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 n moving points in $\mathbb{R}^{\dim}$ that minimizes the maximum total weight (under any convex distance function) 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 a minimum moving spanning tree, and a 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 when dim is a constant, it is NP-hard to compute the minimum moving spanning tree even for $\dim = 2$. We provide a simple $O(n^2)$-time 2-approximation and a $O(n \log n)$-time $(2 + \varepsilon)$-approximation for the latter problem, for any constant dim and any constant $\varepsilon > 0$.

1 Introduction

A Euclidean minimum spanning tree (EMST) of a point set in the Euclidean plane is a 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. Euclidean minimum spanning tree is a classic tool in computational geometry and it has found many uses in network design and in approximating NP-hard problems.

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 [27]. In this paper, we consider a set of moving points in $\mathbb{R}^{\dim}$ and we want to maintain all points connected throughout the motion by the same tree (the tree does not change topologically during the time frame). We consider the case when each point moves at constant speed along a straight line over the time interval $[0, 1]$. The weight of an edge pq between points p and q is defined by a convex distance function. Note that the weight of an edge changes over time. We define a *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 define a *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.

*Research supported in part by NSERC. A preliminary version of this paper has been published in the proceedings of the 17th Algorithms and Data Structures Symposium (WADS 2021) [3].

†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.

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 [23]. 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 between any two moving points is maximized at time $t = 0$ or $t = 1$. We use this property in an exact $O(n^2)$-time algorithm for the MBMST as shown in Section 3. Our algorithm computes a minimum bottleneck tree in a complete graph G_S 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 show that the problem of finding an MMST is NP-hard; this is shown by a reduction from the Partition problem. In Section 4.2, we present an $O(n^2)$-time 2-approximation for MMST by computing an MST of G_S. We also show that our analysis for the approximation ratio is tight. In Section 4.5, we improve the running time to $O(n \log n)$, but the approximation ratio becomes $2 + \varepsilon$, for any constant $\varepsilon > 0$. All our results hold for any convex distance function and any constant dimension dim.

Since the first version of our result appeared at WADS 2021, Wang and Zhao [32] improved the running time for the MBMST problem to $O(n^{4/3} \log^3 n)$ for the special case when the points are moving in $\mathbb{R}^2$ under the Euclidean distance metric.

Related work. In the visualization community, a series of methods generalize Euler diagrams to represent spatial data [13, 4, 15, 26]. 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 [15] and colored spanning graphs [22] 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. Examples of visualizations of time-varying spatial data are space-time cubes [25], that represent varying 2D data points with a third dimension, and motion rugs [9, 33], 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" [17]. In the context of algorithms dealing with time-varying data Meulemans et al. [27] introduce 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 [29] study the number of topological changes that occur in the EMST when one point is allowed to move.

The problem of finding an MMST and MBMST of moving points can be seen as a bicriteria optimization problem if the points move linearly (as shown in Section 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 [19, p. 214]. Arkin et al. analyse other criteria combined with the shortest path problem [6], 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 [7]. Kinetic data structures have been proposed to maintain the EMST [1, 31] and minimum Steiner tree [34]. The MMST and MBMST problems also lie under a broader context of MST in parametric or time-varying graphs where the edge weights vary with time. In the parametric minimum spanning tree problem we are given a graph (with n vertices and m edges) with edge weights that are linear functions of a parameter λ and we want to compute the sequence of minimum spanning trees generated as λ varies. This problem has a rich background; see e.g. [2, 11, 16, 18, 20]. Research in this area has focused on bounds on the number of combinatorial changes or transitions in the MST and on computing the updated MST. Agarwal et al. [2] have given data structures to maintain the MST over time, with a cost of $O(n^{2/3}\text{polylog } n)$ per change. Eppstein [16] has shown that the number of different minimum spanning trees obtained as λ varies can be $\Omega(m \log n)$. Chan [11] has given a randomized $O(n(m/n)^{\varepsilon} \log n + m)$ expected time algorithm for any fixed $\varepsilon > 0$ that finds the time value at which the weight of the largest MST edge is minimized.

