On the Restricted 1-Steiner Tree Problem*

Prosenjit Bose[1][0000−0002−1825−0097], Anthony D'Angelo[1] (✉), and Stephane Durocher[2]

[1] Carleton University, Ottawa ON K1S-5B6, Canada
jit@scs.carleton.ca,anthony.dangelo@carleton.ca
http://www.scs.carleton.ca/~jit
[2] University of Manitoba, Winnipeg MB R3T-2N2, Canada
durocher@cs.umanitoba.ca

Abstract. Given a set P of n points in $\mathbb{R}^2$ and an input line γ, we present an algorithm that runs in optimal $\Theta(n \log n)$ time and $\Theta(n)$ space to solve a restricted version of the 1-Steiner tree problem. Our algorithm returns a minimum-weight tree interconnecting P using at most one Steiner point $s \in \gamma$ where edges are weighted by the Euclidean distance between their endpoints.

Keywords: Minimum k-Steiner tree · Steiner point restrictions

1 Introduction

Finding the shortest interconnecting network for a given set of points is an interesting problem for anyone concerned with conserving resources. Sometimes, we are able to add new points in addition to the given input points to reduce the total length of the edges in the interconnecting network. These extra points are called Steiner points. However, finding where to place these Steiner points and how many to place is NP-hard [11, 12, 23, 33], and so a natural question is: *What is the shortest spanning network that can be constructed by adding only k Steiner points to the given set of points?* This is the k-Steiner point problem.

Consider a set P of n points in the 2-D Euclidean plane, which are also called *terminals* in the Steiner tree literature. The **Minimum Spanning Tree** (**MST**) problem is to find the minimum-weight tree interconnecting P where edges are weighted by the Euclidean distance between their endpoints. Let $\mathrm{MST}(P)$ be a Euclidean minimum spanning tree on P and let $|\mathrm{MST}(P)|$ be the sum of its edge-weights (also called the *length* of the tree). Imagine we are given another set S of points in the 2-D Euclidean plane. The set S is the set of *Steiner points* that we may use as intermediate nodes in addition to the points of P to compute the minimum-weight interconnection of P. An MST on the union of the terminals P with some subset of Steiner points $S' \subseteq S$, i.e., $\{P \cup S'\}$, is a Steiner tree. In the Euclidean **Minimum Steiner Tree** (**MStT**) problem, the goal is to find

* Funded in part by the Natural Sciences and Engineering Research Council of Canada (NSERC).

a subset $S' \subseteq S$ such that $|\,\mathrm{MST}(\{P \cup S'\})|$ is no longer than $|\,\mathrm{MST}(\{P \cup X\})|$ for any $X \subseteq S$. Such a minimum-weight tree is a MStT. For our restricted k-Steiner tree problem, we are given an input line γ in $\mathbb{R}^2$; the line $\gamma = S$ and the cardinality of S' is at most k.

As 3-D printing enters the mainstream, material-saving and time-saving printing algorithms are becoming more relevant. Drawing on the study of MStTs, Vanek et al. [36] presented a geometric heuristic to create support-trees for 3-D printed objects where the forking points in these trees are solutions to a constrained Steiner point problem. Inspired by the work of Vanek et al. as well as the solutions for the 1-Steiner and k-Steiner point problems in the 2-D Euclidean plane [8, 12, 24], we present an efficient algorithm to compute an exact solution for the 1-Steiner point problem where the placement of the Steiner point is constrained to lie on an input line. We present another motivating example. Imagine we have a set V of wireless nodes that must communicate by radio transmission. To transmit a longer distance to reach more distant nodes requires transmitting at a higher power. The MST of V can be used to model a connected network that spans the nodes of V while minimizing total power consumption. Suppose that an additional wireless node is available to be added to V, but that the new node's position is restricted to lie on a road γ on which it will be delivered on a vehicle. Where on γ should the additional node be positioned to minimize the total transmission power of the new network?

We refer to our problem as a 1-*Steiner tree problem restricted to a line*. For our purposes, let an ***optimal Steiner point*** be a point $s \in \gamma$ such that $|\,\mathrm{MST}(P \cup \{s\})| \leq |\,\mathrm{MST}(P \cup \{u\})|$ for all $u \in \gamma$.

Problem Given a set of n points P in $\mathbb{R}^2$ and a line γ in $\mathbb{R}^2$, compute the MStT of P using at most 1 optimal Steiner point $s \in \gamma$.

