The Minimum Moving Spanning Tree Problem*

Hugo A. Akitaya[†] Ahmad Biniaz[‡] Prosenjit Bose[†] Jean-Lou De Carufel[§]
Anil Maheshwari[†] Luís Fernando Schultz Xavier da Silveira[†] Michiel Smid[†]

Abstract

We investigate the problem of finding a spanning tree of a set of moving points in the plane that minimizes the maximum total weight (sum of Euclidean distances between edge endpoints) or the maximum bottleneck throughout the motion. The output is a single tree, i.e., it does not change combinatorially during the movement of the points. We call these trees the minimum moving spanning tree, and the minimum bottleneck moving spanning tree, respectively. We show that, although finding the minimum bottleneck moving spanning tree can be done in $O(n^2)$ time, it is NP-hard to compute the minimum moving spanning tree. We provide a simple $O(n^2)$-time 2-approximation and a $O(n \log n)$-time $(2+\varepsilon)$-approximation for the latter problem.

1 Introduction

The Euclidean minimum spanning tree (EMST) of a point set is the minimum weight graph that connects the given point set, where the weight of the graph is given by the sum of Euclidean distances between endpoints of edges. EMST is a classic data structure in computational geometry and it has found many uses in network design and in approximating NP-hard problems. In the visualization community, a series of methods generalize Euler diagrams to represent spatial data [8, 2, 9, 16]. These approaches represent a set by a connected colored shape containing the points in the plane that are in the given set. In order to reduce visual clutter, approaches such as Kelp Diagrams [9] and colored spanning graphs [13] try to minimize the area (or "ink") of such colored shapes. Each shape can be considered as a generalization of the EMST of points in the set.

Motivated by visualizations of time-varying spatial data, we investigate a natural generalization of the minimum spanning tree (MST) and the minimum bottleneck spanning tree (MBST) for a set of moving points. In general it is desirable that visualizations are stable, i.e., small changes in the input should produce small changes in the output [17]. In this paper, we want to maintain all points connected throughout the motion by the same tree (the tree does not change topologically during the time frame). Consider points in the plane moving on a straight line with constant speed over a time interval $[0, 1]$. The weight of an edge pq between points p and q is defined to be the Euclidean distance $\|pq\|$. Note that the weight of an edge changes over time. We define the *Minimum Moving Spanning Tree* (MMST) of a set of moving points to be a spanning tree that minimizes the maximum sum of weights of its edges during the time interval. Analogously, we

*Research supported in part by NSERC.
[†]School of Computer Science, Carleton University, Ottawa, ON, Canada.
[‡]School of Computer Science, University of Windsor, Windsor, ON, Canada.
[§]School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON, Canada.

define *Minimum Bottleneck Moving Spanning Tree* (MBMST) of a set of moving points to be a spanning tree that minimizes the maximum individual weight of edges in the tree during the time interval.

Apart from this motivation, the concepts of MMST and MBMST are relevant in the context of moving networks. Motivated by the increase in mobile data consumption, network architecture containing mobile nodes have been considered [14]. In this setting, the design of the topology of the networks is a challenge. Due to the mobility of the vertices, existing methods update the topology dynamically and the stability becomes important since there are costs associated with establishing new connections and handing over ongoing sessions. The MMST and MBMST offer stability in mobile networks.

Results and Organization. We study the problems of finding an MMST and an MBMST of a set of points moving linearly, each at constant speed. Section 2 provides formal definitions and proves that the distance function between points is convex in this setting. We use this property in an exact $O(n^2)$-time algorithm for the MBMST as shown in Section 3. Our algorithm computes the minimum bottleneck tree in a complete graph G on the moving points in which the weight of each edge is the maximum distance between the pairs of points during the time frame. In Section 4.1 we present an $O(n^2)$-time 2-approximation for MMST by computing the MST of G. In Section 4.2 we provide an example that shows our analysis for the approximation ratio is tight. In Section 4.3, we show that the MMST is equal to the minimum spanning tree of a point set in $\mathbb{R}^4$ with a non-Euclidean metric. Since this metric space has doubling dimension $O(1)$, we obtain an $O(n \log n)$-time $(2+\varepsilon)$-approximation algorithm. Finally, we show that the problem of finding the MMST is NP-hard in Section 4.4 by reducing from the Partition problem.

Related work. Examples of other visualizations of time-varying spatial data are space-time cubes [15], that represent varying 2D data points with a third dimension, and motion rugs [6, 21], that reduces the dimentionality of the movement of data points to 1D, presenting a 2D static overview visualizations. The representation of time-varying geometric sets were also the theme of a recent Dagstuhl Seminar 19192 “Visual Analytics for Sets over Time and Space” [10]. In the context of algorithms dealing with time-varying data Meulemans et al. [17] introduces a metric for stability, analysing the trade-off between quality and stability of results, and applying it to the EMST of moving points. Monma and Suri [18] study the number of topological changes that occur in the EMST when one point is allowed to move.

The problem of finding the MMST and MBMST of moving points can be seen as a bicriteria optimization problem if the points move linearly (as shown in Section 2.2). In this context, the addition of a new criterion could lead to an NP-hard problem, such as the bi-criteria shortest path problem in weighted graphs. Garey and Johnson show that given a source and target vertices, minimizing both length and weight of a path from source to target is NP-hard [11, p. 214]. Arkin et al. analyse other criteria combined with the shortest path problem [4], such as the total turn length and different norms for path length.

Maintaining the EMST and other geometric structures of a set of moving points have been investigated by several papers since 1985 [5]. Kinetic data structures have been proposed to maintain the EMST [20, 1]. Research in this area have focused on bounds on the number of combinatorial changes in the EMST and efficient algorithms. To the best of our knowledge, the problem of finding the MMST and MBMST (a single tree that does not change during the movement of points) has not been investigated.

2 Preliminaries

In this section we formally define the minimum moving spanning tree and the minimum bottleneck moving spanning tree of a set of moving points. We then prove that, for points moving linearly, the distance function between a pair of points is convex.

