PLOS ONE

RESEARCH ARTICLE

An efficient 3-approximation algorithm for the Steiner tree problem with the minimum number of Steiner points and bounded edge length

Donghoon Shin[1], **Sunghee Choi**[2]*

1 Department of Electrical Engineering and Computer Science, DGIST, Daegu, South Korea, **2** School of Computing, KAIST, Daejeon, South Korea

* sunghee@kaist.edu

Citation: Shin D, Choi S (2023) An efficient 3-approximation algorithm for the Steiner tree problem with the minimum number of Steiner points and bounded edge length. PLoS ONE 18(11): e0294353. https://doi.org/10.1371/journal.pone.0294353

Editor: Praveen Kumar Donta, TU Wien: Technische Universitat Wien, AUSTRIA

Received: June 16, 2023

Accepted: October 30, 2023

Published: November 21, 2023

Data Availability Statement: In this paper, as the authors do not utilize any specific dataset, there is no available data to provide.

Funding: D. Shin has been granted by National Research Foundation of Korea(No.NRF-2022R1G1A1011933) and the funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript. No.NRF-2022R1G1A1011933 National Research Foundation of Korea https://www.nrf.re.kr/ S. Choi has been granted by Institute of

Abstract

We present improved algorithms for the Steiner tree problem with the minimum number of Steiner points and bounded edge length. Given n terminal points in a 2D Euclidean plane and an edge length bound, the problem asks to construct a spanning tree of n terminal points with minimal Steiner points such that every edge length of the spanning tree is within the given bound. This problem is known to be NP-hard and has practical applications such as relay node placements in wireless networks, wavelength-division multiplexing(WDM) optimal network design, and VLSI design. The best-known deterministic approximation algorithm has $O(n^3)$ running time with an approximation ratio of 3. This paper proposes an efficient approximation algorithm using the Voronoi diagram that guarantees an approximation ratio of 3 in $O(n \log n)$ time. We also present the first exact algorithm to find an optimal Steiner tree for given three terminal points in constant time. Using this exact algorithm, we improve the 3-approximation algorithm with better performance regarding the number of required Steiner points in $O(n \log n)$ time.

1 Introduction

In this paper, we present improved algorithms for *the Steiner tree problem with the minimum number of Steiner points and bounded edge length* (STP-MSPBEL) introduced by Lin and Xue [1]. Given a set of n terminal points $P = \{p_1, p_2, \ldots, p_n\}$ in the two-dimensional Euclidean space $\mathbb{R}^2$ and a positive constant R, STP-MSPBEL asks for a tree spanning a superset of P such that each edge in the spanning tree has a length no more than R and the number of points other than those in P, called *Steiner points* [2], is minimized [1]. This problem has been applied to the relay node placement problem in wireless sensor networks [3, 4]. In monitoring sensor networks, connectivity among sensors is crucial for gathering information. Due to harsh environments like earthquakes, fires, or floods, a number of sensors can be simultaneously disabled. In this scenario, relay sensors are required to restore network connectivity. When the communication distance sets to R, STP-MSPBEL can be employed to recover the entire network at a low cost (with a minimal number of relay nodes). This problem also finds

Information & communications Technology Planning & Evaluation (IITP)(No.2019-0-01158) and the funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript. No.2019-0-01158 Institute of Information & communications Technology Planning & Evaluation (IITP) https://www.iitp.kr/.

Competing interests: The authors have declared that no competing interests exist.

applications in wavelength-division multiplexing(WDM) optimal network design, VLSI design, and evolutionary/phylogenetic tree constructions in computational biology [1].

The classical Steiner tree problem, which aims to minimize the total length of the spanning tree, has been widely studied. This problem is NP-hard [5], and it is proven to be APX-hard, so it cannot be approximated within 96/95 in polynomial time unless $p = np$ [6]. A number of approximation algorithms have been proposed [7–11]. The currently best-known approximation scheme achieved the approximation ratio of $\ln 4 + \epsilon \leq 1.39$ using linear programming (LP) and iterative randomized rounding by Byrka et al. [7]. Later, Chen and Hsieh [8] proposed an efficient two-phase heuristic in greedy strategy with an approximation ratio of 1.4295. Recently, Traub and Zenklusen [10] proposed the purely combinatorial approach providing the approximation ratio of $\ln 4 + \epsilon$, which is the same as the best-known approximation factor without solving an LP problem. Zhang et al. [11] presented a truthful-in-expectation mechanism that achieves the same approximation ratio. Berman et al. [12] presented a 1.25-approximation algorithm for the variation of the problem where a complete graph is given with edge lengths 1 and 2. Chen and Hsieh [10] analyzed the algorithm of Byrka et al. [7] and showed that it gives $1.162 + \epsilon$ approximation for the problem on complete graphs with edge distances 1 and 2.

In STP-MSPBEL, Lin and Xue [1] proved that the problem is NP-hard and presented a polynomial-time approximation algorithm by exploiting a minimum spanning tree. Let $len(e)$ be the length of an edge e. Given an edge e_i whose length is greater than the given bounded edge length R, this algorithm inserts $\lceil \frac{len(e_i)}{R} \rceil - 1$ Steiner points to break the edge e_i into small pieces of length at most R. Cheng et al. [3] called such edge as a *steinerized edge*. Given a tree, we can produce a *steinerized tree* for STP-MSPBEL using steinerized edges whenever the edge length exceeds the bounded edge length R. Lin and Xue [1] construct a *steinerized minimum spanning tree* from the minimum spanning tree of terminal points [13]. They showed that this algorithm has an approximation ratio of 5. Later, Chen et al. [13] proved that the steinerized minimum spanning tree indeed has an approximation ratio of exactly 4. They also presented a 3-approximation algorithm with $O(n^4)$ running time. At initialization, the algorithm adds edges between any pair of two terminal points if the distance between them is less than or equal to R. The algorithm examines all possible combinations of four terminal points from the entire set to determine if they are currently not connected and can be connected using a single Steiner point. If so, it inserts one Steiner point and four edges to connect those points and the Steiner point. After examining all subsets of four terminal points, the algorithm connects disconnected components using steinerized edges if the resulting graph is still disconnected. Cheng et al. [3] presented a 3-approximation algorithm with $O(n^3)$ running time. Instead of inspecting every subset of four terminal points, they investigate all subsets of three terminal points. Moreover, they proposed a randomized algorithm to give a 2.5 approximation with a probability of at least 0.5 in $poly(n, q_P)$ time where $q_P = max\{q_{a,b,c} | a, b, c \in P\}$ and $q_{a,b,c}$ is the number of Stiner points used to steinerize the optimum Steiner tree on three terminal points $\{a, b, c\}$ [3]. Senel and Younis [14] suggested an $O(n^4)$-time algorithm to reduce Steiner points based on the Fermat point for all subsets of three terminal points. Later, they improved the algorithm with the discrete Fermat point in [15]; however, the time complexity of their algorithm to find the discrete Fermat point is not bounded because the algorithm should examine a large number of candidate points when the distance between input points is long. In summary, the best-known approximation ratio of deterministic algorithms for STP-MSPBEL has been 3 with running time $O(n^3)$.

We propose an efficient approximation algorithm for STP-MSPBEL using the Voronoi diagram, a well-known geometric data structure in computational geometry. This paper extends our previous work [16]. Our algorithm runs in $O(n \log n)$ time with the worst-case approximation ratio of 3, which is the same as the best-known deterministic approximation ratio and

remarkably reduces the running time compared to the previous 3-approximation algorithms [3, 13]. In the classical Steiner tree problem, it is known that the optimal Steiner tree for the given three points can be computed in constant time by using the Fermat point [17]; however, in STP-MSPBEL, there has been no exact algorithm even for three terminal points in constant time. After we presented the exact algorithm in our preliminary paper [16], Senel and Younis [18] proposed a simpler method to find the optimal Steiner tree for three terminal points; however, this paper addresses the counterexample that shows their method indeed cannot find the optimal solution. In this paper, we present the exact algorithm to find a Steiner tree for given three points in constant time, and we propose an improved algorithm for STP-MSPBEL by using the exact algorithm. The proposed algorithm ensures a worst-case approximation ratio of 3 in $O(n \log n)$ time while achieving improved performance since we utilize the optimal solution of three terminal points.

Contributions

- We propose an efficient approximation algorithm for STP-MSPBEL using the Voronoi diagram that guarantees an approximation ratio of 3 in $O(n \log n)$ time. Previously, the best-known deterministic algorithm has an approximation of 3 in $O(n^3)$ time. The proposed algorithm remarkably reduces the running time.
- We present the first exact algorithm to provide an optimal Steiner tree for three input points in constant time. Since no exact algorithm for STP-MSPBEL has been addressed so far, we believe that our algorithm gives a good clue to finding a locally optimal solution. Moreover, it can be used to improve the performance of other approximation algorithms.
- By combining this exact algorithm and the efficient 3-approximation algorithm, we develop another 3-approximation algorithm that runs in $O(n \log n)$ time with better performance in terms of the number of required Steiner points.

2 Preliminaries

We briefly introduce a Voronoi diagram and the Fermat point, mainly used by the proposed algorithms.

The Voronoi diagram

Denote the Euclidean distance between two points p and q by $|pq|$. Let $P = \{p_1, p_2, \ldots, p_n\}$ be a set of n distinct points in the plane; these points are the sites. A *Voronoi diagram* of P denoted by *VD* is defined as a subdivision of the plane into n cells, one for each site in P, with the property that a point q lies in the cell corresponding to a site p_i if and only if $|p_iq| < |p_jq|$ for each $p_j \in P$ with $j \neq i$ [19]. In this paper, *VD* indicates only the edges and vertices of the subdivisions. The following properties hold:

Property 1. *A point q is a vertex of VD if and only if its latest empty circle of q as its center with respect to P contains three or more sites on its boundary.*

Property 2. *The bisector between sites p_i and p_j defines an edge of VD if and only if there is a point q on the bisector such that its latest empty circle of q as its center with respect to P contains both p_i and p_j on its boundary but no other site.*

Under the general position assumption that no four sites lie on the same circle, each vertex of the Voronoi diagram has a degree of three which means the Voronoi vertex has equidistance to three relevant sites. The number of Voronoi vertices is at most $2n - 5$, and the number of Voronoi edges is at most $3n - 6$. It can be efficiently computed in $O(n \log n)$ time [20].