A restricted version of our problem has been studied for the case when the input point set P lies to one side of the given input line and a point from the line *must* be chosen. Chen and Zhang gave an $O(n^2)$-time algorithm to solve this problem [15]. Similar problems have also been studied by Li et al. [29] building on the research of Holby [28]. The two settings they study are: (a) the points of P lie anywhere and *must* connect to the input line using any number of Steiner points, and any part of the input line used in a spanning tree does not count towards its length; and (b) the same problem, but the optimal line to minimize the network length is not given and must be computed. Li et al. provide 1.214-approximation[3] algorithms for both (a) and (b) in $O(n \log n)$ and $O(n^3 \log n)$ time respectively. The problems of Chen and Zhang, Li et al., and Holby are different than our problem since we are not required to connect to our input line, we have no restriction on the placement of the points of P with respect to the line, and travel in our network has the same cost ***on*** the input line as ***off*** of it. For example, one can imagine if the points of the point set were close

[3] This means the length of their tree is at most 1.214 times the length of the optimal solution. Here they take advantage of the result of Chung and Graham [18] showing that the MST is a 1.214-approximation (to three decimals) of the MStT.

to the line but far from each other, in which case the solution of Li et al. [29] would connect the points to the line and get a tree with much less weight/length than even the MStT. Such an example is shown in Fig. 1. In Fig. 1 the MStT of points $\{a, b, c, d\}$ is the same as its MST since all triples form angles larger than $\frac{2\pi}{3}$ [12, 25]. In our setting, the MST is the best solution for this point set, whereas in the setting of Holby [28] and Li et al. [29], the best solution connects each input point directly to γ to form a spanning tree between the points using pieces of γ. The length of the MST is significantly larger than the length of the other solution since in their setting, only the edges connecting the points to γ contribute to the length of the spanning tree.

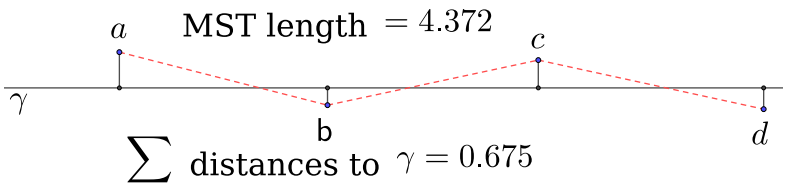


Fig. 1. Here we have γ as the x-axis, $a = (0.489, 0.237)$, $b = (1.865, -0.114)$, $c = (3.26, 0.184)$, and $d = (4.75, -0.141)$. The MST of $\{a, b, c, d\}$ in red dashed line segments and its length, the input line γ, a spanning tree of $\{a, b, c, d\}$ connecting each point to γ, and the length of this spanning tree for the setting of Holby [28] and Li et al. [29].

We use a type of Voronoi diagram in our algorithm whose regions are bounded by rays and segments. We make a general position assumption that γ is not collinear with any ray or segment in the Voronoi diagrams. In other words, the intersection of the rays and segments of these Voronoi diagrams with γ is either empty or a single point. We also assume that the edges of MST(P) have distinct weights. In this paper we show the following.

Theorem 1. *Given a set P of n points in the Euclidean plane and a line γ, there is an algorithm that computes in optimal $\Theta(n \log n)$ time and optimal $\Theta(n)$ space a minimum-weight tree connecting all points in P using at most one point of γ.*

Section 2 reviews the tools and properties we will need for our algorithm, and Section 3 presents our algorithm and the proof of Theorem 1.

2 Relevant Results

There has been a lot of research on Steiner trees in various dimensions, metrics, norms, and under various constraints. See the surveys by Brazil et al. [7] and Brazil and Zachariasen [12] for a good introduction. In the general Euclidean case it has been shown that Steiner points that reduce the length of the MST

have degree 3 or 4 [32]. There are results for building Steiner trees when the *terminal set* is restricted to *zig-zags* [4,20], *curves* [33], *ladders* [19], and *checkerboards* [6, 9, 10]; for when the angles between edges are constrained [11, 12]; for obstacle-avoiding Steiner trees [37–41] (which include geodesic versions where the terminals, Steiner points, and tree are contained in polygons); and for k-Steiner trees with k as a fixed constant where you can use at most k Steiner points (for terminals and Steiner points in various normed planes including the 2-D Euclidean plane, there is an $O(n^{2k})$-time algorithm) [8, 12, 24].

2.1 Tools

Without loss of generality, we consider the positive x-axis to be the basis for measuring angles, so that 0 radians is the positive x-axis, $\frac{\pi}{3}$ radians is a counterclockwise rotation of the positive x-axis about the origin by $\frac{\pi}{3}$ radians, etc.