2.1 Definitions

A *moving point* p in the plane is a continuous function $p : [0,1] \to \mathbb{R}^2$. We assume that p moves on a straight line segment in $\mathbb{R}^2$. We say that p is at $p(t)$ at time t. We are given a set $S = \{p_1, ..., p_n\}$ of moving points in the plane. A *moving spanning tree* T of S has S as its vertex set and weight function $w_T : [0,1] \to \mathbb{R}$ defined as $w_T(t) = \sum_{pq \in T} \|p(t)q(t)\|$. Let $\mathcal{T}(S)$ denote the set of all moving spanning trees of S. Let $w(T) = \sup_t w_T(t)$ be the weight of the moving spanning tree T. A minimum moving spanning tree (MMST) of S is a moving spanning tree of S with minimum weight. In other words an MMST is in

$$\operatorname*{arg\,min}_{T \in \mathcal{T}(S)} \left(w(T) \right).$$

Let $b_T(t) = \sup_{pq \in T} \|p(t)q(t)\|$ denote the *bottleneck* of a tree T at time t. A minimum bottleneck moving spanning tree (MBMST) of S is a moving spanning tree of S that minimizes the bottleneck over all $t \in [0,1]$. In other words an MBMST is in

$$\operatorname*{arg\,min}_{T \in \mathcal{T}(S)} \left(\max_t b_T(t) \right).$$

2.2 Convexity

Let p and q be two moving points in the plane. We assume that these points move along (possibly different) lines at (possibly different) constant velocities. Thus, for any real number t, we can write the positions of p and q at time t as

$$p(t) = (a_p + u_p t, b_p + v_p t)$$

and

$$q(t) = (a_q + u_q t, b_q + v_q t),$$

where a_p, u_p, b_p, v_p are constants associated with the point p. At time $t = 0$, p is at (a_p, b_p), and the velocity vector of p is (u_p, v_p). Let $d(t) = \|p(t)q(t)\|$ denote the Euclidean distance between p and q at time t. In the next lemma we prove that d is a convex function. The convexity of d is also implied by a result of Alt and Godau [3] that the free space diagram of any two line segments is convex.

Lemma 1. *The function d is convex.*

Proof. It suffices to prove that the second derivative of d is non-negative for all real numbers t. We can write

$$d(t) = \sqrt{At^2 + Bt + C},$$

where A, B, and C depend only on $a_p, u_p, b_p, v_p, a_q, u_q, b_q$, and v_q. Observe that $A \geq 0$. Since $d(t)$ represents a distance, $At^2 + Bt + C \geq 0$ for all t in $\mathbb{R}$. It follows that the discriminant of this quadratic function is non-positive, i.e.,

$$B^2 - 4AC \leq 0. \tag{1}$$

Let $\alpha = -B/2A$ and $\beta = C/A - B^2/(4A^2)$. Then

$$d(t) = \sqrt{A} \cdot \sqrt{(t-\alpha)^2 + \beta}.$$

The second derivative of the function $f(t) = \sqrt{t^2 + \beta}$ is given by

$$f''(t) = \frac{\beta}{(t^2 + \beta)^{3/2}}.$$

It follows from (1) that $\beta \geq 0$. Thus, $f''(t) \geq 0$ for all t in $\mathbb{R}$. Since $d(t) = \sqrt{A} \cdot f(t - \alpha)$, we have $d''(t) \geq 0$ for all t in $\mathbb{R}$, and in particular, for $t \in [0, 1]$. □

The convexity of the distance function between two moving points (Lemma 1) implies the following corollary.

Corollary 2. *The largest distance between two moving points is attained either at the start time or at the finish time.*

Let S be a set of n moving points in the plane. For two points p and q in S, we denote by $\|p(0)q(0)\|$ and $\|p(1)q(1)\|$ the distances between p and q at times $t = 0$ and $t = 1$, respectively. Moreover, we denote by $|pq|_{\max}$ the largest distance between p and q during time interval $[0, 1]$. By Corollary 2 we have

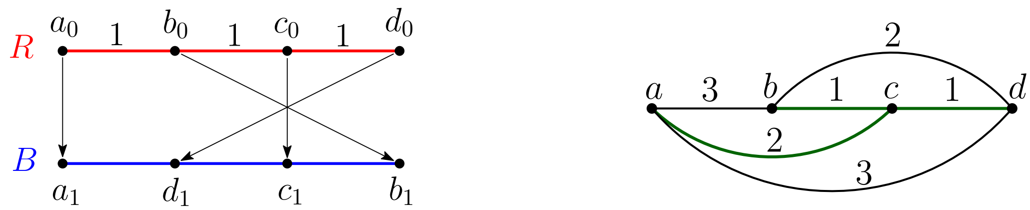
$$|pq|_{\max} = \max\{\|p(0)q(0)\|, \|p(1)q(1)\|\}. \tag{2}$$

3 Minimum bottleneck moving spanning tree

Since by Corollary 2 the largest length of an edge is attained either at time 0 or at time 1, it might be tempting to think that the MBMST of S is also attained at times 0 or 1. However the example in Figure 1(a) shows that this may not be true. In this example we have four points a, b, c, and d that move from time 0 to time 1 as depicted in the figure. The MBST of these points at time 0 is the red tree R, and their MBST at time 1 is the blue tree B. Recall that $b_T(t)$ is the bottleneck of tree T at time t. Let $b(T) = \max_t b_T(t)$ be the *bottleneck* of T. In R the weight of ab at time 0 is 1 while its weight at time 1 is 3, and thus $b(R) = 3$. In B the weight of ad at time 1 is 1 while its weigh at time 0 is 3, and thus $b(B) = 3$. However, for this point set the tree $T = \{ac, cb, cd\}$ has bottleneck 2.

Although the above example shows that the computation of the MBMST is not straightforward, we present a simple algorithm for finding the MBMST. Let G be the complete graph on points of S where the weight $w(pq)$ of every edge pq is the largest distance between p and q during time interval $[0, 1]$, that is, $w(pq) = |pq|_{\max}$; see Figure 1(b).

Lemma 3. *The bottleneck of an MBMST of S is not smaller than the bottleneck of an MBST of G.*



(a) (b)

Figure 1: Four points that move from time 0 to time 1. (a) R is the MBST at time 0, and B is the MBST at time 1. (b) The graph G; green edges form an MBST of this graph.