Perhaps the closest research to our model is by Katoh et al. [24] who investigate the numbers of transitions of the minimum and maximum spanning trees where the points move along different straight lines at different but fixed speeds. They show that the maximum number of possible transitions of MST in L_1 and L_∞ metrics for n linearly moving points in any constant dimension is $O(n^{5/2}\alpha(n))$ where $\alpha(n)$ is the inverse Ackermann's function.

To the best of our knowledge, the problem of finding an 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 a minimum moving spanning tree and a minimum bottleneck moving spanning tree of a set of moving points. We also prove that the distance between two linearly moving points is maximized at time $t = 0$ or $t = 1$.

2.1 Convex Distance Functions

In this section, we recall the notion of a *convex distance function* as appeared in [8, 12, 28].

Let C be a convex body (i.e., a convex compact set with nonempty interior) in $\mathbb{R}^{\dim}$, where dim is a constant. We assume that the origin is contained in the interior of C and that C is centrally symmetric with respect to the origin. For any real number $\lambda \geq 0$, the λ-scaled copy of C is the set

$$\lambda C = \{\lambda z : z \in C\}.$$

For any two points p and q in $\mathbb{R}^{\dim}$, we define their convex distance, with respect to C, as

$$\mathsf{dist}_C(p, q) = \min\{\lambda \geq 0 : q - p \in \lambda C\}.$$

We can visualize this as follows: First, we translate C by the vector p (thus, the origin is translated to p). Then, we scale this translate, by increasing λ, until it "hits" the point q.

Chew and Drysdale [12] have shown that dist_C is a metric on $\mathbb{R}^{\dim}$. Throughout this paper, we assume that dim is constant and $\mathsf{dist}_C(p, q)$ can be computed in $O(1)$ time.

2.2 Definitions

A *moving point* p in the plane is described by a continuous function $p : [0, 1] \to \mathbb{R}^{\dim}$. We assume that p moves on a straight line segment in $\mathbb{R}^{\dim}$. We say that p is at $p(t)$ at time t. We are given a set $S = \{p_1, ..., p_n\}$ of moving points as well as a centrally symmetric convex body C in $\mathbb{R}^{\dim}$. Throughout this paper, we shall use $\overline{w}$ for weight functions of moving points and w for weight functions of ordinary graphs. A *moving spanning tree* T of S is a spanning tree of S and has weight function $\overline{w}_T : [0,1] \to \mathbb{R}$ defined as $\overline{w}_T(t) = \sum_{pq \in T} \mathsf{dist}_C(p(t), q(t))$. Let $\mathcal{T}(S)$ denote the set of all moving spanning trees of S. Let $\overline{w}(T) = \max_t \overline{w}_T(t)$ 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

$$\arg\min_{T \in \mathcal{T}(S)} (\overline{w}(T)) .$$

Let $b_T(t) = \max_{pq \in T} \mathsf{dist}_C(p(t), q(t))$ denote the *bottleneck* of a tree T at time t. Let $b(T) = \max_t b_T(t)$ be the bottleneck of the moving spanning tree 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

$$\arg\min_{T \in \mathcal{T}(S)} (b(T)) .$$

The upper bound graph. Throughout this paper, we shall use a graph whose edge weights are upper bounds for distances between points. We define G_S, as the *upper bound graph* of a set S of moving points, to 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 $[0, 1]$; see Figure 1(b).

2.3 Maximizing the distance between two moving points

Let p and q be two linearly moving points in $\mathbb{R}^{\dim}$. Thus, for any real number t, we can write the positions of p and q at time t as

$$p(t) = a + tu$$

and

$$q(t) = b + tv,$$

where a and b are the positions of p and q at time $t = 0$, and u and v are the velocity vectors of p and q, respectively.

