b, and c possibly via a degree-3 Steiner point s such that the length of each edge thus obtained is at most B'.

Computing the weight for $(a, b, c) \in F$ needs some care. Suppose $|ab| \leqslant |bc| \leqslant |ca|$. Then

$$w = \min\left\{ \left\lceil \frac{|ab|}{B'} \right\rceil + \left\lceil \frac{|bc|}{B'} \right\rceil - 2, k \right\}$$

is an upper bound of the weight, where k is the given number of Steiner points in an optimal solution. We do not have to consider the edges where

$$\left\lceil \frac{|ab|}{B'} \right\rceil + \left\lceil \frac{|bc|}{B'} \right\rceil - 2 > k.$$

Let i, j and l be the numbers of degree-2 Steiner points in the segments as, bs and cs such that $f = i + j + l + 1$ is minimized. To compute f, we only have to guess the values of i, j and l. This can be done in $O(w^3) \leqslant O(k^3)$ time by trying all possibilities. For each guessed value i, j and l, we have to test if the three circles centered at a, b and c with radius $i \times B'$, $j \times B'$ and $l \times B'$ have a point in common. This can be done in constant time.

Lemma 2. *Testing whether three circles have a point in common can be done in constant time.*

Proof. Let $\odot a$, $\odot b$ and $\odot c$ be the three circles with a, b and c as centers and $i \times B'$, $j \times B'$ and $l \times B'$ as radii.

It is easy to test whether $\odot a$ and $\odot c$ are disjoint or one completely contains the other. In both cases, it is trivial to test if the three circles have a point in common. Now, we only consider the case where $\odot a$ and $\odot c$ have two intersection points, say, x and y, on the boundary. The following cases arise:

(b1) $\odot b$ has no common point on the boundary of $\odot a \cap \odot c$. In this case, if the three circles have some common point(s), we have to consider the two subcases. That is,
(b1.1) $\odot b$ is in $\odot a \cap \odot c$; and
(b1.2) $\odot a \cap \odot c$ is in $\odot b$.

See Figs. 4(a) and 4(b). In this case, to test if the three circles have a point in common, we only have to test (1) if point b is in both $\odot a$ and $\odot c$ for (b1.1); and (2) if x or y is in $\odot b$ for (b1.2).

(b2) $\odot b$ has some common point(s) on the boundary of $\odot a \cap \odot c$. In this case, we have to consider up to six intersection points on the boundaries of $\odot a \cap \odot b$, $\odot a \cap \odot c$ and $\odot b \cap \odot c$. (See Fig. 4(c).) It is easy to test whether one of the six points is in all three circles. This completes the proof. □

After obtaining $H_3(V, F, w)$, we can use the algorithm in [5] to find the minimum spanning tree for $H_3(V, F, w)$.

Theorem 3 [5]. *There exists a randomized algorithm for the minimum spanning tree problem in* 3*-hypergraphs running in poly*$(n, w_{\max})$ *time with probability at least* 0.5, *where n is the number of nodes in*

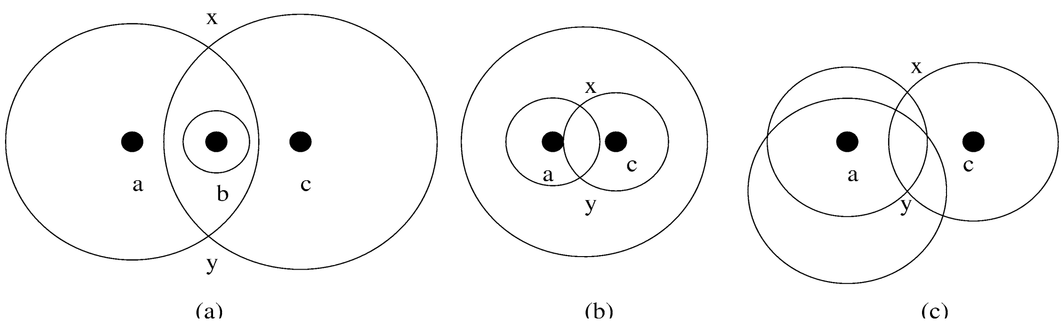

Fig. 4. (a) $\odot b$ is in $\odot a \cap \odot c$. (b) $\odot a \cap \odot c$ is in $\odot b$. (c) $\odot b$ has some common point(s) on the boundary of $\odot a \cap \odot c$.

Algorithm Bottleneck Steiner Tree

Input A set P of n terminals in the Euclidean plane, an integer k and a positive number ε

Output A 3-restricted Steiner tree T with at most k Steiner points.

1. Call the ratio-2 approximation algorithm for bottleneck Steiner tree problem in [7] and obtain a number X as the length of the longest edge.
2. **for** $B = \frac{X}{2}, \frac{X}{2}(1+\varepsilon), \frac{X}{2}(1+2\varepsilon), \ldots, \frac{X}{2}(1+\varepsilon \times \lceil \frac{1}{\varepsilon} \rceil)$ **do**
3. Construct a weighted hypergraph $H_3(V, F, w)$.
4. Call the randomized algorithm in [5] to compute a minimum spanning tree T for $H_3(V, F, w)$;
5. Consider the solution T' of the smallest B such that $w(T') \leqslant k$.
6. Replace every edge f of the minimum spanning tree T' on $H_3(V, F, w)$ with a Steiner tree with $w(f)$ Steiner points such that the maximum length of each edge in the tree is at most B and output the obtained tree.

Fig. 5. Algorithm bottleneck problem.

the hypergraph and $w_{\max}$ *is the largest weight of edges in the hypergraph.*

Note that, there are $O(n^3)$ edges in the hypergraph, where n is the number of given terminals in the plane. To compute the weight for an edge (a triple) in the hypergraph, we need $O(k^3)$ time to guess the radii of the three circles, where k is the given number of Steiner points in an optimal solution. Thus, we can construct a weighted 3-hypergraph in $O(n^3k^3)$ time. Therefore, we have

Theorem 4. *For any given* ε, *there exists a randomized algorithm that computes a Steiner tree with* n *terminals and* k *Steiner points such that the longest edge in the tree is at most* $1.866 + \varepsilon$ *times of the optimum running in* $\frac{1}{\varepsilon} \times \mathit{poly}(n, k)$ *time with probability at least* 0.5.

The complete algorithm is given in Fig. 5. In fact, one can speed up Step 2 by using binary search.

Acknowledgement

We thank the referees for their useful suggestions. This work is fully supported by HK RGC Grant 9040351.

References

[1] D. Chen, D.-Z. Du, X. Hu, G. Lin, L. Wang, G. Xue, Approximations for Steiner trees with minimum number of Steiner points, J. Global Optimization 18 (2000) 17–33.

[2] C. Chiang, M. Sarrafzadeh, C.K. Wong, A powerful global router: based on Steiner min-max tree, IEEE Trans. Computer-Aided Design 19 (1990) 1318–1325.

[3] G. Lin, G. Xue, Steiner tree problem with minimum number of Steiner points and bounded edge-length, Inform. Process. Lett. 69 (1999) 53–57.

[4] I.I. Mandoiu, A.Z. Zelikovsky, A note on the MST heuristic for bounded edge-length Steiner trees with minimum number of Steiner points, Inform. Process. Lett. 75 (2000) 165–167.

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

[6] M. Sarrafzadeh, C.K. Wong, Bottleneck Steiner trees in the plane, IEEE Trans. Comput. 41 (1992) 370–374.

[7] L. Wang, D.-Z. Du, Approximations for a bottleneck Steiner tree problem, Algorithmica, to appear.