Proof. Our proof is by contradiction. Let T^* be an MBMST of S and let T be an MBST of G. For the sake of contradiction assume that $b(T^*) < b(T)$, where we abuse the notation for simplicity making $b(T) = \max_{pq \in T} w(pq)$ the bottleneck of T. Let pq be a bottleneck edge of T, that is $b(T) = w(pq)$. Denote by T_p and T_q the two subtrees obtained by removing pq from T, and denote by V_p and V_q the vertex sets of these subtrees. Since the vertex set of T is the same as that of T^*, there is an edge, say rs, in T^* that connects a vertex of V_p to a vertex of V_q. Since the bottleneck of T^* is its largest edge-length in time interval $[0, 1]$, we have that $|rs|_{\max} \leqslant b(T^*)$. Since in G we have $w(rs) = |rs|_{\max}$, the following inequality is valid: $w(rs) = |rs|_{\max} \leqslant b(T^*) < b(T) = w(pq)$. Let T' be the spanning tree of G that is obtained by connecting T_p and T_q by rs. Then $b(T') \leqslant b(T^*)$. If we repeat this process for all bottleneck edges of T, then we obtain a tree T' whose bottleneck is strictly smaller than that of T. This contradicts the fact that T is an MBST of G. $\square$

It is implied from Lemma 3 that any MBST of G is an MBMST of S. Since an MBST of a graph can be computed in time linear in the size of the graph [7], an MBST of G can be computed in $O(n^2)$ time. The following theorem summarizes our result in this section.

Theorem 4. *A minimum bottleneck moving spanning tree of n moving points in the plane can be computed in $O(n^2)$ time.*

4 Minimum moving spanning tree

In this section we study the problem of computing an MMST of moving points. At the end of this section we prove that this problem is NP-hard. We start by proposing a 2-approximation algorithm for the MST problem. In Section 4.2 we show that our analysis of the approximation ratio is tight.

4.1 A 2-approximation algorithm

Our algorithm is very simple and just computes a MST of the graph G that is constructed in Section 3.

Lemma 5. *The weight of any MST of G is at most two times the weight of any MMST of S.*

Proof. Let T be any MST of G and let T^* be any MMST of S. Let $w(T^*) = \sup_t w_T(t)$ be the weight of the moving spanning tree T^*. We abuse the notation for simplicity making $w(T) = \sum_{pq \in T} w(pq)$ the weight of the spanning tree T. We are going to show that $w(T) \leqslant 2 \cdot w(T^*)$. Let T' be a tree that is combinatorially equivalent to T^*, i.e., has the same topology as T^*. Assign to each edge pq

of T' the weight $w(pq) = |pq|_{\max}$. After this weight assignment, T' is a spanning tree of G. Since T is a MST of G, we have $w(T) \leqslant w(T')$.

By Corollary 2 the largest distance between two points is achieved either at time 0 or at time 1. Let E_0^* be the set of edges of T^* whose endpoints largest distance is achieved at time 0. Define E_1^* analogously. Then $w(E_0^*) \leqslant w(T^*)$ and $w(E_1^*) \leqslant w(T^*)$. Moreover, $w(T') = w(E_0^*) + w(E_1^*)$. By combining these inequalities we get

$$w(T) \leqslant w(T') = w(E_0^*) + w(E_1^*) \leqslant w(T^*) + w(T^*) = 2 \cdot w(T^*).$$

$\square$

A minimum spanning tree of G can be computed in $O(n^2)$ time using Prim's MST algorithm. The following theorem summarizes our result in this section.

Theorem 6. *There is an $O(n^2)$-time 2-approximation algorithm for computing the minimum moving spanning tree of n moving points in the plane.*

4.2 The approximation factor 2 is tight

In this section, we build a set of moving points showing that the approximation factor of our 2-approximation algorithm can be arbitrarily close to 2.

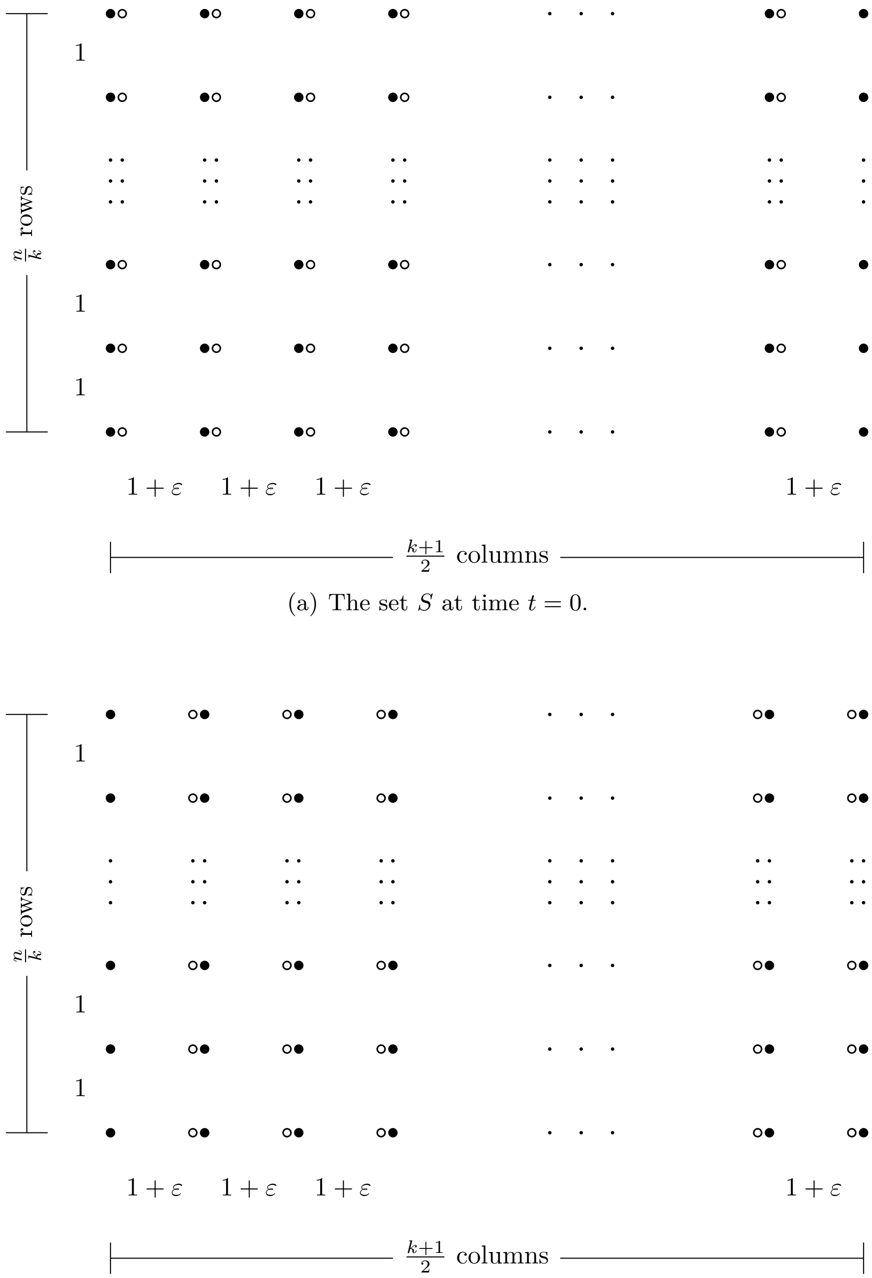
Let $\varepsilon > 0$ be a real number and n be a positive integer. Consider the following point set S containing n points. For all $0 \leq i \leq \frac{k-1}{2}$ and all $0 \leq j \leq \frac{n}{k} - 1$, there is an immobile point $p_{i,j} = (i + i\varepsilon, j) \in S$ (in Figures 2(a) and 2(b), they correspond to the small solid disks). At time $t = 0$, for all $0 \leq i \leq \frac{k-3}{2}$ and all $0 \leq j \leq \frac{n}{k} - 1$, there is a point $p'_{i,j} = (i + i\varepsilon, j) \in S$ (in Figures 2(a) and 2(b), they correspond to the small circles). All the points $p'_{i,j}$ move at constant speed from $(i + i\varepsilon, j)$ at time $t = 0$ to $(i + 1 + (i + 1)\varepsilon, j)$ at time $t = 1$.