The Fermat point

The *Fermat point* (or called *Torricelli* point) p_F of a triangle $\Delta p_a p_b p_c$ is a point such that the sum of distances from p_F to each vertex ($|p_F p_a| + |p_F p_b| + |p_F p_c|$) is minimized. In the classical Steiner tree problem [21], the Fermat point is used to build the optimal Steiner tree for the given three points. There are two cases to compute the Fermat point. In one case where a triangle has an angle $\geq \frac{2}{3}\pi$, the Fermat point is one of the input points; precisely, the Fermat point is sited at the obtuse-angled. In the other case, the Fermat point is inside the triangle $\Delta p_a p_b p_c$ and can be found by drawing equilateral triangles on each side [17]. In this case, the following property holds:

Property 3. *The angles subtended by the sides of the triangle at the Fermat point are all equal to* $\frac{2}{3}\pi$ *when the Fermat point is inside the triangle.*

3 An efficient 3-approximation algorithm

We propose an $O(n \log n)$-time approximation algorithm with an approximation ratio of at most three by exploiting the Voronoi diagram as described in Fig 1. More precisely, it takes $O(n \log n)$ time to determine the number of Steiner points. It takes $O(n \log n + S)$ to report the locations of Steiner points where S is the number of Steiner points in the optimal Steiner tree for STP-MSPBEL since the algorithm produces no more than three times the number of Steiner points in the optimal Steiner tree. Indeed, the number of Steiner points S is independent of the number of terminal points n. Similar to the previous works [3, 13], this paper does not consider the cost of making steinerized edges when analyzing algorithms. We first compute a minimum spanning tree and sort the edges in increasing order in Step 1. If the length of the edge is

AlgA: A 3-approximation algorithm
Input: A set P of n terminal points, a positive constant R
Output: A Steiner tree T_A

Step 1. Compute a *minimum spanning tree* T_{MST} of P
Sort edges of T_{MST} in increasing order of lengths $e_1, e_2, \ldots, e_{n-1}$
Set $T_A = (P, \phi)$

Step 2. FOREACH e_i in T_{MST}
IF $len(e_i) \leq R$ THEN put e_i into T_A

Step 3. Compute a *Voronoi diagram* G_{VD} of P
FOREACH v_j in G_{VD}
Let p_a, p_b, p_c be the three terminal points(sites) associated with v_j
IF p_a, p_b, p_c are in different connected components of T_A
THEN
IF a triangle $\triangle p_a p_b p_c$ is obtuse AND $\max(|p_a p_b|, |p_a p_c|, |p_b p_c|) \leq 2R$
THEN add one Steiner point q at the middle of the longest side of $\triangle p_a p_b p_c$ and edges (p_a, q), (p_b, q), (p_c, q) into T_A
ELSE IF a triangle $\triangle p_a p_b p_c$ is not obtuse AND $|v_j p_a| \leq R$
THEN add one Steiner point q at the location of v_j and edges (p_a, q), (p_b, q), (p_c, q) into T_A

Step 4. FOREACH $e_i = (p_s, p_e)$ in T_{MST}
IF p_s and p_e are in different connected components of T_A
THEN add the steinerized edge e_i into T_A
Return T_A

Fig 1. *AlgA*: An efficient 3-approximation algorithm.

https://doi.org/10.1371/journal.pone.0294353.g001

less than or equal to R, we insert the edge into the result tree in Step 2. If several connected components still remain after adding edges, it requires Steiner points to connect those components. In Step 3, we construct a Voronoi diagram of the terminal points as sites. At each Voronoi vertex, we investigate its three relevant sites, $\{p_a, p_b, p_c\} \in P$, to determine whether these points are in different connected components and whether the radius of the smallest enclosing circle of three sites is less than R. If so, we add one Steiner point at the center of the circle and insert three edges to connect relevant terminal points from the Steiner point. The center of the smallest enclosing circle for p_a, p_b, and p_c is located at the Voronoi vertex when the triangle $\Delta p_a p_b p_c$ is not an obtuse triangle by property 1. If $\Delta p_a p_b p_c$ is an obtuse triangle, the center of the smallest enclosing circle is located at the middle of the longest side of $\Delta p_a p_b p_c$. Note that if the Voronoi vertex is associated with more than three sites, the same approach can be applied using the same Voronoi vertex. Lastly, if two endpoints of the edges in the minimum spanning tree are in different connected components, we steinerize the edge and insert it into the result tree in Step 4.

Now, we analyze the time complexity and performance of the proposed algorithm.

Lemma 1. *AlgA takes* $O(n \log n)$ *time*.

Proof. In Step 1, it takes $O(n \log n)$ time to compute the minimum spanning tree and to sort the edges of the minimum spanning tree. In Steps 2 and 4, we check at most $n - 1$ edges whether the edge connects two different connected components. Connected components can be efficiently maintained by *disjoint-set forest*, a well-known data structure in which connectivity query can be answered in $O(\log n)$ time, and two trees are merged in $O(1)$ time [22]. Consequently, Steps 2 and 4 take $O(n \log n)$ time. The Voronoi diagram of n sites in the plane can be constructed in $O(n \log n)$ time [4]. Since the number of vertices in the Voronoi diagram is at most $2n - 5$ in 2D, Step 3 also takes in $O(n \log n)$ time. Overall, *AlgA* takes $O(n \log n)$ time.

We show that the *AlgA* needs Steiner points no more than three times the number of Steiner points in the optimal solution in a similar way of [3, 13]. We briefly introduce the terminology used in the previous work. A *full component* of a Steiner tree is a subtree in which every Steiner point is a non-leaf node, and every input point is a leaf node. Given a Steiner tree T, $C(T)$ denotes the number of Steiner points in T.

Lemma 2. [1]. *There exists a shortest length optimal Steiner tree for STP-MSPBEL such that every Steiner point has degree at most five.*

Lemma 3. [13]. *Every steinerized minimum spanning tree has the minimum number of Steiner points among steinerized spanning trees.*

Lemma 4. [13]. *Let* T^* *be a shortest optimal tree for STP-MSPBEL with property that every Steiner point has degree at most five. Let* T_j *be a full component of* T^*. *Then the followings hold*:

(1). *The steinerized minimum spanning tree on terminals in* T_j *contains at most* $3 \cdot C(T_j) + 1$ *Steiner points*.

(2). *If* T_j *contains a Steiner point with degree at most four, then the steinerized minimum spanning tree on terminals in* T_j *contains at most* $3 \cdot C(T_j)$ *Steiner points*.

(3). *If the steinerized minimum spanning tree on terminals in* T_j *contains an edge between two terminals, then it contains at most* $3 \cdot C(T_j)$ *Steiner points*.

We slightly modify the lemma in [13] to apply it to our algorithm.

Lemma 5. *Let* T^* *be an optimal tree for* STP-MSPBEL *and* T_A *be the tree produced by* AlgA. *Then* $C(T_A) \leq 3 \cdot C(T^*)$.

Proof. Let T_S be the steinerized minimum spanning tree. Let k be the number of Steiner points inserted by *AlgA* in Step 3. Since each of those Steiner points combines three different connected components at once and every edge whose length is less than or equal to R in the

minimum spanning tree is added in Step 2, it is obvious that

$$C(T_A) \leq C(T_S) - k. \tag{1}$$

By Lemma 2, we assume that T^* is the shortest length optimal Steiner tree such that every Steiner point has a degree at most five. We split T^* into full components $T_j (1 \leq j \leq N_f)$ where N_f is the number of full components. For each full component T_j, we construct corresponding steinerized minimum spanning trees T_j'. If any full component contains at least one Steiner point whose degree is at most four, the corresponding steinerized minimum spanning tree has not more than 3 times the number of Steiner points in the full components by Lemma 4 (2). Moreover, if the distance between any pair of two points in the full component is not more than R, the corresponding steinerized minimum spanning tree requires no more than 3 times the number of Steiner points in the full components by Lemma 4 (3). Now, we consider a spanning tree $T' (= \cup_{j=1}^{N_f} T_j')$ which consists of steinerized minimum spanning trees of each full component. By Lemma 4,

$$C(T') \leq 3 \cdot C(T^*) + g \tag{2}$$

where g is the number of full components in which every Steiner point has degree 5 and the distance between any two points is greater than R. By Lemma 3,

$$C(T_S) \leq C(T'). \tag{3}$$

From (1), (2) and (3), $C(T_A) \leq 3 \cdot C(T^*) + g - k$. Therefore, the approximation ratio produced by *AlgA* is bounded by 3 if $g \leq k$. To show that $g \leq k$, we construct two forests F_1 and F_2. We put all input points into F_1 and F_2 each and then add edges whose length is less than or equal to R in the minimum spanning tree. Let p be the number of connected components in F_1 (= F_2) so far, which is the result of Step 2 of *AlgA*. We add k Steiner points and their induced edges to F_1 according to Step 3. The number of connected components in F_1 is equal to $p - 2k$.

Now, we examine g full components in which every Steiner point has a degree of 5. Such components are either only one Steiner point or multiple Steiner points. In the case of components with a single Steiner point in which 5 input points are in different connected components and enclosed by a circle with a radius of R, the input points belong to at most three connected components in F_1 after Step 3. We deal with three Voronoi vertices related 5 input points in Step 3. Let us consider one of Voronoi vertices and its relevant three points. If they are in all the different components, they are connected by a Steiner point. Otherwise, at least two of the points are already connected. Then we look into another Voronoi vertex associated with two input points unrelated to the former Voronoi vertex. Similarly, at least two of these points become a part of the same component. Accordingly, we connect two pairs of input points by adding two edges in F_2. Even though we merge them, the number of connected components in F_2 is still greater than or equal to that in F_1. Now, we consider the other case. If the full component has more than one Steiner point, we can always find two Steiner points whose four neighbor points are input points. These four points belong to at most three connected components in F_1 after Step 3. In a similar way to the previous case, at least three points are examined for connectivity due to the associated Voronoi vertex in Step 3. A Steiner point connects them, or two of them are already connected. Even if we add two edges into F_2 to connect a pair of input points at each of two Steiner points, the number of connected components in F_2 is still greater than or equal to that in F_1. In both cases, the number of connected components is equal to $p - 2g$ in F_2 since we add two edges for every g full component. It satisfies that $p - 2k \leq p - 2g$. Therefore, $g \leq k$.

Lemmas 1 and 5 yield the following theorem.