Observation 2. *Given a point set $V \subset \mathbb{R}^2$, if we build* MST(V)*, each point $v \in V$ will have at most* 6 *neighbours in the MST. This is because, due to the sine law, for any two neighbours w and z of v in* MST(V) *the angle $\angle wvz$ must be at least $\frac{\pi}{3}$ radians. These potential neighbours can be found by dividing the plane up into* 6 *interior-disjoint cones of angle $\frac{\pi}{3}$ all apexed on v. The closest point of V to v in each cone is the potential neighbour of v in the MST in that cone.*

Consider our input line γ as being the real number line, represented by the x-axis in the Euclidean plane. This line can be parametrized by x-coordinates. Let an ***interval*** on γ be the set of points on γ in between and including two fixed x-coordinates, called the endpoints of the interval. Our approach will be to divide the input line into $O(n)$ intervals using a special kind of Voronoi diagram outlined below. The intervals have the property that for any given interval I, if we compute MST($P \cup \{s\}$) for any $s \in I$, the subset of possible neighbours of s in the MST is constant. For example, Fig. 2 shows a set V of input points with the blue points labelled p_i for $1 \leq i \leq 6$, the input line γ, and a green interval I. The plane is divided into 6 cones of 60 degrees, all apexed on the red point $x \in I$. In MST($V \cup \{x\}$), if x connects to a point in cone i, it connects to p_i. The green interval I has the property that this is true anywhere we slide x and its cones in I.

Oriented Voronoi Diagrams The 1-Steiner point algorithm of Georgakopoulos and Papadimitriou (we refer to this algorithm as *GPA*) [24] works by subdividing the plane into $O(n^2)$ regions defined by the cells of the *Overlaid Oriented Voronoi Diagram* (overlaid **OVD**).[4] Refer to the cone $\mathbb{K}$ defining an OVD as an *OVD-cone*. Let $\mathbb{K}_v$ be a copy of the OVD-cone whose apex coincides with point $v \in \mathbb{R}^2$. OVDs are a type of Voronoi diagram made up of oriented Voronoi

[4] In the Georgakopoulos and Papadimitriou paper [24] this is referred to as *Overlaid Oriented Dirichlet Cells*.

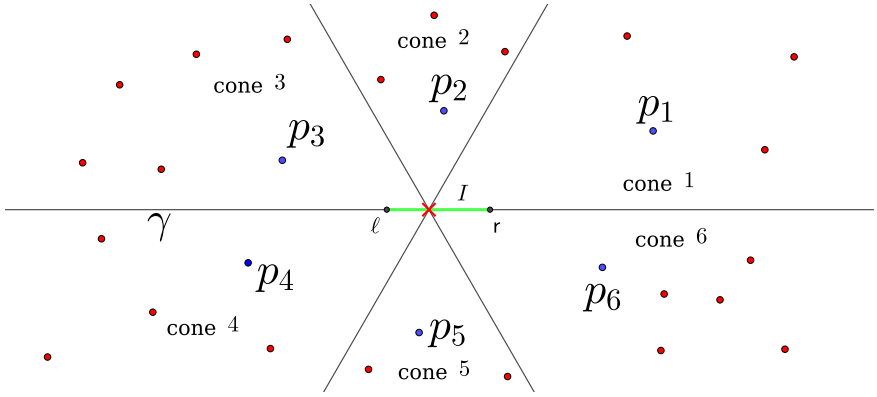


Fig. 2. Every point along the green interval I of γ (i.e., between the ℓ endpoint and the r endpoint) has the same potential MST neighbour (the blue points) in the same cone.

regions (**OVRs**) where the OVR of a site $p \in P$ is the set of points $w \in \mathbb{R}^2$ for which p is the closest site in $\mathbb{K}_w \cap P$. If $\mathbb{K}_w \cap P = \emptyset$ we say w belongs to an OVR whose site is the empty set. These notions are illustrated in Fig. 3.

Chang et al. [13] show us that the OVD for a given OVD-cone of angle $\frac{\pi}{3}$ (e.g., the OVD in Fig. 3) can be built in $O(n \log n)$ time using $O(n)$ space. The OVD is comprised of segments and rays that are subsets of bisectors and cone boundaries which bound the OVRs. The size of the OVD is $O(n)$.

Since by Observation 2 a vertex of the MST has a maximum degree of 6, by overlaying the 6 OVDs for the 6 cones of angle $\frac{\pi}{3}$ that subdivide the Euclidean plane (i.e., each of the six cones defines an orientation for a different OVD) the GPA creates $O(n^2)$ regions. Each of these regions has the property that if we place a Steiner point s in the region, the points of P associated with this region (up to 6 possible points) are the only possible neighbours of s in the MStT (similar to the example in Fig. 2). The GPA then iterates over each of these regions. In region R, the GPA considers each subset of possible neighbours associated with R. For each such subset it then computes the optimal location for a Steiner point whose neighbours are the elements of the subset, and then computes the length of the MStT using that Steiner point, keeping track of the best solution seen. The generalized algorithm for placing k Steiner points [8, 12] essentially does the same thing k times (by checking the topologies of the MStT for all possible placements of k points), but is more complicated (checking the effects that multiple Steiner points have on the MStT is more complex).