We now describe the moving spanning tree T produced by our 2-approximation algorithm on S. For all $0 \leq i \leq \frac{k-1}{2}$ and all $0 \leq j \leq \frac{n}{k} - 2$ the distance between $p_{i,j}$ and $p_{i,j+1}$ is 1 at all time. Therefore, in $G_{\max}$, the edge $\{p_{i,j}, p_{i,j+1}\}$ has length 1. For all $0 \leq i \leq \frac{k-3}{2}$ and all $0 \leq j \leq \frac{n}{k} - 2$ the distance between $p'_{i,j}$ and $p'_{i,j+1}$ is 1 at all time. Therefore, in $G_{\max}$, the edge $\{p'_{i,j}, p'_{i,j+1}\}$ has length 1.

For all $0 \leq i \leq \frac{k-3}{2}$ and all $0 \leq j \leq \frac{n}{k} - 1$ the distance between $p_{i,j}$ and $p_{i+1,j}$ is $1 + \varepsilon$ at all time. Therefore, in $G_{\max}$, the edge $\{p_{i,j}, p_{i+1,j}\}$ has length $1 + \varepsilon$. For all $0 \leq i \leq \frac{k-3}{2}$ and all $0 \leq j \leq \frac{n}{k} - 1$, since $p'_{i,j}$ is moving, the edge $\{p_{i,j}, p'_{i,j}\}$ has length $1 + \varepsilon$ in $G_{\max}$. For all $0 \leq i \leq \frac{k-3}{2}$ and all $0 \leq j \leq \frac{n}{k} - 1$, since $p'_{i,j}$ is moving, the edge $\{p'_{i,j}, p_{i+1,j}\}$ has length $1 + \varepsilon$ in $G_{\max}$. For all $0 \leq i \leq \frac{k-5}{2}$ and all $0 \leq j \leq \frac{n}{k} - 1$, the distance between $p'_{i,j}$ and $p'_{i+1,j}$ is $1 + \varepsilon$ at all time. Therefore, in $G_{\max}$, the edge $\{p'_{i,j}, p'_{i+1,j}\}$ has length $1 + \varepsilon$.

All other edges in $G_{\max}$ have length strictly larger than $1 + \varepsilon$. Hence, if we run Kruskal's algorithm to compute the MST of $G_{\max}$, we first get the equivalent of $\frac{k+1}{2}$ vertical line segments of length $\frac{n}{k} - 1$ that connect the $\frac{k+1}{2}$ columns of immobile points. We also get the equivalent of $\frac{k-1}{2}$ vertical line segments of length $\frac{n}{k} - 1$ that connect the $\frac{k-1}{2}$ columns of moving points. Hence, we have a total of $\frac{k+1}{2} + \frac{k-1}{2} = k$ vertical line segments of length $\frac{n}{k} - 1$. Then, Kruskal's algorithm adds the equivalent of $\frac{k-1}{2}$ horizontal line segments of length $1 + \varepsilon$ which connect the k vertical line segments. As a result, we have a tree which spans $G_{\max}$. Hence, the weight of T is

$$k\left(\frac{n}{k} - 1\right) + \frac{k-1}{2}(1 + \varepsilon) = \frac{2n - k - 1}{2} + \frac{k-1}{2}\varepsilon. \tag{3}$$



(a) The set S at time $t = 0$.

(b) The set S at time $t = 1$.

Figure 2: Our 2-approximation algorithm has an approximation factor that is arbitrarily close to 2 on the point set S. The small solid disks are the points $p_{i,j}$ and the small circles are the points $p'_{i,j}$.