Theorem 1. AlgA *produces a Steiner tree in which the number of Steiner points is no more than 3 times the number of Steiner points in the optimal Steiner tree in* $O(n \log n)$ *time.*

4 An exact algorithm for three input points

Given three terminal points p_a, p_b, and p_c, we present the first exact algorithm *AlgB* to determine the optimal number of Steiner points for STP-MSPBEL in constant time as well as we provide one of the optimal Stiner trees. An optimal tree connecting p_a, p_b, and p_c is formed as either a 3-star or a wedge, as shown in Fig 2. If an optimal Steiner tree is shaped as a wedge, we can easily find the optimal tree because the steinerized minimum spanning tree becomes optimal by Lemma 3. In the other case, p_a, p_b, and p_c are connected by a *junction point* p_J, which is a tree in the shape of a 3-star. In this case, it is important to seek the location of p_J at which $\lceil \frac{|p_a p_J|}{R} \rceil + \lceil \frac{|p_b p_J|}{R} \rceil + \lceil \frac{|p_c p_J|}{R} \rceil$ is minimized. Since the optimal Steiner tree is formed as either a wedge, a 3-star, or both, it can be obtained by comparing the 3-star and the wedge. Since finding a steinerized minimum spanning tree is trivial for three points, the main part of *AlgB* aims to seek an optimal junction point.

The Fermat point p_F can be a good candidate as a junction point of p_a, p_b, and p_c because the Fermat point is defined as a point that minimizes $|p_a p_F| + |p_b p_F| + |p_c p_F|$. In STP-MSPBEL, unfortunately, p_F does not always become an optimal location for the junction point, as shown in Fig 2. Consider an equilateral triangle formed by p_a, p_b, and p_c where each length of a side is $2.25R$. The steinerized minimum spanning tree and a steinerized 3-star tree using the Fermat point require four Steiner points to satisfy bounded edge length R. The optimal Steiner tree, however, needs three Steiner points, as shown in Fig 2(c).

Let T_F be a 3-star tree connecting input points p_a, p_b, and p_c using the Fermat point p_F as a junction point p_J.

Lemma 6. *There is no steinerized 3-star tree* T_{3s} *such that* $C(T_{3s}) < C(T_F) - 2$.

Prrof. It is obvious that $\lceil \frac{|p_a p_F|}{R} \rceil + \lceil \frac{|p_b p_F|}{R} \rceil + \lceil \frac{|p_c p_F|}{R} \rceil - 3 < \frac{|p_a p_F| + |p_b p_F| + |p_c p_F|}{R} \leq \lceil \frac{|p_a p_F|}{R} \rceil + \lceil \frac{|p_b p_F|}{R} \rceil + \lceil \frac{|p_c p_F|}{R} \rceil$ and $C(T_F) = \lceil \frac{|p_a p_F|}{R} \rceil + \lceil \frac{|p_b p_F|}{R} \rceil + \lceil \frac{|p_c p_F|}{R} \rceil - 2$. Suppose that there is a steinerized 3-star tree T_{3s} such that $C(T_{3s}) < C(T_F) - 2$. Let $p_{J'}$ be a junction point of T_{3s}. To satisfy $C(T_{3s}) < C(T_F) - 2$, the following inequalities should hold: $\frac{|p_a p_{J'}| + |p_b p_{J'}| + |p_c p_{J'}|}{R} \leq \lceil \frac{|p_a p_{J'}|}{R} \rceil + \lceil \frac{|p_b p_{J'}|}{R} \rceil + \lceil \frac{|p_c p_{J'}|}{R} \rceil \leq \lceil \frac{|p_a p_F|}{R} \rceil + \lceil \frac{|p_b p_F|}{R} \rceil + \lceil \frac{|p_c p_F|}{R} \rceil - 3$. Since T_F minimizes the sum of edge lengths,

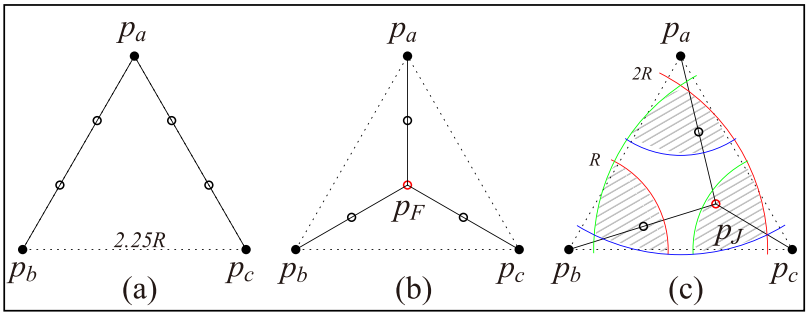

Fig 2. Black dots represent input terminal points p_a, p_b, and p_c. Empty dots represent Steiner points. (a) the steinerized minimum spanning tree requires 4 Steiner points (b) the steinerized 3-star tree using the Fermat point requires 4 Steiner points (c) the optimal Steiner tree for STP-MSPBEL requires 3 Steiner points. A junction point can be optimally placed in any of 3 shaded regions.

https://doi.org/10.1371/journal.pone.0294353.g002

$\lceil \frac{|p_a p_F|}{R} \rceil + \lceil \frac{|p_b p_F|}{R} \rceil + \lceil \frac{|p_c p_F|}{R} \rceil - 3 < \frac{|p_a p_F| + |p_b p_F| + |p_c p_F|}{R} < \frac{|p_a p_{I'}| + |p_b p_{I'}| + |p_c p_{I'}|}{R}$. It contradicts that $\lceil \frac{|p_a p_{I'}|}{R} \rceil + \lceil \frac{|p_b p_{I'}|}{R} \rceil + \lceil \frac{|p_c p_{I'}|}{R} \rceil \leq \lceil \frac{|p_a p_F|}{R} \rceil + \lceil \frac{|p_b p_F|}{R} \rceil + \lceil \frac{|p_c p_F|}{R} \rceil - 3$. Thus, such T_{3s} cannot exist.

Let C_x^y be a circle centered at a point p_x with a radius of $y \cdot R$. We subdivide the plane by circles centered at each input point p_a, p_b, and p_c with various radii; we draw concentric circles C_a^q, C_b^q, and C_c^q, $q \in \mathbb{N}$ as illustrated in Fig 4.

Observation 1. *When the optimal Steiner tree of three points is formed as 3-star, the optimal location of p_J is inside the convex cell, which is an intersection area of $C_a^{i^*}$, $C_b^{j^*}$, and $C_c^{k^*}$ such that $i^* + j^* + k^*$ is minimized. Note that an optimal convex cell can be a single point if circles intersect at a point*.

Proof. It is obvious that any point p_x in C_a^q (or C_b^q, C_c^q) requires at least $q - 1$ Steiner points to reach p_a (or p_b, p_c). When p_x is a junction point and C_a^i, C_b^j, and C_c^k are the smallest circles containing p_x, we need $i + j + k - 2$ Steiner points including p_x to connect all input points. If p_x is not in the convex cell, we can always find the cell in which any point $p_{x'}$ can connect p_a, p_b, and p_c with fewer Steiner points. Suppose that the cell containing p_x has at least one concave arc on its boundary and w.l.o.g., we assume that this arc is the part of circle C_a^{i-1}. Since p_x is outside of C_a^{i-1}, any point $p_{x'}$ contained by C_a^{i-1}, C_b^j, and C_c^k requires one fewer Steiner points than that of p_x. Therefore, an optimal junction point p_J should be inside of the convex region generated by $C_a^{i^*}$, $C_b^{j^*}$, and $C_c^{k^*}$ such that $i^* + j^* + k^*$ is minimum.

Based on Observation 1, it is easy to show that the optimal location of a junction point can be represented as a region rather than a single point. Therefore, we focus on searching for a convex region induced by the intersection of circles; more precisely, we search for an optimal region among convex cells in the plane subdivided by concentric circles.

Observation 2. *An optimal convex cell may not be unique*.

In the example of Fig 2(c), there are three cells in which we can optimally put a junction point. As the distance between terminal points increases, the number of convex regions requiring the same number of Steiner points can be highly many. Indeed, the number of optimal regions is not bounded by the input size. In this paper, we provide an algorithm to find at least one optimal convex cell to construct the Steiner tree, which requires the least number of Steiner points among all possible 3-star trees connecting p_a, p_b, and p_c.

Observation 3. *The intersection of triangle $\Delta p_a p_b p_c$ and optimal regions for the junction point of p_a, p_b, and p_c is not empty*.

Proof. Let $\overline{pq}$ be a line through two points p and q. Suppose that an optimal convex cell is completely outside of $\Delta p_a p_b p_c$ and w.l.o.g., this convex cell lies on the opposite side of $\overline{p_a p_b}$ from p_c. The boundary of the convex cell does not consist of C_c^k, $k \in \mathbb{N}$ since p_c is on the opposite side. If the convex cell consists of C_a^i and C_b^j, it should stab the line $\overline{p_a p_b}$. Therefore, the optimal convex cells are either completely inside of $\Delta p_a p_b p_c$ or on the edges of $\Delta p_a p_b p_c$.

We present *AlgB*, which provides an optimal Steiner tree for three input points as described in Fig 3. We consider a 3-star tree first and then compare it with the minimum spanning tree. The convex region formed by the intersection of three circles C_a^i, C_b^j, and C_c^k is denoted by $CV(C_a^i, C_b^j, C_c^k)$. Let $I_a(C_b^j, C_c^k)$ be the intersection point of C_b^j and C_c^k that lies on the same side of $\overline{p_b p_c}$ as p_a or on $\overline{p_b p_c}$, in other words, the intersection point closer to p_a.

In the first phase, we aim to construct an optimal 3-star tree which induces a Steiner tree requiring the minimum number of Steiner points under the bounded edge length R among all possible 3-star trees (refer to Step 1 and 2 of *AlgB* in Fig 3). By Observations 1 and 3, we can find the optimal location for the junction point by checking all convex cells in $\Delta p_a p_b p_c$. Because there could be a number of optimal regions in $\Delta p_a p_b p_c$ by Observation 2, the algorithm focuses on finding one optimal convex cell. The first phase consists of three sub-parts symmetrically in

AlgB: An exact algorithm for 3 terminal points
Input: 3 terminal points p_a, p_b, p_c, a positive constant R
Output: A Steiner tree T_B

Step 1. Compute the Fermat point p_F of $\triangle p_a p_b p_c$
Set $CV(C_a^{i_F}, C_b^{j_F}, C_c^{k_F})$ as *candidate* where $i_F = \lceil \frac{|p_a p_F|}{R} \rceil$, $j_F = \lceil \frac{|p_b p_F|}{R} \rceil$, $k_F = \lceil \frac{|p_c p_F|}{R} \rceil$
$t = i_F + j_F + k_F - 2$

Step 2. Step 2 consists of three sub-parts based on each terminal point: step 2-1 for p_a, step 2-2 for p_b, and step 2-3 for p_c.