Below, we will prove that the distance $\mathsf{dist}_C(p(t), q(t))$, for $0 \le t \le 1$, is maximized at time $t = 0$ or $t = 1$.

We first consider the case when the point p is stationary, i.e., u is the zero vector, so that $p(t) = a$ for all t.

Lemma 1. *Assume that the point p is stationary. Then, the distance $\mathsf{dist}_C(p(t), q(t))$, for $0 \le t \le 1$, is maximized at time $t = 0$ or $t = 1$.*

Proof. We may assume without loss of generality that a is the origin and, thus, the point p is at the origin at all times. We observe that

$$\max_{0 \le t \le 1} \mathsf{dist}_C(p(t), q(t)) = \min\{\lambda \ge 0 : \lambda C \text{ contains the line segment } q(0)q(1)\}.$$

We may assume without loss of generality that $\mathsf{dist}_C(p(0), q(0)) \le \mathsf{dist}_C(p(1), q(1))$. Set $\lambda = \mathsf{dist}_C(p(1), q(1))$. Then λC contains both $q(0)$ and $q(1)$, with $q(1)$ being on the boundary. Since λC is convex, it contains the entire line segment $q(0)q(1)$. Therefore, for all t with $0 \le t \le 1$, $\mathsf{dist}_C(p(t), q(t)) \le \mathsf{dist}_C(p(1), q(1))$. $\square$

Lemma 2. *Let p and q be two linearly moving points in $\mathbb{R}^{\dim}$. Then, the distance $\mathsf{dist}_C(p(t), q(t))$, for $0 \le t \le 1$, is maximized at time $t = 0$ or $t = 1$.*

Proof. We write $p(t) = a + tu$ and $q(t) = b + tv$, and observe that

$$q(t) - p(t) = (b + t(v - u)) - a.$$

Thus, if we define the stationary point $p'(t) = a$ and the moving point $q'(t) = b + t(v - u)$, then

$$\begin{aligned} \mathsf{dist}_C(p(t), q(t)) &= \min\{\lambda \ge 0 : q(t) - p(t) \in \lambda C\} \\ &= \min\{\lambda \ge 0 : q'(t) - p'(t) \in \lambda C\} \\ &= \mathsf{dist}_C(p'(t), q'(t)). \end{aligned}$$

By Lemma 1, $\mathsf{dist}_C(p'(t), q'(t))$ is maximized at time $t = 0$ or $t = 1$. □

3 Minimum bottleneck moving spanning tree

Since by Lemma 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, and that $b(T) = \max_t b_T(t)$ 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.

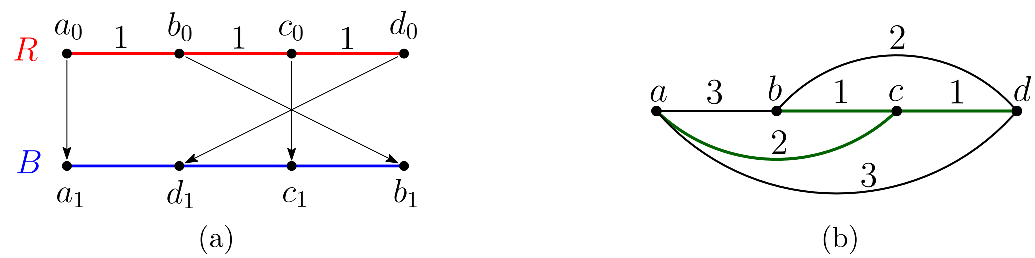

Figure 1: Four points that move from time 0 to time 1. (a) R is the MBST at time 0, and B is the MBST at time 1. (b) The graph G_S; green edges form an MBST of this graph.

Although the above example shows that the computation of an MBMST is not straightforward, we present a simple algorithm for finding an MBMST. Let G_S be the upper bound graph of S as defined in Section 2.2.