We now define another moving spanning tree T' of S. Take $\frac{n}{k}$ horizontal line segments of length $\frac{k-1}{2}(1+\varepsilon)$ that connect each row of points. Then, take one vertical line segment of length $\frac{n}{k}-1$ that connects all points within one single column (all points $p_{i,j}$ for a fixed i and all $0 \leq j \leq \frac{n}{k}-1$). We obtain a tree which connects all points of S at all time and whose total length is

$$\frac{n}{k}\frac{k-1}{2}(1+\varepsilon)+\frac{n}{k}-1=\frac{(k+1)n-2k}{2k}+\frac{(k-1)n}{2k}\varepsilon. \tag{4}$$

Since the cost of the optimal solution is at most (4), the approximation factor is at least the ratio between (3) and (4):

$$\frac{k(2n-k-1)+k(k-1)\varepsilon}{(k+1)n-2k+(k-1)n\varepsilon}.$$

By taking $k=\sqrt{n}$, we get

$$\frac{2\sqrt{n}+1+\varepsilon}{(1+\varepsilon)\sqrt{n}+2} \xrightarrow{n\to\infty} \frac{2}{1+\varepsilon}.$$

Therefore, by taking n large enough and ε sufficiently small, we get a point set on which our 2-approximation algorithm has an approximation ratio that is arbitrarily close to 2.

4.3 An $O(n \log n)$-time $(2+\varepsilon)$-approximation algorithm

Section 4.1 showed that the weight of the minimum spanning tree of the graph G, defined in Section 3, gives a 2-approximation to the MMST. Since G has $\Theta(n^2)$ edges, it takes $\Theta(n^2)$ time to compute its MST. In this section, we prove that a $(1+\varepsilon)$-approximation to the minimum spanning tree of G can be computed in $O(n \log n)$ expected time. Thus, if we replace ε by $\varepsilon/2$, we obtain a $(2+\varepsilon)$-approximation to computing the MMST of a set of linearly moving points S.

For any point p in S, define the point

$$P=(p(0),p(1))$$

in $\mathbb{R}^4$. Doing this for all points in S, we obtain a set S' of n points in $\mathbb{R}^4$. For any two points P and Q in S', define their distance to be

$$\operatorname{dist}(P,Q)=\max(\|p(0)q(0)\|,\|p(1)q(1)\|).$$

Since $\operatorname{dist}(P,Q)=w(pq)$, the minimum spanning tree of our graph G has the same weight as the minimum spanning tree (under dist) of the point set S'.

Lemma 8 below states that dist satisfies the properties of a metric. Its proof uses the following lemma, which is probably well known.

Lemma 7. *Let V be an arbitrary set and let $d_1 : V \times V \to \mathbb{R}$ and $d_2 : V \times V \to \mathbb{R}$ be two functions, such that both (V,d_1) and (V,d_2) are metric spaces. Define the function $d : V \times V \to \mathbb{R}$ by*

$$d(a,b)=\max(d_1(a,b),d_2(a,b))$$

for all a and b in V. Then (V,d) is a metric space.

Proof. It is clear that, for all a and b in V, $d(a,a) = 0$, $d(a,b) > 0$ if $a \neq b$, and $d(a,b) = d(b,a)$. It remains to prove that the triangle inequality holds.

Let a, b, and c be elements of V. Then

$$\begin{aligned} d(a,b) &= \max(d_1(a,b), d_2(a,b)) \\ &\leq \max(d_1(a,c) + d_1(c,b), d_2(a,c) + d_2(c,b)). \end{aligned}$$

Using the inequality

$$\max(\alpha + \beta, \gamma + \delta) \leq \max(\alpha, \gamma) + \max(\beta, \delta),$$

it follows that

$$\begin{aligned} d(a,b) &\leq \max(d_1(a,c), d_2(a,c)) + \max(d_1(c,b), d_2(c,b)) \\ &= d(a,c) + d(c,b). \end{aligned}$$

□

Lemma 8. *The pair (S', dist) is a metric space.*

Proof. The proof follows from Lemma 7 and the definition of dist. □

The next lemma states that the metric space (S', dist) has bounded doubling dimension. We recall the definition. For any point P in S' and any real number $\rho > 0$, the ball with center P and radius ρ is the set

$$\text{ball}^{\text{dist}}(P, \rho) = \{Q \in S' : \text{dist}(P, Q) \leq \rho\}.$$

Let λ be the smallest integer such that for every real number $\rho > 0$, every ball of radius ρ can be covered by at most λ balls of radius $\rho/2$. The doubling dimension of (S', dist) is defined to be $\log \lambda$.

Lemma 9. *The doubling dimension of the metric space (S', dist) is $O(1)$.*

Proof. Recall that S' is a set of points in $\mathbb{R}^4$. We denote the Euclidean distance between two points P and Q of S' by $\|PQ\|$. The Euclidean ball with center P and radius ρ is denoted by $\text{ball}^e(P, \rho)$. Thus,

$$\text{ball}^e(P, \rho) = \{Q \in S' : |PQ| \leq \rho\}.$$

It is easy to verify that

$$\text{dist}(P, Q) \leq \|PQ\| \leq \sqrt{2} \cdot \text{dist}(P, Q). \tag{5}$$

Let P be a point in S', let $\rho > 0$ be a real number, and let $B^{\text{dist}} = \text{ball}^{\text{dist}}(P, \rho)$. We will prove that B^{dist} can be covered by $O(1)$ balls of radius $\rho/2$.

Let B^e be the Euclidean ball with center P and radius $\rho \cdot \sqrt{2}$. It follows from (5) that

$$B^{\text{dist}} \subseteq B^e.$$

It is well known that the doubling dimension of the Euclidean space $\mathbb{R}^4$ is bounded by a constant. Thus, by applying the definition of doubling dimension twice, we can cover B^e by $k = O(1)$ Euclidean balls $B_1^e, \ldots, B_k^e$ balls, each of radius $\rho \cdot \sqrt{2}/4 \leq \rho/2$. Let these balls have centers $C_1, \ldots, C_k$. For each $i = 1, \ldots, k$, define $B_i^{\text{dist}} = \text{ball}^{\text{dist}}(C_i, \rho/2)$. It follows from (5) that

$$B_i^e \subseteq B_i^{\text{dist}}.$$

Thus,

$$B^{\text{dist}} \subseteq B^e \subseteq \bigcup_{i=1}^{k} B_i^e \subseteq \bigcup_{i=1}^{k} B_i^{\text{dist}},$$

i.e., we have covered the ball B^{dist} by $k = O(1)$ balls of radius $\rho/2$. ☐

Lemma 10. *Let $\varepsilon > 0$ be a constant. In $O(n \log n)$ expected time, we can compute a $(1+\varepsilon)$-approximation to the minimum spanning tree of the metric space (S', dist).*

Proof. As (S', dist) has a constant doubling dimension (by Lemma 9), a result of Har-Peled and Mendel [12] implies that a $(1+\varepsilon)$-spanner of (S', dist) with $O(n)$ edges can be computed in $O(n \log n)$ expected time. Their algorithm assumes that any distance in the metric space can be computed in $O(1)$ time; this is the case for our distance function dist.