Step 2-1. (Assume that $|p_b p_F| \geq |p_c p_F|$. If not, exchange p_b for p_c.)
Solve Equation (6) with a set of parameters $(i, j, k) \in \{(i_F - 2, j_F, k_F), (i_F - 3, j_F, k_F + 1), (i_F - 4, j_F, k_F + 2), (i_F - 3, j_F + 1, k_F)\}$ and find $CV(C_a^i, C_b^j, C_c^k)$ such that $i + j + k - 2 = t - 2$
IF there exists $CV(C_a^i, C_b^j, C_c^k)$ such that $i + j + k - 2 = t - 2$,
THEN set $CV(C_a^i, C_b^j, C_c^k)$ as *candidate* and goto Step 3

Solve Equation (6) with a set of parameters $(i, j, k) \in \{(i_F - 1, j_F, k_F), (i_F - 2, j_F, k_F + 1), (i_F - 3, j_F, k_F + 2), (i_F - 2, j_F + 1, k_F)\}$ and find $CV(C_a^i, C_b^j, C_c^k)$ such that $i + j + k - 2 = t - 1$
IF there exists $CV(C_a^i, C_b^j, C_c^k)$ such that $i + j + k - 2 = t - 1$,
THEN set $CV(C_a^i, C_b^j, C_c^k)$ as *candidate*

Step 2-2. and 2-3. Do the same process as Step 2-1 with different parameters.

Step 3. Compute a *minimum spanning tree* T_{MST} of 3 terminal points
(Let (p_1, p_2), (p_2, p_3) be two edges of T_{MST})
$C(T_{MST}) = \lceil \frac{|p_1 p_2|}{R} \rceil + \lceil \frac{|p_2 p_3|}{R} \rceil - 2$
Construct T_{3s} by adding a junction point p_J in *candidate* and edges (p_a, p_J), (p_b, p_J), (p_c, p_J)
$C(T_{3s}) = \lceil \frac{|p_a p_J|}{R} \rceil + \lceil \frac{|p_b p_J|}{R} \rceil + \lceil \frac{|p_c p_J|}{R} \rceil - 2$
IF $C(T_{3s}) < C(T_{MST})$ THEN $T_B = T_{3s}$
ELSE $T_B = T_{MST}$
Return steinerized T_B

Fig 3. *AlgB*: An exact algorithm for three terminal points.

https://doi.org/10.1371/journal.pone.0294353.g003

which each input point is used as a reference point. W.l.o.g., we explain the sub-part on the basis of a point p_a. By Lemma 6, the number of Steiner points for an optimal 3-star tree is between $i_F + j_F + k_F - 4$ and $i_F + j_F + k_F - 2$ where $i_F = \lceil \frac{|p_a p_F|}{R} \rceil$, $j_F = \lceil \frac{|p_b p_F|}{R} \rceil$, and $k_F = \lceil \frac{|p_c p_F|}{R} \rceil$. In the case that the optimal number of Steiner points is equal to $i_F + j_F + k_F - 2$, the Fermat point becomes one of the optimal junction points. Otherwise, the algorithm searches for the convex cell $CV(C_a^i, C_b^j, C_c^k)$ such that $i+ j+ k-2$ is equal to $i_F+ j_F+ k_F-3$ or $i_F+ j_F+ k_F-4$. We first assume that the optimal 3-star tree requires $i_F + j_F + k_F - 4$ Steiner points. To figure out the existence of the target convex cell, we consider the series of particular convex cells which form $CV(C_a^{i-2\delta_z}, C_b^{j+\delta_z}, C_c^{k+\delta_z})$ where $i + j + k = i_F + j_F + k_F - 2$ and $i, j, k, \delta_z \in \mathbb{Z}$. When j and k are given, we check whether there exists δ_z satisfying that $CV(C_a^{i-2\delta_z}, C_b^{j+\delta_z}, C_c^{k+\delta_z})$ is not empty where $i = i_F + j_F + k_F - j - k - 2$. We build a distance function $d(j, k, \delta_r)$, $\delta_r \in \mathbb{R}$ that signifies the distance between p_a and $I_a(C_b^{j+\delta_r}, C_c^{k+\delta_r})$. If there exists any integer value δ_z^* satisfying $d(j, k, \delta_z^*) \leq (i - 2\delta_z^*)R$, there exists a convex cell $CV(C_a^{i-2\delta_z^*}, C_b^{j+\delta_z^*}, C_c^{k+\delta_z^*})$. Given i, j, and k, we can find the interval of δ_r to satisfy the following inequality:

$$f(i, j, k, \delta_r) = d(j, k, \delta_r) - (i - 2\delta_r)R \leq 0. \quad (4)$$

If the intervals of δ_r satisfying $f(i, j, k, \delta_r) \leq 0$ include any integer value δ_z^*, $CV(C_a^{i-2\delta_z^*}, C_b^{j+\delta_z^*}, C_c^{k+\delta_z^*})$ is nominated for a candidate cell. If there is no convex cell satisfying $i + j + k - 2 = i_F + j_F + k_F - 4$, the algorithm seeks convex cells which require $i_F + j_F + k_F - 3$ Steiner points in the same manner.

For easy representation, we use the similarity transform by setting $p_b = (0, 0)$ and $p_c = (1, 0)$. The scale factor of the transform is $\frac{1}{|p_b p_c|}$. Accordingly, p_a is transformed to $p'_a = (x_a, y_a)$ and bounded edge length R becomes $R' = \frac{R}{|p_b p_c|}$. For given j and k, the coordinate of transformed $I_a(C_b^{j+\delta_r}, C_c^{k+\delta_r})$ is represented by (x_i, y_i) where $x_i = \frac{(jR'+\delta_r R')^2 - (kR'+\delta_r R')^2 + 1}{2}$ and $y_i = \sqrt{(jR' + \delta_r R')^2 - x_i^2}$. Now, we rewrite the function $f(i, j, k, \delta_r)$ in (4) as follows.

$$f'(i, j, k, \delta_r) = d'(j, k, \delta_r) - (i - 2\delta_r)R' \tag{5}$$

where $d'(j, k, \delta_r) = \sqrt{(x_i - x_a)^2 + (y_i - y_a)^2}$.

By substituting $\delta'_r - \frac{j+k}{2} + \frac{1}{2R'}$ for δ_r, we can rewrite the function $f'(i, j, k, \delta_r)$ as follows:

$$\begin{aligned} f'(i, j, k, \delta'_r) = \sqrt{(((2a-1)R'\delta'_r + a) - x_a)^2 + (2\sqrt{a(1-a)R'\delta'_r(1 + R'\delta'_r)} - y_a)^2} \\ - R'(t + 2 - 2\delta'_r) + 1 \end{aligned} \tag{6}$$

where $a = \frac{(j-k)R'+1}{2}$ and t is the desired number of Steiner points. Note that t is initially set to $i_F + j_F + k_F - 4$ and later is set to $i_F + j_F + k_F - 3$. We can solve Eq (6) by the quartic formula since $f'(i, j, k, \delta'_r)$ is of quartic form of δ'_r for given i, j, and k. We calculate the valid interval of δ_r from δ'_r satisfying inequality (7). If we find an integer value δ_z^* in the valid interval of δ_r, $CV(C_a^{i-2\delta_z^*}, C_b^{j+\delta_z^*}, C_c^{k+\delta_z^*})$ is set to a candidate cell.

$$f'(i, j, k, \delta'_r) \leq 0. \tag{7}$$

Once we set the desired number of Steiner points t, we should investigate $CV(C_a^{i-2\delta_z}, C_b^{j+\delta_z}, C_c^{k+\delta_z})$ with all possible combinations of j and k. In this paper, we reveal that only four pairs of j and k are enough to find the optimal cell for each of the two i values in each sub-part. In other words, instead of checking all convex cells, we solve Eq (6) at most 8 times in each sub-part with the parameters below. W.l.o.g., we assume that $|p_b p_F| \geq |p_c p_F|$. If $|p_b p_F| < |p_c p_F|$, we regard p_b as p_c and p_c as p_b. In the case that the target number of Steiner points t is $i_F + j_F + k_F - 4$, we assign a set of parameters for Eq (6) to $(i, j, k) \in \{(i_F - 2, j_F, k_F), (i_F - 3, j_F, k_F + 1), (i_F - 4, j_F, k_F + 2), (i_F - 3, j_F + 1, k_F)\}$. If there exists any convex cell satisfying the condition, the convex cell becomes the optimal solution for the 3-star tree. Otherwise, we increase the i value of the above set of parameters by 1 and solve the equations again in order to examine whether there exists a 3-star tree requiring $i_F + j_F + k_F - 3$ Steiner points. If there is still no convex cell satisfying the condition, $CV(C_a^{i_F}, C_b^{j_F}, C_c^{k_F})$ becomes the optimal solution for the 3-star tree.

In the second phase, we construct the minimum spanning tree T_{MST} of terminal points as described in Step 3. If the number of required Steiner points of T_{MST} is less than that of the optimal 3-star tree T_{3S}, we return steinerized T_{MST} as an optimal Steiner tree for the three terminal points. Otherwise, we return steinerized T_{3S}.