Updating Minimum Spanning Trees In order to avoid actually computing each of the candidate MSTs on the set of P with the addition of our candidate Steiner points, we instead compute the *differences in length* between MST(P) and the candidate MStTs. Georgakopoulos and Papadimitriou similarly avoid repeated MST computations by performing $O(n^2)$ preprocessing to allow them

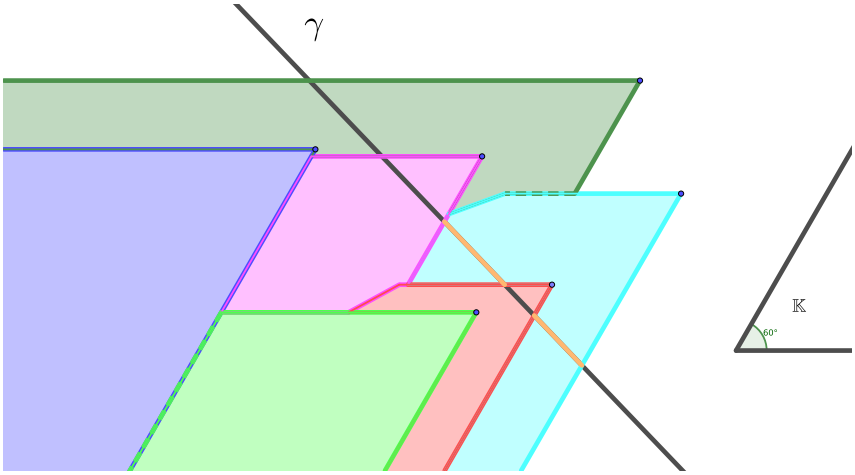


Fig. 3. An example of an OVD for 6 points defined by the OVD-cone $\mathbb{K}$ with bounding rays oriented towards 0 and $\frac{\pi}{3}$. The 6 sites (i.e., the points) are the blue top-right points of the coloured OVRs. When intersected with γ, the OVD creates intervals along γ. Each interval corresponds to exactly one OVR, but an OVR may create multiple intervals (for example, the light-blue OVR creates the two orange intervals). The site corresponding to an interval outside of a coloured OVR is a special site represented by the empty set.

to answer queries of the following type in constant time: given that the edges $ab_1, ab_2, \ldots, ab_j$ are decreased by $\delta_1, \delta_2, \ldots, \delta_j$ for constant j, what is the new MST? They then use these queries to find the length of the MStT for each candidate Steiner point. Refer to [24] for details. Brazil et al. also perform some preprocessing in time between $O(n^2)$ and $O(n^3)$ [8]. However, using an approach involving an auxiliary tree and lowest common ancestor (LCA) queries, we can compute what we need in $o(n^2)$ time. We first compute MST(P) and build an auxiliary tree in $O(n \log n)$ time and process the auxiliary tree in $O(n)$ time [27] to support LCA queries in $O(1)$ time [5, 30].

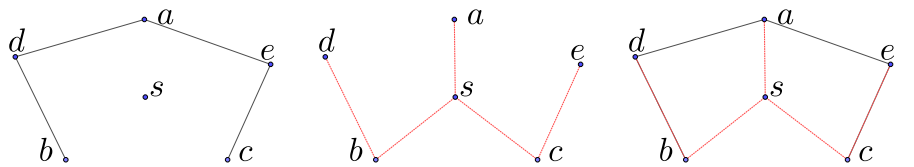
3 Algorithm

In this section we present our algorithm and prove Theorem 1. The algorithm computes OVDs for the 6 cones of angle $\frac{\pi}{3}$ that divide up the Euclidean plane (i.e., each of the 6 cones defines an orientation for a different OVD). Though they can be overlaid in $O(n^2)$ time, we do not need to overlay them. As mentioned in Section 2.1, each OVD has $O(n)$ size and is therefore comprised of $O(n)$ rays and segments. As illustrated in Fig. 3, intersecting any given OVD with a line γ carves γ up into $O(n)$ intervals since we have $O(n)$ rays and segments, each of which intersect a line $O(1)$ times.[5] Each interval corresponds to an intersection

[5] This follows from the zone theorem [3, 14, 21].

of γ with exactly one OVR of the OVD since OVDs are planar, but multiple non-adjacent intervals may be defined by the same OVR, as in Fig. 3. Therefore each interval I is a subset of an OVR, and for every pair of points $u_1, u_2 \in I$ the closest point in $\mathbb{K}_{u_1} \cap P$ is the same as in $\mathbb{K}_{u_2} \cap P$, where $\mathbb{K}$ is the OVD-cone of the OVD being considered.