It is known that a minimum spanning tree of a $(1+\varepsilon)$-spanner is a $(1+\varepsilon)$-approximation to the minimum spanning tree. (See, e.g., [19, Theorem 1.3.1].)

Since the spanner has $O(n)$ edges, its minimum spanning tree can be computed in $O(n \log n)$ time using Prim's MST algorithm combined with a binary min-heap. ☐

As a consequence of Lemma 10 and the fact that $\text{dist}(P, Q) = w(pq)$, we have the following theorem.

Theorem 11. *In $O(n \log n)$ expected time, we can compute a $(2+\varepsilon)$-approximation for the minimum moving spanning tree of a set of linearly moving points in the plane.*

4.4 NP-hardness of MMST

Inspired by Arkin et. al. [4], we reduce the Partition problem, which is known to be NP-hard [11], to the MMST problem. In one formulation of the Partition problem, we are given $n > 0$ positive integers $a_0, \ldots, a_{n-1}$ and must decide whether there is a subset $S \subseteq \{0, \ldots, n-1\}$ such that

$$\sum_{i \in S} a_i = \frac{1}{2} \sum_{i=0}^{n-1} a_i.$$

Construction. We construct an instance of a decision version of the MMST problem defined as follows. First we let $\ell = \max\{a_0, \ldots, a_{n-1}\}$ and then, for each $i \in \{0, \ldots, n-1\}$, we put the following points into our set P of moving points (Figure 3):

- A_i, stationary at $(i\ell, 0)$;
- B_i, stationary at $(i\ell, \ell)$;
- C_i, moving from $(i\ell, \ell)$ to $(i\ell, \ell + a_i)$;
- D_i, stationary at $(i\ell, \ell + a_i)$; and
- E_i, moving from $(i\ell, \ell + a_i)$ to $(i\ell, \ell)$.

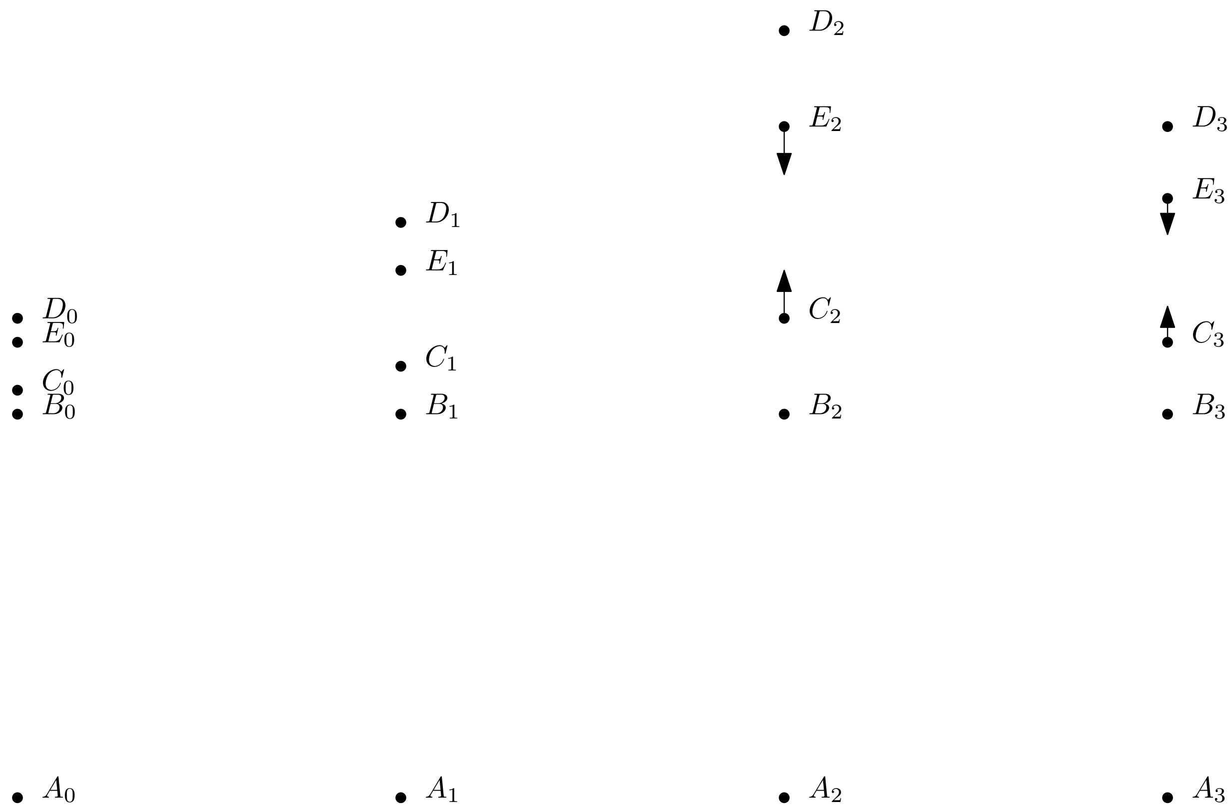


Figure 3: The positions of the points in P at time $t = 1/4$ when $n = 4$ and $(a_0, a_1, a_2, a_3) = (1, 2, 4, 3)$. The velocities of C_2, E_2, C_3 and E_3 are depicted.