Now, we prove that the result tree of *AlgB* is an optimal Steiner tree for STP-MSPBEL given three terminal points. We begin by demonstrating that the steinerized 3-star tree, as generated through Step 2, demands the fewest Steiner points compared to all possible 3-star trees connecting terminal points. This is achieved by comparing the candidate convex cell produced by

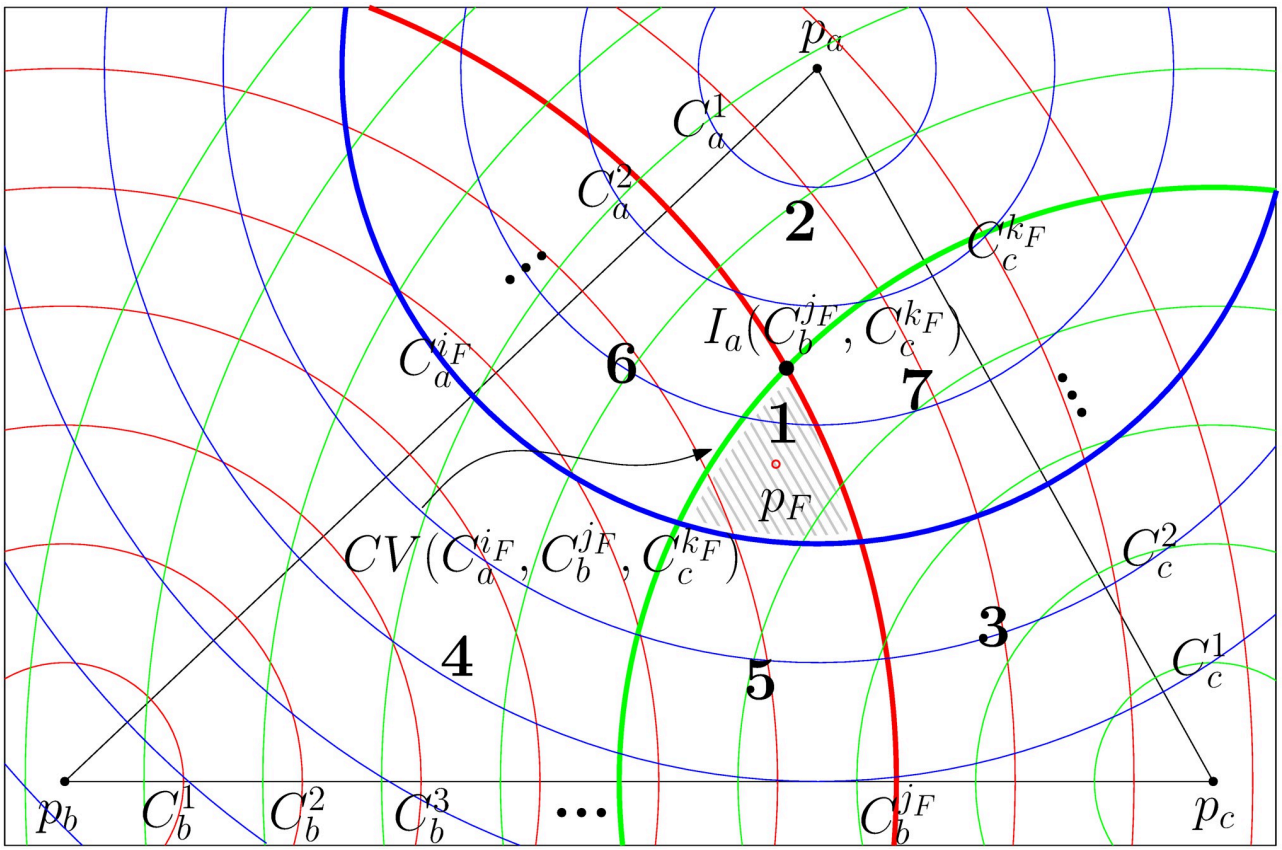

Fig 4. The optimal location for a junction point of a 3-star tree can be identified within the convex cells formed by concentric circles centered at each terminal point. The triangle $\Delta p_a p_b p_c$ is partitioned into 7 regions by $C_a^{i_F}$, $C_b^{j_F}$, and $C_c^{k_F}$.

https://doi.org/10.1371/journal.pone.0294353.g004

AlgB with all convex cells that are in the triangle $\Delta p_a p_b p_c$ or intersect with the boundary of the triangle $\Delta p_a p_b p_c$. We partition the triangle $\Delta p_a p_b p_c$ into seven distinct regions, delineated by $C_a^{i_F}$, $C_b^{j_F}$, and $C_c^{k_F}$ as illustrated in Fig 4. Subsequently, we investigate convex cells within each respective region.

First, we look into region 1, the intersection area of $C_a^{i_F}$, $C_b^{j_F}$, and $C_c^{k_F}$, which indicates the smallest convex cell containing the Fermat point.

Lemma 7. *No convex cell* $CV(C_a^i, C_b^j, C_c^k)$ *exists such that* $i + j + k < i^* + j^* + k^*$, $i \leq i_F, j \leq j_F$, *and* $k \leq k_F$ *where* $CV(C_a^{i^*}, C_b^{j^*}, C_c^{k^*})$ *is the candidate cell for the junction point of a 3-star tree as the result of* AlgB.

Proof. In region 1, a convex cell $CV(C_a^i, C_b^j, C_c^k)$ satisfies that $i + j + k \geq i_F + j_F + k_F - 2$, $i_F - 2 \leq i \leq i_F$, $j_F - 2 \leq j \leq j_F$, and $k_F - 2 \leq k \leq k_F$ by Lemma 6. In Step 2-1 of *AlgB*, convex cells $CV(C_a^{i_F-\delta 1}, C_b^{j_F-\delta 2}, C_c^{k_F-\delta 2}), \delta 1 \in \{0, 1, 2\}, \delta 2 \in \{0, 1, 2\}$ are investigated by Inequality (7) with parameters $(i, j, k) \in \{(i_F - 2, j_F, k_F), (i_F - 1, j_F, k_F)\}$. Also, convex cells $CV(C_a^{i_F-\delta 2}, C_b^{j_F-\delta 1}, C_c^{k_F-\delta 2})$ and $CV(C_a^{i_F-\delta 2}, C_b^{j_F-\delta 2}, C_c^{k_F-\delta 1}), \delta 1 \in \{0, 1, 2\}, \delta 2 \in \{0, 1, 2\}$ are checked in Step 2-2 and 2-3 respectively. Consequently, all convex cells are investigated in $CV(C_a^{i_F}, C_b^{j_F}, C_c^{k_F})$ by *AlgB*.

To prove Lemmas 8 and 9, we partition the plane into six wedges centered around the Fermat point p_F using the lines $\overline{p_a p_F}$, $\overline{p_b p_F}$, and $\overline{p_c p_F}$. Note that each wedge has an interior angle of

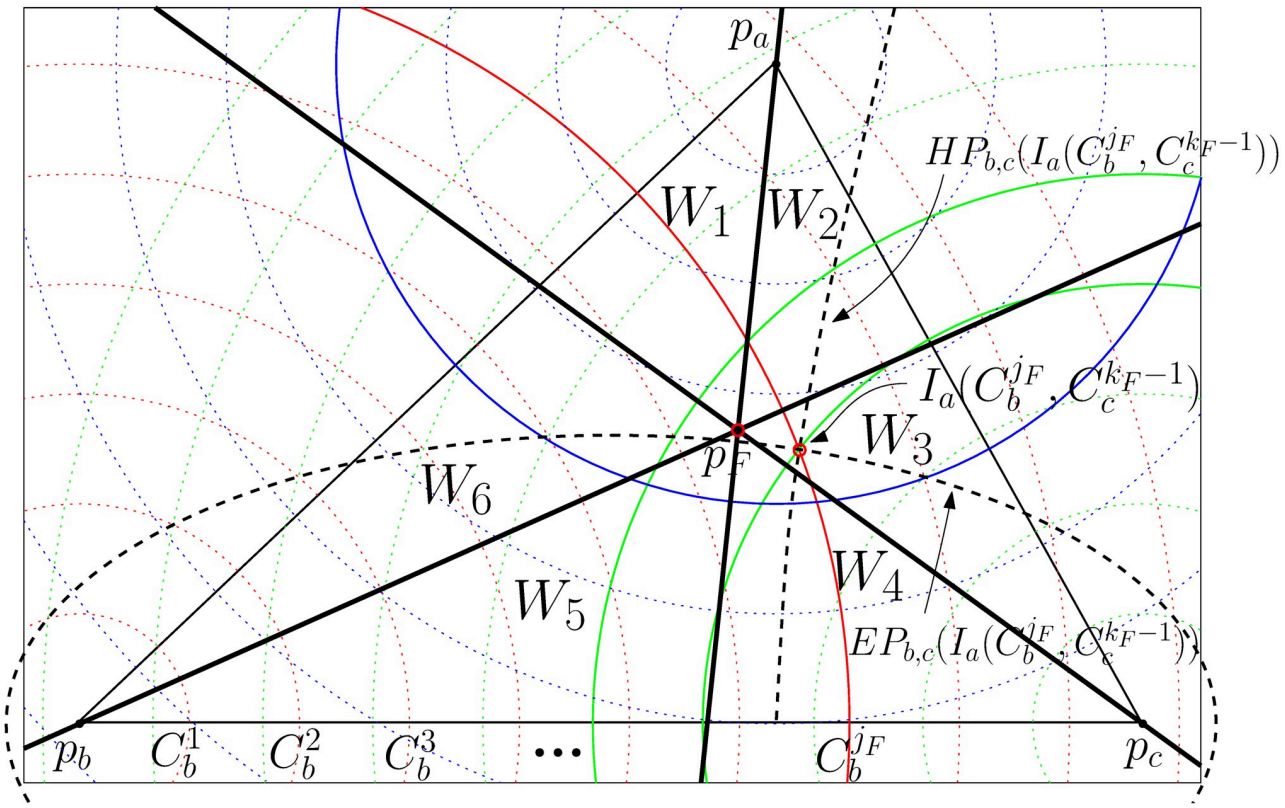

Fig 5. 6 wedges ($W_1, \cdots, W_6$) are delineated by lines $\overline{p_a p_F}$, $\overline{p_b p_F}$, and $\overline{p_c p_F}$ (thick lines). $I_a(C_b^{j_F}, C_c^{k_F-1})$ denotes an intersection point of circles $C_b^{j_F}$ and $C_c^{k_F-1}$, positioned closer to p_a. Dotted curves represent an ellipse $EP_{b,c}(I_a(C_b^{j_F}, C_c^{k_F-1}))$ and a hyperbola $HB_{b,c}(I_a(C_b^{j_F}, C_c^{k_F-1}))$ with two focal points p_b and p_c through a point $I_a(C_b^{j_F}, C_c^{k_F-1})$.

https://doi.org/10.1371/journal.pone.0294353.g005

60˚. As illustrated in Fig 5, we denote W_1 as the wedge on the same side as p_b among two adjacent wedges of a line segment (p_a, p_F). Accordingly, we denote $W_2, \ldots, W_6$ in clockwise order.

Lemma 8. *If* $I_a(C_b^j, C_c^k) \in (W_1 \cup W_2)$ *and* $I_a(C_b^j, C_c^k) \notin C_a^i$, *no convex cell* $CV(C_a^{i-2\delta_N}, C_b^{j+\delta_N}, C_c^{k+\delta_N})$ *exists where* $\delta_N < i/2$, *and* $i, j, k, \delta_N \in \mathbb{N}$.