If we do this with all six OVDs, γ is subdivided into $O(n)$ intervals. As in Fig. 2, each interval I has the property that for any point $u \in I$, if we were to build MST($P \cup \{u\}$), the ordered set of six potential neighbours is a constant-sized set.[6] Each element of this ordered set is defined by a different OVD and corresponds to the closest point in $\mathbb{K}_u \cap P$. In each interval we solve an optimization problem to find the optimal placement for a Steiner point in that interval (i.e., minimize the sum of distances of potential neighbours to the Steiner point) which takes $O(1)$ time since each of these $O(1)$ subproblems has $O(1)$ size.



(a) MST($\{a, b, c, d, e\}$) (b) MST($\{a, b, c, d, e, s\}$) (c) Union of the trees from Figs. 4a and 4b

Fig. 4. The union of the trees in Figs. 4a and 4b give the graph in Fig. 4c with cycles (s, b, d, a), (s, a, e, c), and (s, b, d, a, e, c) whose longest edges excluding s are (d, a) and (a, e).

Once we have computed an optimal placement for a Steiner point for each computed interval of our input line γ, we want to compute which one of these $O(n)$ candidates produces the MStT, i.e., the candidate s that produces the smallest length of the MST($P \cup \{s\}$). Let T^* be the union of MST(P) and MST($P \cup \{s\}$), as in Fig. 4. For a candidate s, the savings are calculated by summing the length of the longest edge on each cycle of T^* excluding the edges incident to s minus the sum of the lengths of the edges incident to s in MST($P \cup \{s\}$). For example, in Fig. 4c, the candidate edges on the left cycle are (b, d) and (d, a), and on the right cycle they are (a, e) and (e, c); we sum the lengths of the longest candidate edge from each cycle, i.e., (d, a) and (a, e), and subtract the sum of the lengths of edges (s, a), (s, b), and (s, c) to calculate the savings we get from choosing s as the solution Steiner point. Note that the longest edge on the cycle (s, b, d, a, e, c) is either (d, a) or (e, c). As will be seen in the proof of Theorem 1, the sum of the lengths of the edges incident to s are computed

[6] In other words, each $u \in I$ has the same constant-sized set of fixed candidate topologies that could be the result of MST($P \cup \{u\}$).

when determining s. What remains to find are the lengths of the longest edges of MST(P) on the cycles of T^*. The following theorem from Bose et al. [5] tells us that with $O(n \log n)$ preprocessing of MST(P), we can compute the sum in which we are interested in $O(1)$ time for each candidate Steiner point.[7] First an auxiliary binary tree is computed whose nodes correspond to edge lengths and leaves correspond to points of P. This tree has the property that the LCA of two leaves is the longest edge on the path between them in MST(P). They then take advantage of a result that uses $O(n)$ preprocessing on the auxiliary tree enabling them to perform $O(1)$-time LCA queries (either Harel and Tarjan [27], Schieber and Vishkin [34], or Bender and Farach-Colton [2]).

Lemma 1 (Bose et al. 2004 **[5, paraphrased Theorem 2]).** *We can preprocess a set of n points in $\mathbb{R}^2$ in $O(n \log n)$ time into a data structure of size $O(n)$ such that the longest edge on the path between any two points in the MST can be computed in $O(1)$ time.*

We are now ready to finish proving Theorem 1.

Theorem 1. *Given a set P of n points in the Euclidean plane and a line γ, there is an algorithm that computes in optimal $\Theta(n \log n)$ time and optimal $\Theta(n)$ space a minimum-weight tree connecting all points in P using at most one point of γ.*

Proof. The tree $T = \text{MST}(P)$ and its length are computed in $O(n \log n)$ time and $O(n)$ space by computing the Voronoi diagram in those bounds [1,3,22,26], walking over the Voronoi diagram creating the dual and weighting the edges in $O(n)$ time and space (the reasoning for which follows from Shamos [35]), and computing the MST from the Delaunay triangulation in $O(n)$ time and space [16, 31]. By Lemma 1, in $O(n \log n)$ time and $O(n)$ space we compute the longest edge auxiliary tree T' and preprocess it to answer LCA queries in $O(1)$ time. Each of the 6 OVDs is then computed in $O(n \log n)$ time and $O(n)$ space [8, 13, 17]. In $O(n)$ time and space we extract $\mathbb{L}$, the set of rays and segments defining each OVR of each OVD. While computing the OVDs, in $O(n)$ time we add labels to the boundary rays and segments describing which OVD-cone defined them and the two sites corresponding to the two OVRs they border.