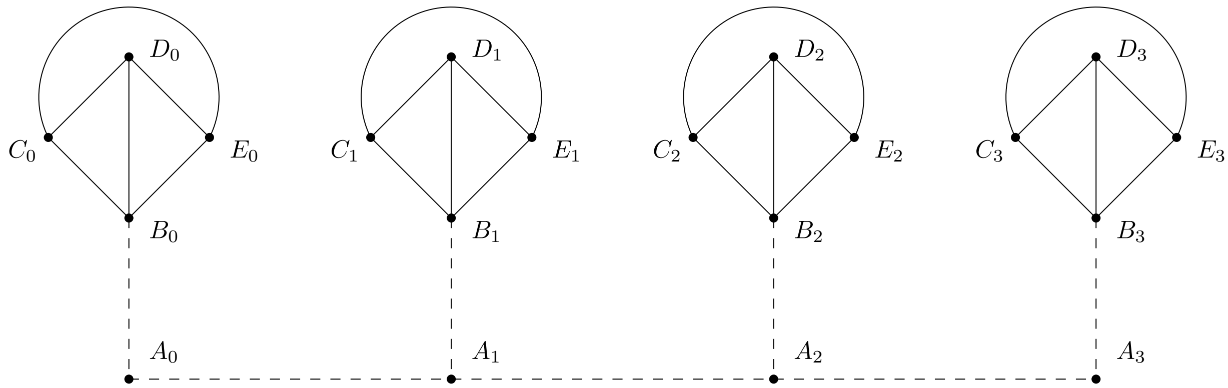


Figure 4: The (topological) edges in K_0 (dashed) and in $K_1 \setminus K_0$ (solid).

We then ask whether there is a moving spanning tree T with

$$w(T) \leq (2n-1)\ell + \frac{3}{2}\sum_{i=0}^{n-1} a_i.$$

Theorem 12. *The decision version of the MMST problem is weakly NP-hard.*

Proof. Let T be a moving spanning tree on vertex set P. Recall that $w_T(t)$ denotes the weight of T at time t. By Lemma 1, w_T is a convex function and the weight of T is indeed $w(T) = \max\{w_T(0), w_T(1)\}$.

Let K_0 be the set of edges A_iB_i for $i \in \{0, \ldots, n-1\}$ and A_iA_{i+1} for $i \in \{0, \ldots, n-2\}$ and let K_1 be the set of edges among B_i, C_i, D_i and E_i for each $i \in \{0, \ldots, n-1\}$ together with K_0 (Figure 4). We claim that there is a moving spanning tree T^* of minimum cost, i.e., an optimal solution to the MMST problem, whose edges are all in K_1. Assume the contrary for contradiction. Let T be an MMST whose intersection with K_1 is maximum. By assumption, T has at least an edge $e \notin K_1$. We now consider the two components obtained from deleting e from T. There must be at least one edge $e' \in K_1$ between the two components, since K_1 spans P. However, at any point in time, every edge in K_1 weights at most ℓ while every edge outside of K_1 weights at least ℓ, so if we bridge the two components with e', we will be left with a spanning tree T' with $w(T') \leq w(T)$ and with a larger intersection with K_1, contradicting the definition of T.

As every edge in K_0 is a bridge in the graph (P, K_1), the spanning tree T^* must contain K_0, so T^* consists of K_0 and, for each $i \in \{0, \ldots, n-1\}$, of a subtree T_i spanning $\{B_i, C_i, D_i, E_i\}$. The weights $w_{T_i}(0)$ and $w_{T_i}(1)$ must both be a multiple of a_i since so are the Euclidean distances between the vertices of T_i at these two times. There are two notable ways to build T_i: one is $T_i = \{B_iC_i, C_iD_i, D_iE_i\}$, which satisfies $w_{T_i}(0) = a_i$ and $w_{T_i}(1) = 2a_i$ and is thus called the $(1,2)$-tree; and the other is $T_i = \{B_iE_i, E_iD_i, D_iC_i\}$, which satisfies $w_{T_i}(0) = 2a_i$ and $w_{T_i}(1) = a_i$ and is thus called the $(2,1)$-tree.

We shall show that the $(1,2)$-tree or the $(2,1)$-tree have minimum weight among all moving spanning trees of $\{B_i, C_i, D_i, E_i\}$. Indeed, T_i is made of three edges and, since there are no three edges with weight zero at time 0, as can be seen in Figure 5, we must have $w_{T_i}(0) \geq a_i$ and, similarly, $w_{T_i}(1) \geq a_i$. Furthermore, each edge between B_i, C_i, D_i and E_i adds up to at least a_i in terms of their weight at time 0 and at time 1. Therefore, $w_{T_i}(0) + w_{T_i}(1) \geq 3a_i$, so either

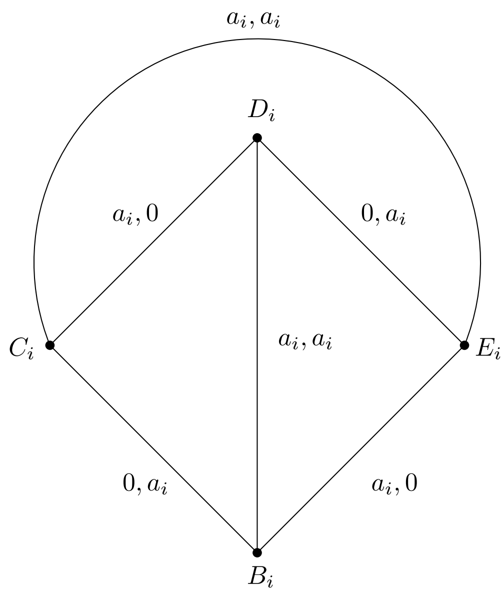


Figure 5: Edges between B_i, C_i, D_i and E_i labeled with their weights at times 0 and 1.

$w_{T_i}(0) \geq 2a_i$, , or $w_{T_i}(1) \geq 2a_i$. As a result, we may assume, without loss of generality, that T_i is either the $(1, 2)$-tree or the $(2, 1)$-tree.