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

Proof. Our proof is by contradiction. Let T^* be an MBMST of S and let T be an MBST of G_S of minimum weight. 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 $w(rs) \leqslant b(T^*)$. Thus $w(rs) \leqslant b(T^*) < b(T) = w(pq)$. Let T' be the spanning tree of G_S that is obtained by connecting T_p and T_q by rs. Then $b(T') \leqslant b(T)$ and $w(T') < w(T)$. Then, T' is an MBST of G_S with smaller weight than T, contradicting its definition. □

It follows from Lemma 3 that any MBST of G_S is an MBMST of S. Since an MBST of a graph can be computed in time linear in the size of the graph [10], an MBST of G_S 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 $\mathbb{R}^{\dim}$ under any convex distance function can be computed in $O(n^2)$ time, provided that* $\dim$ *is a constant.*

4 Minimum moving spanning tree

In this section we study the problem of computing an MMST of moving points. First we prove that this problem is NP-hard, even in $\mathbb{R}^2$ under the Euclidean distance metric. Then we propose a simple a simple $O(n^2)$-time 2-approximation algorithm. We also show that our analysis of the approximation ratio is tight. Then in Section 4.5 we improve the running time to $O(n \log n)$ but obtain a $(2+\varepsilon)$-approximation.

4.1 NP-hardness of MMST

Inspired by Arkin et al. [6], we reduce the Partition problem, which is known to be weakly NP-hard [19], to the MMST problem in the Euclidean plane. In one formulation of the Partition problem, we are given $n > 0$ positive integers $a_0, \dots, a_{n-1}$ and must decide whether there is a subset $S \subseteq \{0, \dots, n-1\}$ 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 $\ell = \max\{a_0, \dots, a_{n-1}\}$ and then, for each $i \in \{0, \dots, n-1\}$, we put the following points into our set P of moving points (Figure 2):

- 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 $(i\ell, \ell)$.