Since γ is a line, it intersects any element of $\mathbb{L}$ $O(1)$ times and we can compute each of these intersections in $O(1)$ time. Therefore, computing the intersections of γ with $\mathbb{L}$ takes $O(n)$ time and space. Assume without loss of generality that γ is the x-axis. Given our $O(n)$ intersection points, we can make a list of the $O(n)$ intervals they create along γ in $O(n \log n)$ time and $O(n)$ space by sorting the intersection points by x-coordinate and then walking along γ. During this process we also use the labels of the elements of $\mathbb{L}$ to label each interval with its six potential neighbours described above in $O(1)$ time per interval.

By the triangle inequality, an optimal Steiner point has degree more than 2. In [32] it was shown to have degree no more than 4. Therefore an optimal Steiner

[7] A similar result was shown in Monma and Suri [30, Lemma 4.1, pg. 277].

point has degree 3 or 4. We then loop over each interval looking for the solution by finding the optimal placement of a Steiner point in the interval for $O(1)$ fixed topologies. Consider an interval I and its set of potential neighbours $P' \subset P$ of size at most 6. For each subset $\mathbb{P}$ of P' of size 3 and 4 (of which there are $O(1)$), we compute $O(1)$ candidate optimal Steiner points in γ. Note that γ is actually a polynomial function, $\gamma(x)$. Our computation is done using the following distance function $d(x)$, where a_x and a_y are the x and y coordinates of point a respectively, and $\gamma(x)$ is the evaluation of γ at x: $d_{\mathbb{P}}(x) = \sum_{a \in \mathbb{P}}^{|\mathbb{P}|} \sqrt{(a_x - x)^2 + (a_y - \gamma(x))^2}$. We then take the derivative of this distance function and solve for the global minima by finding the roots within the domain specified by the endpoints of I. Since the size of $\mathbb{P}$ is bounded by a constant and since the degree of the polynomial γ is a constant, this computation takes $O(1)$ time and $O(1)$ space and the number of global minima is $O(1)$. Note that the value of the distance function at a particular x for a particular $\mathbb{P}$ tells us the sum of edge lengths from the point $u = (x, \gamma(x))$ to the points in $\mathbb{P}$. We associate this value with u. Out of the $O(1)$ candidate points, we choose the one for which $d_{\mathbb{P}}(x)$ is minimum. We can break ties arbitrarily, since a tie means the points offer the same amount of savings to the MST since they both have the same topology in the MST (meaning they have the same cycles in $\mathrm{MST}(P) \cup \mathrm{MST}(P \cup \{u\})$), and since the value of $d_{\mathbb{P}}(x)$ being the same means that the sum of adjacent edges is the same.

Once we have our $O(1)$ candidate optimal Steiner points for I, we need to compare each one against our current best solution s. In other words, for each candidate u we need to compare $|\mathrm{MST}(P \cup \{u\})|$ with $|\mathrm{MST}(P \cup \{s\})|$. We take advantage of the following: if we compute the union of $\mathrm{MST}(P)$ and $\mathrm{MST}(P \cup \{u\})$ we get at most $\binom{4}{2} = 6$ simple cycles[8] through u. Let this connected set of cycles be Q. If $\mathbb{P}_u$ is the set of neighbours associated with the candidate u, we have $|\mathrm{MST}(P \cup \{u\})| = |\mathrm{MST}(P)| + d_{\mathbb{P}_u}(u) - \Delta$, where Δ is the sum of the longest edge in each cycle of Q excluding from consideration the edges incident to u. By Lemma 1, we can compute Δ in $O(1)$ time using T'. Due to space constraints, we omit the proof that removing the longest edge from each cycle of Q results in a tree. If $|\mathrm{MST}(P \cup \{u\})| < |\mathrm{MST}(P \cup \{s\})|$ we set $s = u$.

Finally, we check if $|\mathrm{MST}(P \cup \{s\})| < |T|$. If so, we return $\mathrm{MST}(P \cup \{s\})$. Otherwise we return T.

Now we show the space and time optimality. The $\Omega(n)$-space lower-bound comes from the fact that we have to read in the input. The $\Omega(n \log n)$-time lower-bound comes from a reduction from the *closest pair* problem (CPP). The CPP is where we are given n points in $\mathbb{R}^2$ and we are supposed to return a closest pair with respect to Euclidean distance. The CPP has an $\Omega(n \log n)$ lower-bound [31, Theorem 5.2]. Indeed, given an instance of CPP, we can turn it into our problem in $O(n)$ time by using the points as the input points P and choosing an arbitrary γ.