Let now $S^* \subseteq \{0, \ldots, n-1\}$ be the set of indices i such that T_i is the corresponding $(2, 1)$-tree. As $|K_0| = 2n - 1$, we have

$$w_{T^*}(0) = (2n-1)\ell + \sum_{i=0}^{n-1} a_i + \sum_{i \in S^*} a_i,$$

while

$$w_{T^*}(1) = (2n-1)\ell + \sum_{i=0}^{n-1} a_i + \sum_{i \in \{0,\ldots,n-1\} \setminus S^*} a_i.$$

Therefore, the cost of T^* is

$$(2n-1)\ell + \sum_{i=0}^{n-1} a_i + \max\left\{\sum_{i \in S^*} a_i, \sum_{i \in \{0,\ldots,n-1\} \setminus S^*} a_i\right\}.$$

Because

$$\sum_{i \in S^*} a_i \geq \frac{1}{2}\sum_{i=0}^{n-1} a_i \qquad \text{or} \qquad \sum_{i \in \{0,\ldots,n-1\} \setminus S^*} a_i \geq \frac{1}{2}\sum_{i=0}^{n-1} a_i,$$

then the following holds

$$w(T^*) \geq (2n-1)\ell + \frac{3}{2}\sum_{i=0}^{n-1} a_i. \tag{6}$$

We claim that (6) holds with equality if and only if our instance of the Partition problem has a solution, i.e., there is a set $S \subseteq \{0, \ldots, n-1\}$ such that the sum of a_i for $i \in S$ is half of $a_0 + \cdots + a_{n-1}$.

Indeed, if the equality holds, we can simply let $S = S^*$. To show the converse, we build a tree T from the solution S of the Partition problem. This tree contains K_0, the corresponding $(2,1)$-trees for i in S and the corresponding $(1,2)$-trees for $i \in \{0, \ldots, n-1\} \setminus S$, resulting in a weight of

$$w(T) = (2n-1)\ell + \frac{3}{2}\sum_{i=0}^{n-1} a_i.$$

Because T^* is an MMST, $w(T^*) \leq w(T)$, so the equality holds. □

Acknowledgements

This research was carried out at the *Eighth Annual Workshop on Geometry and Graphs*, held at the Bellairs Research Institute in Barbados, January 31 – February 7, 2020. The authors are grateful to the organizers and to the participants of this workshop. We thank Günther Rote for pointing us to the work of Arkin et. al. [4].

References

[1] M. A. Abam, Z. Rahmati, and A. Zarei. Kinetic pie delaunay graph and its applications. In *Scandinavian Workshop on Algorithm Theory*, pages 48–58. Springer, 2012.

[2] B. Alper, N. Riche, G. Ramos, and M. Czerwinski. Design study of linesets, a novel set visualization technique. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2259–2267, 2011.

[3] H. Alt and M. Godau. Computing the Fréchet distance between two polygonal curves. *Int. J. Comput. Geom. Appl.*, 5:75–91, 1995.

[4] E. M. Arkin, J. S. Mitchell, and C. D. Piatko. Bicriteria shortest path problems in the plane. In *Proc. 3rd Canad. Conf. Comput. Geom*, pages 153–156, 1991.

[5] M. J. Atallah. Some dynamic computational geometry problems. *Computers & Mathematics with Applications*, 11(12):1171 – 1181, 1985.

[6] J. Buchmüller, D. Jäckle, E. Cakmak, U. Brandes, and D. A. Keim. Motionrugs: Visualizing collective trends in space and time. *IEEE transactions on visualization and computer graphics*, 25(1):76–86, 2018.

[7] P. M. Camerini. The min-max spanning tree problem and some extensions. *Information Processing Letters*, 7(1):10–14, 1978.

[8] C. Collins, G. Penn, and S. Carpendale. Bubble sets: Revealing set relations with isocontours over existing visualizations. *IEEE Trans. on Visualization and Computer Graphics (Proc. of the IEEE Conf. on Information Visualization)*, 15(6):1009 – 1016, 2009.

[9] K. Dinkla, M. J. van Kreveld, B. Speckmann, and M. A. Westenberg. Kelp diagrams: Point set membership visualization. *Comput. Graph. Forum*, 31(3pt1):875–884, June 2012.

[10] S. I. Fabrikant, S. Miksch, and A. Wolff. Visual Analytics for Sets over Time and Space (Dagstuhl Seminar 19192). *Dagstuhl Reports*, 9(5):31–57, 2019.

[11] M. R. Garey and D. S. Johnson. *Computers and intractability*, volume 174. freeman San Francisco, 1979.

[12] S. Har-Peled and M. Mendel. Fast construction of nets in low-dimensional metrics and their applications. *SIAM Journal on Computing*, 35(5):1148–1184, 2006.

[13] F. Hurtado, M. Korman, M. [van Kreveld], M. Löffler, V. Sacristán, A. Shioura, R. I. Silveira, B. Speckmann, and T. Tokuyama. Colored spanning graphs for set visualization. *Computational Geometry*, 68:262 – 276, 2018. Special Issue in Memory of Ferran Hurtado.

[14] S. Jaffry, R. Hussain, X. Gui, and S. F. Hasan. A comprehensive survey on moving networks. *arXiv preprint arXiv:2003.09979*, 2020.

[15] M.-J. Kraak. The space-time cube revisited from a geovisualization perspective. In *Proc. 21st International Cartographic Conference*, pages 1988–1996, 2003.

[16] W. Meulemans, N. H. Riche, B. Speckmann, B. Alper, and T. Dwyer. Kelpfusion: A hybrid set visualization technique. *IEEE Transactions on Visualization and Computer Graphics*, 19(11):1846–1858, 2013.

[17] W. Meulemans, B. Speckmann, K. Verbeek, and J. Wulms. A framework for algorithm stability and its application to kinetic euclidean msts. In *Latin American Symposium on Theoretical Informatics*, pages 805–819. Springer, 2018.

[18] C. L. Monma and S. Suri. Transitions in geometric minimum spanning trees. *Discret. Comput. Geom.*, 8:265–293, 1992.

[19] G. Narasimhan and M. Smid. *Geometric Spanner Networks.* Cambridge University Press, Cambridge, UK, 2007.

[20] Z. Rahmati and A. Zarei. Kinetic Euclidean minimum spanning tree in the plane. *Journal of Discrete Algorithms*, 16:2 – 11, 2012. Selected papers from the 22nd International Workshop on Combinatorial Algorithms (IWOCA 2011).

[21] J. Wulms, J. Buchmüller, W. Meulemans, K. Verbeek, and B. Speckmann. Spatially and temporally coherent visual summaries. *arXiv preprint arXiv:1912.00719*, 2019.