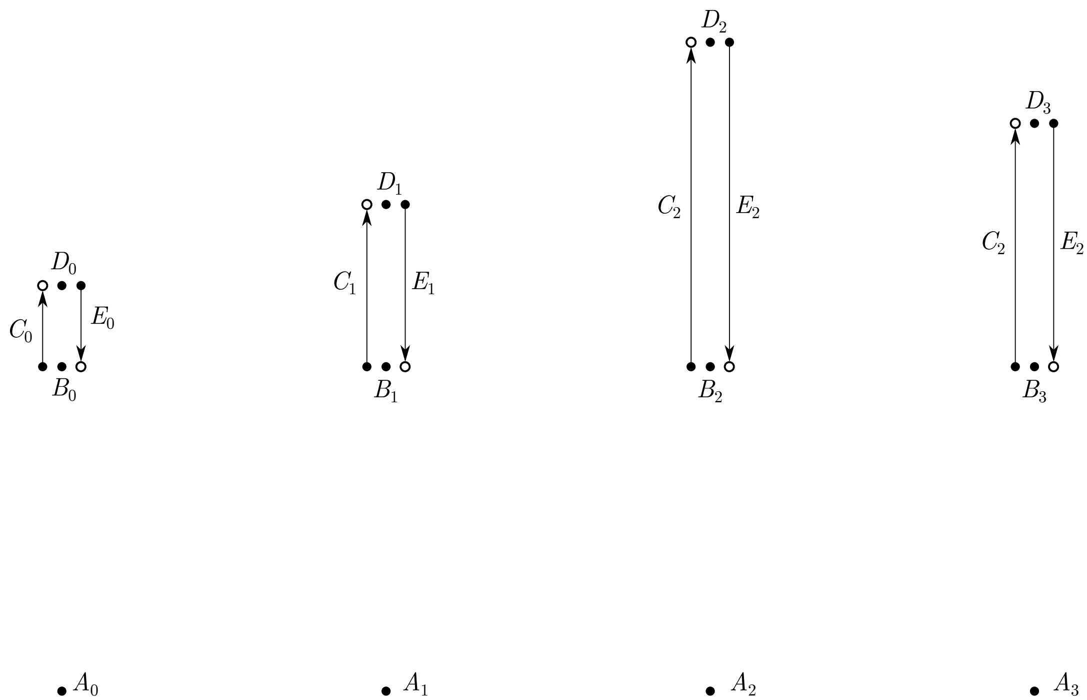

Figure 2: Initial position of the reduction for $n = 4$ and $(a_0, a_1, a_2, a_3) = (1, 2, 4, 3)$. The moving are indicated with arrows and their final position is indicated with a white dot. For clarity, the x-coordinates of points C_i and E_i are slightly shifted so that there is no overlap.

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

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

Theorem 5. *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 $\overline{w}_T(t)$ denotes the weight of T at time t. The distance function between two moving points in the plane is convex (this is implied by the result of Alt and Godau [5] that the free space diagram of any two line segments is convex). Thus the weight of each edge of T is attained at time 0 or 1 (this is also implied by Lemma 2). Indeed $\overline{w}(T) = \max\{\overline{w}_T(0), \overline{w}_T(1)\}$, i.e., the maximum weight of T is also attained at time 0 or 1, because the sum of convex functions is convex.

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 3). 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

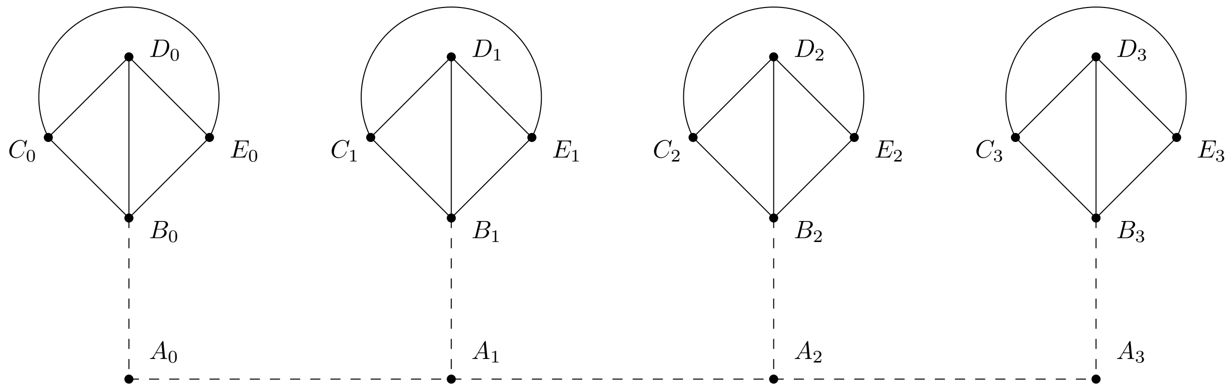

Figure 3: The (topological) edges in K_0 (dashed) and in $K_1 \setminus K_0$ (solid).

we bridge the two components with e', we will be left with a spanning tree T' with $\overline{w}(T') \leq \overline{w}(T)$ 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 $\overline{w}_{T_i}(0)$ and $\overline{w}_{T_i}(1)$ 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 $\overline{w}_{T_i}(0) = a_i$ and $\overline{w}_{T_i}(1) = 2a_i$ 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 $\overline{w}_{T_i}(0) = 2a_i$ and $\overline{w}_{T_i}(1) = a_i$ 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 4. Since the diameter of $\{B_i, C_i, D_i, E_i\}$ is a_i for all $i \in \{0, \ldots, n-1\}$ at $t \in \{0,1\}$, $\overline{w}_{T_i}(0) \geq a_i$ and, similarly, $\overline{w}_{T_i}(1) \geq a_i$. 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 or at time 1. Therefore, $\overline{w}_{T_i}(0) + \overline{w}_{T_i}(1) \geq 3a_i$, so (i) $\overline{w}_{T_i}(0) \geq 2a_i$ or (ii) $\overline{w}_{T_i}(1) \geq 2a_i$. Then, given an optimal solution T^*, we can replace T_1 by the $(1,2)$-tree in case (i) is true, or by the $(2,1)$-tree in case (ii) is true, without affecting the maximum weight or local connectivity. 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