Given the solution to our problem, we can find a closest pair in $O(n)$ time by walking over the resulting tree. First, remove the Steiner point (if any) and its incident edges to break our tree up into $O(1)$ connected components. Consider

[8] In a simple cycle the only vertex seen twice is the first/last vertex.

one of these components $\mathbb{C}$. $\mathbb{C}$ may contain both points of multiple closest pairs, or none. Imagine $\mathbb{C}$ contained both points for exactly one closest pair. Then the edge connecting them will be in $\mathbb{C}$ and it will be the edge with minimum-weight in $\mathbb{C}$; otherwise it contradicts that we had a minimum-weight tree. Imagine $\mathbb{C}$ contained both points for multiple closest pairs. Pick one of the closest pairs. If $\mathbb{C}$ does not contain the edge e connecting the two points of the pair, then there is a path between them in $\mathbb{C}$ consisting of minimum-weight edges (whose weights match e) connecting other closest pairs; otherwise we contradict the minimality of our tree or that both points were in the same connected component. If no component contains both points of a closest pair, then the path between a closest pair goes through the Steiner point. Once again, choose a closest pair (a, b) and let the edge connecting this closest pair be e. Due to the minimality of our tree, the weight of every edge on the path between a and b is no more than that of e. However, since no component contains a closest pair, that means that a and b are incident to the Steiner point. Therefore, we get a solution to the CPP by walking over our resulting tree and returning the minimum among a minimum-weight edge connecting neighbours of the Steiner point and a minimum-weight edge seen walking through our tree excluding edges incident to the Steiner point. □

Corollary 1. *Given a set P of n points in the Euclidean plane and j lines $\Gamma = \{\gamma_1, \ldots \gamma_j\}$, by running the algorithm of Theorem 1 for each $\gamma \in \Gamma$, in $O(jn \log n)$ time and $O(n+j)$ space we compute a minimum-weight tree connecting all points in P using at most one point from $\gamma_1, \ldots \gamma_j$.*

Acknowledgements The authors thank Jean-Lou De Carufel for helpful discussions.

References

1. Aurenhammer, F., Klein, R., Lee, D.: Voronoi Diagrams and Delaunay Triangulations. World Scientific (2013)
2. Bender, M.A., Farach-Colton, M.: The LCA problem revisited. In: LATIN. Lecture Notes in Computer Science, vol. 1776, pp. 88–94. Springer (2000)
3. de Berg, M., Cheong, O., van Kreveld, M.J., Overmars, M.H.: Computational geometry: algorithms and applications, 3rd Edition. Springer (2008)
4. Booth, R.S., Weng, J.F.: Steiner minimal trees for a class of zigzag lines. Algorithmica **7**(2&3), 231–246 (1992)
5. Bose, P., Maheshwari, A., Narasimhan, G., Smid, M.H.M., Zeh, N.: Approximating geometric bottleneck shortest paths. Comput. Geom. **29**(3), 233–249 (2004)
6. Brazil, M., Cole, T., Rubinstein, J.H., Thomas, D.A., Weng, J.F., Wormald, N.C.: Minimal steiner trees for $2^k \times 2^k$ square lattices. J. Comb. Theory, Ser. A **73**(1), 91–110 (1996)
7. Brazil, M., Graham, R.L., Thomas, D.A., Zachariasen, M.: On the history of the euclidean steiner tree problem. Archive for history of exact sciences **68**(3), 327–354 (2014)