Proof. Given a convex cell, we consider two cases: the first case is when one of vertices of the convex cell is closer to p_a ($I_a(C_b^j, C_c^k) \in \Delta p_a p_b p_c$) and the second case is when the closest point to p_a is on the boundary. In the first case, we use $I_a(C_b^j, C_c^k)$ to determine whether C_a^i contains $I_a(C_b^j, C_c^k)$. In the case that $I_a(C_b^j, C_c^k) \notin \Delta p_a p_b p_c$, the intersection point of $CV(C_a^i, C_b^j, C_c^k)$ and the boundary of $\Delta p_a p_b p_c$ is used for proof instead of $I_a(C_b^j, C_c^k)$.

We explain with the case that $I_a(C_b^j, C_c^k) \in \Delta p_a p_b p_c$. We show that $|p_a p_I| - 2R \leq |p_a p_{I+1}|$ for any $p_I = I_a(C_b^j, C_c^k), j \geq j_F, k \geq k_F$, and $p_{I+1} = I_a(C_b^{j+1}, C_c^{k+1})$ which implies that p_{I+1} cannot be contained by C_a^{i-2} unless p_I lies inside C_a^i. As depiced in Fig 6, we denote $l = |p_I p_{I+1}|$ as the distance between p_I and p_{I+1}. If $l \geq 2R$, $\angle p_b p_I p_{I+1} < \frac{2}{3}\pi$ (see Fig 6(b)). Since $p_I \in W_1 \cup W_2$, $\angle p_b p_I p_c < \frac{2}{3}$. Thus, it should hold that $\angle p_b p_I p_{I+1} + \angle p_c p_I p_{I+1} > \frac{4}{3}\pi$. It implies that $l < 2R$. $l + |p_a p_{I+1}| \geq |p_a p_I|$ by triangle inequality. It holds that $|p_a p_{I+1}| \geq |p_a p_I| - l \geq |p_a p_I| - 2R$.

Consequently, $CV(C_a^{i-2\delta_N}, C_b^{j+\delta_N}, C_c^{k+\delta_N})$ does not exist.

Let $EP_{b,c}(p_x)$ be the ellipse through a point p_x with two focal points p_b and p_c (see Fig 5). Let $HB_{b,c}(p_x)$ be the p_c-side curve of the hyperbola with two focal points p_b and p_c through a point p_x.

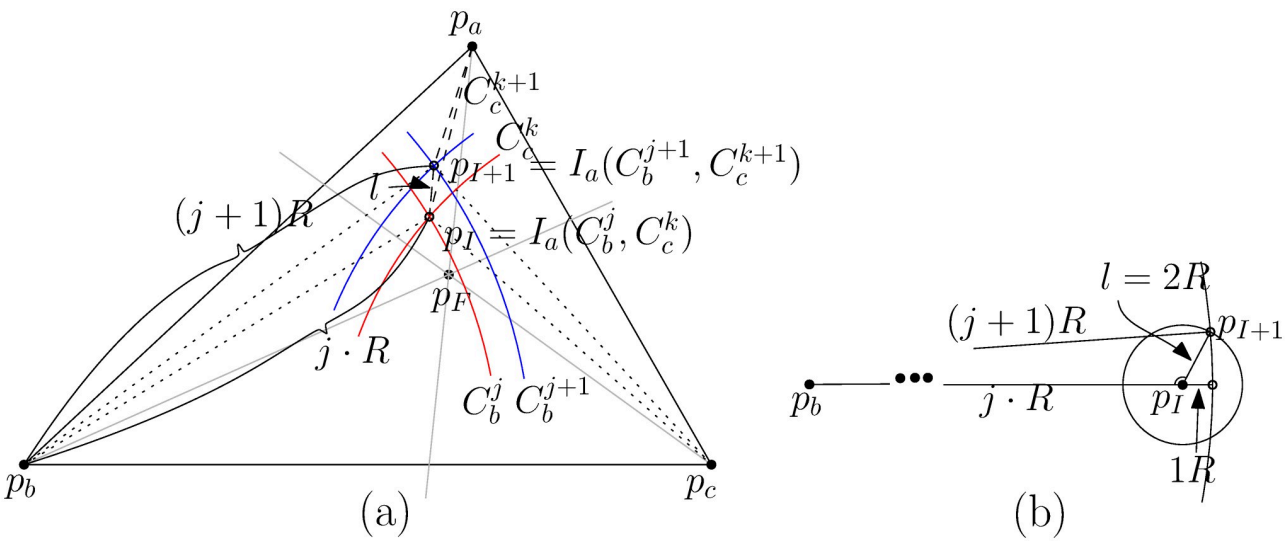

Fig 6. (a) The distance l between the intersection point $p_l = I_a(C_b^j, C_c^k)$ of C_b^j and C_c^k, and $p_{I+1} = I_a(C_b^{j+1}, C_c^{k+1})$. (b) For $l \geq 2R$, the angle $\angle_{p_b p_I p_{I+1}}$ is strictly less than $\frac{2}{3}\pi$.

https://doi.org/10.1371/journal.pone.0294353.g006

Lemma 9. *If* $I_a(C_b^j, C_c^k) \in W_1$ (or W_2) *and* $I_a(C_b^j, C_c^k) \notin C_a^i$, *no convex cell* $CV(C_a^{i-\delta_N}, C_b^j, C_c^{k+\delta_N})$ (*or* $CV(C_a^{i-\delta_N}, C_b^{j+\delta_N}, C_c^k)$), $\delta_N < i, \delta_N \in \mathbb{N}$, *exists*.

Proof. We consider a circle $C_a^{i+\epsilon}$ centered at p_a with a radius of $(i+\epsilon)R = |p_a I_a(C_b^j, C_c^k)|$ and an ellipse $EP_{a,c}(I_b(C_a^{i+\epsilon}, C_b^j))$ as illustrated in Fig 7. Note that $I_a(C_b^j, C_c^k) = I_b(C_a^{i+\epsilon}, C_c^k)$. When one of the intersection points of the ellipse and the circle lies inside of W_1, the other intersection point is located in $W_2 \cup W_3 \cup W_4$ since $|p_b p_F| \leq j \cdot R$. It implies that $I_a(C_b^j, C_c^{k+\delta}) \in \Delta p_a p_b p_c, \delta > 0$ is outside of $EP_{a,c}(I_b(C_a^{i+\epsilon}, C_c^k))$ as well as $EP_{a,c}(I_b(C_a^{i-\delta}, C_c^{k+\delta}))$. Thus, $CV(C_a^{i-\delta_N}, C_b^j, C_c^{k+\delta_N})$ does not exist unless $I_a(C_b^j, C_c^k) \in C_a^i$.

Now, we look into the cells in regions 2, 3, and 4. W.l.o.g., we show that there is no better convex cell compared to the candidate cell of the algorithm in region 2. We assume that $|p_b p_F| \geq |p_c p_F|$ for the proof of Lemmas 10 and 11.

Lemma 10. *No convex cell* $CV(C_a^i, C_b^j, C_c^k)$ *exists such that* $i + j + k < i^* + j^* + k^*$, $i \leq i_F, j \geq j_F$, *and* $k \geq k_F$ *where* $CV(C_a^{i^*}, C_b^{j^*}, C_c^{k^*})$ *is the candidate cell as the result of* AlgB.

Proof. *AlgB* investigates $CV(C_a^i, C_b^{j_F}, C_c^{k_F+2})$, $CV(C_a^i, C_b^{j_F}, C_c^{k_F+1})$, $CV(C_a^i, C_b^{j_F}, C_c^{k_F})$, and $CV(C_a^i, C_b^{j_F+1}, C_c^{k_F})$ by solving Inequality (7) in Step 2-1. We prove that there is no better cell in region 2 by comparing it with these four convex cells.

$EP_{b,c}(p_F)$ is tangent to the circle centered at p_a with a radius of $|p_a p_F|$ and tangent lines of $EP_{b,c}(p_F)$ and $HB_{b,c}(p_F)$ are orthogonal at a point p_F. Thus, $\overline{p_a p_F}$ becomes a tangent line of $HB_{b,c}(p_F)$ at p_F and $HB_{b,c}(p_F)$ lies in $W_2 \bigcup W_3 \bigcup W_4$. Since $|p_b p_F| - |p_c p_F| < (j_F + 1 - k_F)R$, $HB_{b,c}(I_a(C_b^{j_F+1}, C_c^{k_F}))$ is also in $W_2 \bigcup W_3 \bigcup W_4$. Thus, $I_a(C_b^{j_F+1}, C_c^{k_F})$ is in W_2. By Lemma 9, convex cells composed by $C_c^{k_F}$ and $C_b^{j_F+1+\delta_N}$, $\delta_N \in \mathbb{N}$ require at least the same number of Steiner points as that of the convex cell consisting of $C_b^{j_F+1}$ and $C_c^{k_F}$.

$I_a(C_b^{j_F}, C_c^{k_F+2})$ is in W_1 because $j_F R - |p_b p_F| < R$ and the distance between two intersection points of W_2 and $C_b^{j_F}$ is less than $\sqrt{3}R$. By Lemma 9, convex cells composed by $C_b^{j_F}$ and $C_c^{k_F+2+\delta_N}, \delta_N \in \mathbb{N}$ require Steiner points not less than that of the convex cell consisting of $C_b^{j_F}$ and $C_c^{k_F+2}$.