8. Brazil, M., Ras, C.J., Swanepoel, K.J., Thomas, D.A.: Generalised k-steiner tree problems in normed planes. Algorithmica **71**(1), 66–86 (2015)
9. Brazil, M., Rubinstein, J.H., Thomas, D.A., Weng, J.F., Wormald, N.C.: Full minimal steiner trees on lattice sets. J. Comb. Theory, Ser. A **78**(1), 51–91 (1997)
10. Brazil, M., Rubinstein, J.H., Thomas, D.A., Weng, J.F., Wormald, N.C.: Minimal steiner trees for rectangular arrays of lattice points. J. Comb. Theory, Ser. A **79**(2), 181–208 (1997)
11. Brazil, M., Thomas, D.A., Weng, J.F.: On the complexity of the steiner problem. J. Comb. Optim. **4**(2), 187–195 (2000)
12. Brazil, M., Zachariasen, M.: Optimal interconnection trees in the plane: theory, algorithms and applications, vol. 29. Springer (2015)
13. Chang, M., Huang, N., Tang, C.Y.: An optimal algorithm for constructing oriented voronoi diagrams and geographic neighborhood graphs. Inf. Process. Lett. **35**(5), 255–260 (1990)
14. Chazelle, B., Guibas, L.J., Lee, D.T.: The power of geometric duality. BIT Comput. Sci. Sect. **25**(1), 76–90 (1985)
15. Chen, G., Zhang, G.: A constrained minimum spanning tree problem. Comput. Oper. Res. **27**(9), 867–875 (2000)
16. Cheriton, D.R., Tarjan, R.E.: Finding minimum spanning trees. SIAM J. Comput. **5**(4), 724–742 (1976)
17. Chew, L.P., III, R.L.S.D.: Voronoi diagrams based on convex distance functions. In: Symposium on Computational Geometry. pp. 235–244. ACM (1985)
18. Chung, F.R.K., Graham, R.L.: A new bound for euclidean steiner minimal trees. Annals of the New York Academy of Sciences **440**(1), 328–346 (1985)
19. Chung, F., Graham, R.: Steiner trees for ladders. Annals of Discrete Mathematics **2**, 173–200 (1978)
20. Du, D., Hwang, F., Weng, J.: Steiner minimal trees on zig-zag lines. Transactions of the American Mathematical Society **278**(1), 149–156 (1983)
21. Edelsbrunner, H., Seidel, R., Sharir, M.: On the zone theorem for hyperplane arrangements. SIAM J. Comput. **22**(2), 418–429 (1993)
22. Fortune, S.: A sweepline algorithm for voronoi diagrams. Algorithmica **2**, 153–174 (1987)
23. Garey, M.R., Graham, R.L., Johnson, D.S.: The complexity of computing steiner minimal trees. SIAM journal on applied mathematics **32**(4), 835–859 (1977)
24. Georgakopoulos, G.K., Papadimitriou, C.H.: The 1-steiner tree problem. J. Algorithms **8**(1), 122–130 (1987)
25. Gilbert, E., Pollak, H.: Steiner minimal trees. SIAM Journal on Applied Mathematics **16**(1), 1–29 (1968)
26. Guibas, L.J., Stolfi, J.: Ruler, compass and computer. In: Earnshaw, R.A. (ed.) Theoretical Foundations of Computer Graphics and CAD. pp. 111–165. Springer Berlin Heidelberg, Berlin, Heidelberg (1988)
27. Harel, D., Tarjan, R.E.: Fast algorithms for finding nearest common ancestors. SIAM J. Comput. **13**(2), 338–355 (1984)
28. Holby, J.: Variations on the euclidean steiner tree problem and algorithms. Rose-Hulman Undergraduate Mathematics Journal **18**(1), 7 (2017)
29. Li, J., Liu, S., Lichen, J., Wang, W., Zheng, Y.: Approximation algorithms for solving the 1-line euclidean minimum steiner tree problem. J. Comb. Optim. **39**(2), 492–508 (2020)
30. Monma, C.L., Suri, S.: Transitions in geometric minimum spanning trees. Discret. Comput. Geom. **8**, 265–293 (1992)

31. Preparata, F.P., Shamos, M.I.: Computational Geometry - An Introduction. Texts and Monographs in Computer Science, Springer (1985)
32. Rubinstein, J.H., Thomas, D.A., Weng, J.F.: Degree-five steiner points cannot reduce network costs for planar sets. Networks **22**(6), 531–537 (1992)
33. Rubinstein, J.H., Thomas, D.A., Wormald, N.C.: Steiner trees for terminals constrained to curves. SIAM J. Discret. Math. **10**(1), 1–17 (1997)
34. Schieber, B., Vishkin, U.: On finding lowest common ancestors: Simplification and parallelization. SIAM J. Comput. **17**(6), 1253–1262 (1988)
35. Shamos, M.: Computational geometry [ph. d. thesis] (1978)
36. Vanek, J., Galicia, J.A.G., Benes, B.: Clever support: Efficient support structure generation for digital fabrication. Comput. Graph. Forum **33**(5), 117–125 (2014)
37. Winter, P.: Euclidean steiner minimal trees with obstacles and steiner visibility graphs. Discrete Applied Mathematics **47**(2), 187–206 (1993)
38. Winter, P.: Euclidean steiner minimum trees for 3 terminals in a simple polygon. In: Proceedings of the Seventh Canadian Conference on Computational Geometry, Univ. Laval, Quebec, Canada. pp. 247–255 (1995)
39. Winter, P.: Steiner minimum trees in simple polygons. DIMACS Technical Report 95-43 (1995)
40. Winter, P., Zachariasen, M., Nielsen, J.: Short trees in polygons. Discret. Appl. Math. **118**(1-2), 55–72 (2002)
41. Zachariasen, M., Winter, P.: Obstacle-avoiding euclidean steiner trees in the plane: An exact algorithm. In: ALENEX. Lecture Notes in Computer Science, vol. 1619, pp. 282–295. Springer (1999)