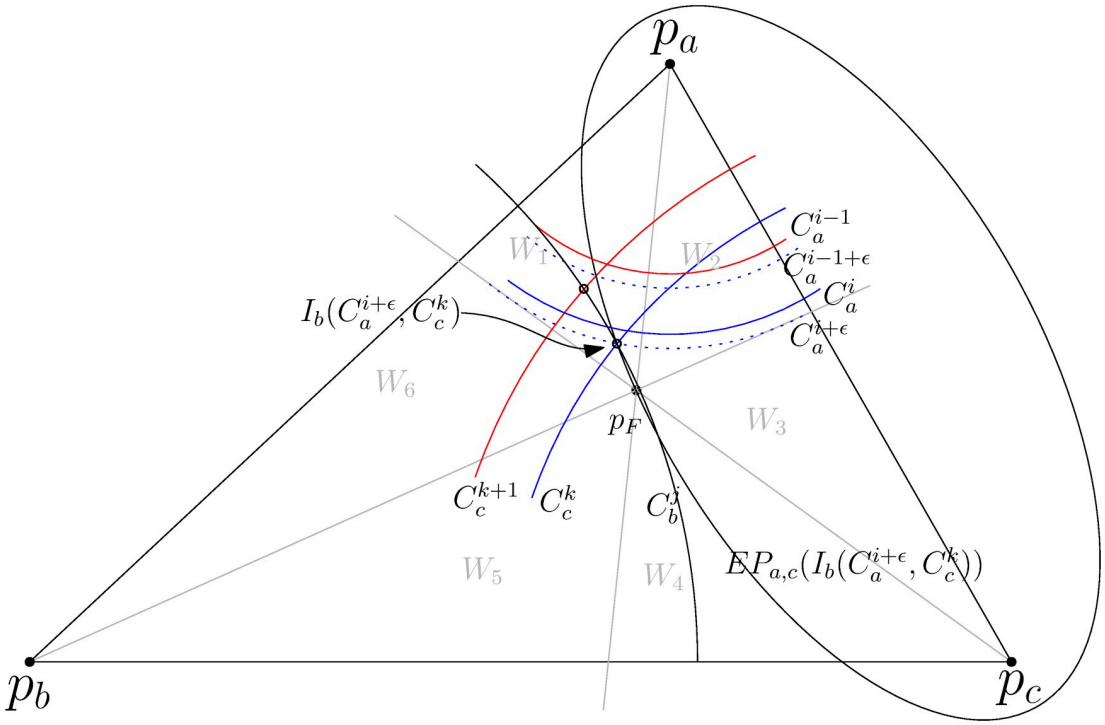

Fig 7. When one of intersection points between $EP_{b,c}(p_x)$ and C_b^j where $j \cdot R > |p_b p_F|$ lies inside of W_1, the other point is inside of $W_2 \cup W_3 \cup W_4$ if it exists inside of the triangle.

https://doi.org/10.1371/journal.pone.0294353.g007

Up to now, we have examined the convex cells $CV(C_a^i, C_b^j, C_c^k)$ such that $i \leq i_F$ and either $j = j_F$ or $k = k_F$. Next, we consider a convex cell $CV(C_a^i, C_b^j, C_c^k)$ such that $i \leq i_F$, $j = j_F + \delta 1_N$, $k = k_F + \delta 2_N$, and $\delta 1_N, \delta 2_N \in \mathbb{N}$. When $\delta 1_N \geq \delta 2_N$, we compare the convex cell $CV(C_a^i, C_b^j, C_c^k)$ with the convex cell CV_{comp} composed by $C_b^{j_F+\delta 1_N-\delta 2_N}$ and $C_c^{k_F}$. By Lemma 8, $CV(C_a^i, C_b^j, C_c^k)$ is not better than CV_{comp} which is already checked above. Equally, we compare the convex cell $CV(C_a^i, C_b^j, C_c^k)$ with the convex cell CV_{comp} composed by $C_b^{j_F}$ and $C_c^{k_F+\delta 2_N-\delta 1_N}$ when $\delta 1_N < \delta 2_N$. By Lemma 8, $CV(C_a^i, C_b^j, C_c^k)$ is also not better than CV_{comp}. Consequently, there does not exist $CV(C_a^i, C_b^j, C_c^k)$ such that $i + j + k < i^* + j^* + k^*$, $i \leq i_F$, $j \geq j_F$, and $k \geq k_F$ where $CV(C_a^{i^*}, C_b^{j^*}, C_c^{k^*})$ is the candidate cell as the result of *AlgB*.

Lastly, we prove that there is no better convex cell in regions 5, 6, and 7 than the candidate cell. W.l.o.g., we show the case of region 5.

Lemma 11. *No convex cell* $CV(C_a^i, C_b^j, C_c^k)$ exists such that $i + j + k < i^* + j^* + k^*$, $i > i_F$, $j < j_F$, and $k < k_F$ where $CV(C_a^{i^*}, C_b^{j^*}, C_c^{k^*})$ is the candidate cell as the result of *AlgB*.

Proof. $HB_{b,c}(p_F)$ is tangent to $\overline{p_a p_F}$ at p_F as explained in Lemma 10. Since the algorithm investigates all convex cells of which one of vertices is on $HB_{b,c}(I_a(C_b^j, C_c^k))$, $j_F - k_F - 2 \leq j - k \leq j_F - k_F + 1$, we first look up convex cells $CV(C_a^i, C_b^j, C_c^k)$ such that $i > i_F$, $j < j_F$, $k < k_F$, and $j - k < j_F - k_F - 2$ and then $CV(C_a^i, C_b^j, C_c^k)$ such that $i > i_F$, $j < j_F$, $k < k_F$, and $j - k > j_F - k_F + 1$. If an ellipse with two focal points p_b and p_c intersects with C_a^i, $i > i_F$ at a single point, the point is in the same side of $HB_{b,c}(p_F)$ as p_c. When the ellipse intersects with C_a^i, $i > i_F$ at two points, if one intersection point is on the same side of $HB_{b,c}(p_F)$ as p_b, the other intersection point should be in the opposite side of $HB_{b,$

$_c(p_F)$. Therefore, if there exists a convex cell $CV(C_a^i, C_b^j, C_c^k)$ such that $i + j + k < i^* + j^* + k^*$, $i > i_F$, $j < j_F$, $k < k_F$, and $j - k < j_F - k_F - 2$, there also exists a convex cell $CV(C_a^i, C_b^{j'}, C_c^{k'})$ such that $j' + k'$ is equal to $j + k$ and $j' - k'$ is either $j_F - k_F - 1$ or $j_F - k_F - 2$. Since the algorithm investigates $HB_{b,c}(I_a(C_b^{j_F-1}, C_c^{k_F}))$ and $HB_{b,c}(I_a(C_b^{j_F-2}, C_c^{k_F}))$, $CV(C_a^i, C_b^j, C_c^k)$ cannot be better than the candidate cell of the algorithm.

Next, we inspect convex cells $CV(C_a^i, C_b^j, C_c^k)$ such that $i > i_F$, $j < j_F$, $k < k_F$, and $j - k > j_F - k_F + 1$. Since all intersection points of $HB_{b,c}(I_a(C_b^{j_F+1}, C_c^{k_F}))$ and C_b^j, $0 < j < j_F$ are in W_4, any convex cell $CV(C_a^i, C_b^j, C_c^k)$ such that $i > i_F$, $j < j_F$, $k < k_F$, and $j - k > j_F - k_F + 1$ cannot be better than $CV(C_a^i, C_b^j, C_c^k)$ induced by $HB_{b,c}(I_a(C_b^{j_F+1}, C_c^{k_F}))$ by Lemma 9 with a slight modification.

Theorem 2. *Given 3 input points*, AlgB *computes an optimal Steiner tree for STP-MSPBEL in constant time*.

Proof. It is obvious that *AlgB* takes constant time. We can compute the Fermat point in constant time in Step 1. In Step 2, we solve Eq (6) at most 8 times for each sub-part. Finally, in Step 3, the minimum spanning tree of three points can also be constructed in constant time.

Now, we show that the result steinerized tree is optimal for STP-MSPBEL. There are two ways to connect three terminal points with minimum Steiner points: one is to connect three points with a junction point, and the other is to connect two pairs of input points directly. Through Step 2, the algorithm presents the optimal 3-star tree among all possible 3-star trees connecting input points. By Observations 1 and 3, at least one optimal convex cell for a junction point intersects with $\Delta p_a p_b p_c$. By Lemmas 9, 10, and 11, we have shown that the result convex cell of the algorithm is better than or equal to other convex cells in $\Delta p_a p_b p_c$. Therefore, an optimal 3-star Steiner tree can be constructed by putting the junction point in the result convex cell and connecting input points with the minimum number of Steiner points. Since the algorithm computes the minimum spanning tree in Step 3, the optimal Steiner tree for STP-MSPBEL is presented by comparing those two trees.

We argue that the optimal location for the junction point may not be found although investigating all intersection points among $C_a^{i_F}$, $C_a^{i_F-1}$, $C_b^{j_F}$, $C_b^{j_F-1}$, $C_c^{k_F}$, and $C_c^{k_F-1}$ which implies that the method in [18] cannot provide the optimal Steiner tree for three points. The counterexample exists when $|p_a p_F|$ is short, and both $|p_b p_F|$ and $|p_c p_F|$ are relatively long. As shown in Fig 8, we set $R = 1$, $|p_a p_F| = 1.739$, $|p_b p_F| = 1100.95$ and $|p_c p_F| = 1000.3007$. Then, the distance between p_a and $I_a(C_b^{j_F}, C_c^{k_F-1})$ is greater than 2; however,the distance between p_a and $I_a(C_b^{j_F-1}, C_c^{k_F-2})$ is

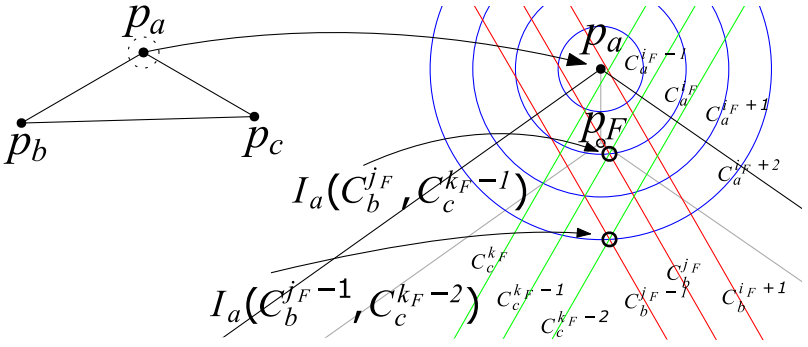

Fig 8. The location of a junction point for an optimal 3-star tree can be found outside of $C_a^{i_F+1}$ when $R = 1$, $|p_a p_F| = 1.739$, $|p_b p_F| = 1100.95$ and $|p_c p_F| = 1000.3007$. It shows that none of the intersection points between $C_a^{i_F}$, $C_a^{i_F-1}$, $C_b^{j_F}$, $C_b^{j_F-1}$, $C_c^{k_F}$, and $C_c^{k_F-1}$ might yield the optimal junction point location.

https://doi.org/10.1371/journal.pone.0294353.g008

AlgC: A 3-approximation algorithm combining the exact algorithm $AlgB$ and 3 approximation algorithm $AlgA$
Input: A set P of n terminal points, a positive constant R
Output: A Steiner tree T_C

Step 1. Set $T_C = (P, \phi)$
Compute a *minimum spanning tree* T_{MST} of P
Sort edges of T_{MST} in increasing order of lengths $e_1, e_2, \ldots, e_{n-1}$

Step 2. Compute a *Voronoi diagram* G_{VD} of P
FOREACH Voronoi vertex $v_i \in G_{VD}$
Compute an optimal Steiner tree at v_i with its relevant vertices by $AlgB$
Sort Voronoi vertices v_i of G_{VD} in increasing order of required Steiner points $v'_1, v'_2, \ldots, v'_{N_{VD}}$

Step 3. Set $t = 0, idx_e = 1, idx_v = 1$
FOR $idx_{dummy} = 1$ to $n - 1$
FOR $i = idx_e$ to $n - 1$
IF $(\lceil \frac{len(e_i)}{R} \rceil - 1) \leq t$ and two endpoints of e_i is disconnected in T_C,
THEN add steinerized edge e_i into T_C
ELSE set $t = \lceil \frac{len(e_i)}{R} \rceil - 1, idx_e = i$ and break
FOR $j = idx_v$ to N_{VD}
IF the required Steiner points of $v'_j \leq 2t$ THEN
IF all related points of v'_j are disconnected in T_C
THEN add a Steiner tree into T_C according to $AlgB$ with v'_j
ELSE $idx_v = j$ and break
Return steinerized T_C

Fig 9. *AlgC*: A combined algorithm.

https://doi.org/10.1371/journal.pone.0294353.g009

less than 4. Thus, if the junction point is located at $I_a(C_b^{j_F-1}, C_c^{k_F-2})$, it requires 2101 Stiner points which is one less than that using $I_a(C_b^{j_F}, C_c^{k_F-1})$. If the junction point is located at any intersection points among $C_a^{i_F}$, $C_a^{i_F-1}$, $C_b^{j_F}$, $C_b^{j_F-1}$, $C_c^{k_F}$, and $C_c^{k_F-1}$, it requires at least 2102 Steiner points. In addition, the steinerized minimum spanning tree needs 2102 Steiner points.

5 A combined algorithm

In this section, we develop another 3-approximation algorithm by combining the exact algorithm *AlgB* and the approximation algorithm *AlgA* to further reduce the number of required Steiner points in $O(n \log n)$ time. As described in Fig 9, *AlgC* computes the minimum spanning tree (MST) of terminal points and sorts MST edges by the number of required Steiner points in ascending order. Next, *AlgC* constructs the Voronoi diagram of the set of terminal points as sites and calculates the number of Steiner points required at each Voronoi vertex to connect its relevant 3 terminal points by *AlgB*. Then, the algorithm sorts the Voronoi vertices by the number of required Steiner points. Let $E_{MST}(i)$ be a set of MST edges requiring i Steiner points and $V_{VD}(j)$ be a set of Voronoi vertices requiring j Steiner points. By looking up the sorted lists of MST edges and Voronoi vertices, we are able to construct a steinerized spanning tree. We process an edge case or a vertex case by the following rules:

Rule 1. $E_{MST}(i)$ is processed before $V_{VD}(j)$ where $i < \lceil j/2 \rceil$.

Rule 2. $V_{VD}(j)$ is processed before $E_{MST}(i)$ where $j \leq 2i$.

In the edge case, if two endpoints are not in the same component, we steinerize the edge and add it to the resulting tree. In the vertex case, if three relevant terminal points of the Voronoi vertex are not in the same component, we add Steiner points and relevant edges into the resulting tree by *AlgB*. For example, at first, we add all MST edges in $E_{MST}(0)$ by Rule 1, and then we look at the Voronoi vertices in $V_{VD}(1)$ and $V_{VD}(2)$ by Rule 2. Iteratively, we increase the result tree size by processing with $E_{MST}(i)$, $V_{VD}(2i + 1)$, and $V_{VD}(2i + 2)$. The basic idea behind this strategy is that if three points are in different connected components after adding edges in $E_{MST}(i)$, at least $2(i + 1)$ Steiner points are required to connect those three components by MST edges. Thus, it is beneficial to use the vertex case if $V_{VD}(2i + 1)$ or $V_{VD}(2i + 2)$ connects those three points. If three points are already in the same connected component before $V_{VD}(j)$ are considered, the bottleneck of the path between any of two points is less than or equal to $\lceil j/2 \rceil - 1$. Thus, Despite removing the two bottleneck edges, the result tree deteriorates when Steiner points are inserted based on $V_{VD}(j)$.

Theorem 3. AlgC *provides the Steiner tree* T_C *satisfying* $C(T_C) \leq C(T_A)$ *in* $O(n \log n)$ *time where* T_A *is the Steiner tree produced by* AlgA.

Proof. It is straightforward that *AlgC* takes $O(n \log n)$ time. Since *AlgB* takes constant time, Step 1 and 2 of *AlgC* takes $O(n \log n)$ time. In Step 3, there are $n - 1$ MST edges and the $O(n)$ Voronoi vertices in the sorted lists. Since each connectivity query takes $O(\log n)$ time, the overall time complexity is bounded by $O(n \log n)$. The performance of *AlgC* is better than or equal to that of *AlgA*. *AlgC* first processes with $E(0)$ and $V(1)$. The resulting tree (forest) is exactly the same as the result of *AlgA* after processing Step 3. In the context of *AlgC*, Steiner points are introduced through the vertex case exclusively if the spanning tree established by *AlgB* outperforms the tree formed by steinerized edges of the minimum spanning tree. Consequently, *AlgC* produces a spanning tree that requires an equal or fewer number of Steiner points than that of *AlgA*.

6 Conclusion

We present approximation algorithms for the Steiner tree problem with the minimum number of Steiner points and bounded edge length. Previously, the best-known deterministic approximation algorithm has $O(n^3)$ running time with an approximation ratio of 3. In this paper, we propose $O(n \log n)$-time approximation algorithm with the same approximation ratio, significantly improving the time complexity. Additionally, to enhance performance, we introduce the first exact algorithm that can provide an optimal Steiner tree for any given three points in constant time. By using this exact algorithm, we develop another 3-approximation algorithm that runs in $O(n \log n)$ time with better performance in terms of the number of required Steiner points.

Author Contributions

Supervision: Sunghee Choi.

Writing – original draft: Donghoon Shin.

References

1. Lin GH, Xue G. Steiner tree problem with minimum number of Steiner points and bounded edge-length. Information Processing Letters. 1999; 69(2):53–57. https://doi.org/10.1016/S0020-0190(98)00201-4
2. Gilbert EN, Pollak HO. Steiner Minimal Trees. SIAM Journal on Applied Mathematics. 1968; 16(1):1–29. https://doi.org/10.1137/0116001

3. Cheng X, Du DZ, Wang L, Xu B. Relay sensor placement in wireless sensor networks. Wireless Networks. 2008; 14(3):347–355. https://doi.org/10.1007/s11276-006-0724-8
4. Younis M, Senturk IF, Akkaya K, Lee S, Senel F. Topology management techniques for tolerating node failures in wireless sensor networks: A survey. Computer Networks. 2013;.
5. Hartmanis J. Computers and intractability: a guide to the theory of np-completeness (michael r. garey and david s. johnson). Siam Review. 1982; 24(1):90. https://doi.org/10.1137/1024022
6. Chlebík M, Chlebíková J. The Steiner tree problem on graphs: Inapproximability results. Theoretical Computer Science. 2008; 406(3):207–214. https://doi.org/10.1016/j.tcs.2008.06.046
7. Byrka J, Grandoni F, Rothvoß T, Sanità L. Steiner tree approximation via iterative randomized rounding. Journal of the ACM (JACM). 2013; 60(1):1–33. https://doi.org/10.1145/2432622.2432628
8. Chen CY. An efficient approximation algorithm for the Steiner tree problem. In: Proceedings of the 2nd International Conference on Information Science and Systems; 2019. p. 179–184.
9. Chen CY, Hsieh SY. An improved algorithm for the Steiner tree problem with bounded edge-length. Journal of Computer and System Sciences. 2022; 123:20–36. https://doi.org/10.1016/j.jcss.2021.07.003
10. Traub V, Zenklusen R. Local search for weighted tree augmentation and Steiner tree. In: Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA). SIAM; 2022. p. 3253–3272.
11. Zhang J, Liu Z, Deng X, Yin J. Truthful Mechanisms for Steiner Tree Problems. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37; 2023. p. 5884–5891.
12. Berman P, Karpinski M, Zelikovsky A. 25-Approximation Algorithm for Steiner Tree Problem with Distances 1 and 2. In: Workshop on Algorithms and Data Structures. Springer; 2009. p. 86–97.
13. Chen D, Du DZ, Hu XD, Lin GH, Wang L, Xue G. Approximations for Steiner trees with minimum number of Steiner points. Journal of Global Optimization. 2000; 18(1):17–33. https://doi.org/10.1023/A:1008384012064
14. Senel F, Younis M. Relay node placement in structurally damaged wireless sensor networks via triangular steiner tree approximation. Computer Communications. 2011; 34(16):1932–1941.
15. Senel F, Younis M. Optimized relay node placement for establishing connectivity in sensor networks. In: Global Communications Conference (GLOBECOM), 2012 IEEE; 2012. p. 512–517.
16. Shin D. Efficient algorithms with performance guarantees for geometric problems in wireless networks; 2016. Doctoral dissertation, KAIST, http://library.kaist.ac.kr/search/detail/view.do?bibCtrlNo=648280.
17. Gueron S, Tessler R. The fermat-steiner problem. The American Mathematical Monthly. 2002; 109 (5):443–451. https://doi.org/10.1080/00029890.2002.11919871
18. Senel F, Younis M. Novel relay node placement algorithms for establishing connected topologies. Journal of Network and Computer Applications. 2016; 70:114–130. https://doi.org/10.1016/j.jnca.2016.04.025
19. Mark dB, Otfried C, Marc vK, Mark O. Computational geometry algorithms and applications. Spinger; 2008.
20. Sack JR, Urrutia J. Handbook of computational geometry. Access Online via Elsevier; 1999.
21. Hwang FK, Richards DS, Winter P. The Steiner tree problem. Elsevier; 1992.
22. Leiserson CE, Rivest RL, Stein C, Cormen TH. Introduction to algorithms. The MIT press; 2001.