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

while

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

Therefore, the cost of T^* is

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

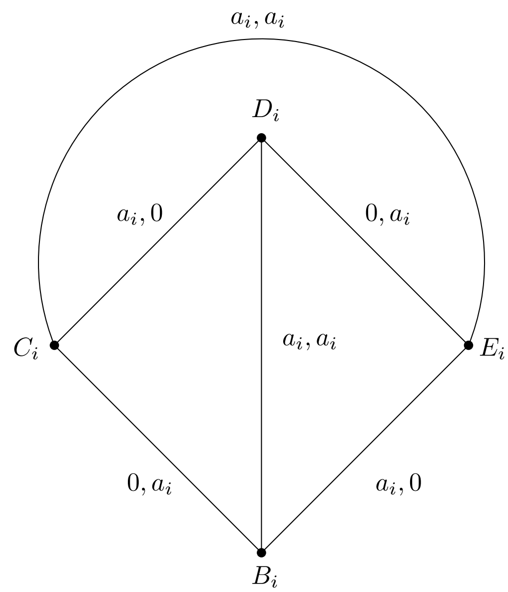

Figure 4: Edges between B_i, C_i, D_i and E_i labeled with their weights at times 0 and 1.

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

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

We claim that (1) 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

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

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

4.2 A 2-approximation algorithm

Our algorithm is very simple and just computes an MST of the upper bound graph G_S defined in Section 2.2.

Lemma 6. *The weight of any MST of G_S is at most two times the weight of any MMST of S.*

Proof. Let T be any MST of G_S and let T^* be any MMST of S. Recall that $\overline{w}(T^*) = \max_t \overline{w}_{T^*}(t)$ is the weight of the moving spanning tree T^*. Let $w(T) = \sum_{pq \in T} w(pq)$ be the weight of the spanning tree T. We are going to show that $w(T) \leqslant 2 \cdot \overline{w}(T^*)$. Let T' be a spanning tree of G_S isomorphic to T^*. Similar to G_S, each edge pq of T' has weight $w(pq)$ which is the maximum distance between p and q in the time interval $[0, 1]$. Since T is an MST of G_S, we have $w(T) \leqslant w(T')$.

By Lemma 2 the largest distance between two points is achieved at time 0 or at time 1. Let E_0^* be the set of edges of T^* whose endpoints achieve their largest distance at time 0. Define E_1^* analogously. Then $\sum_{pq \in E_0^*} w(pq) \leqslant \overline{w}_{T^*}(0) \leqslant \overline{w}(T^*)$ and $\sum_{pq \in E_1^*} w(pq) \leqslant \overline{w}_{T^*}(1) \leqslant \overline{w}(T^*)$. Moreover, $w(T') = \sum_{pq \in E_0^*} w(pq) + \sum_{pq \in E_1^*} w(pq)$ by definition. By combining these inequalities we get

$$w(T) \leqslant w(T') = \sum_{pq \in E_0^*} w(pq) + \sum_{pq \in E_1^*} w(pq) \leqslant \overline{w}(T^*) + \overline{w}(T^*) = 2 \cdot \overline{w}(T^*).$$

□

A minimum spanning tree of G_S can be computed in $O(n^2)$ time using Prim's MST algorithm using Fibonacci heaps [14]. The following theorem summarizes our result in this section.

Theorem 7. *There is an $O(n^2)$-time 2-approximation algorithm for computing the minimum moving spanning tree of n moving points in $\mathbb{R}^{\dim}$ under any convex distance function, provided that* $\dim$ *is a constant.*

4.3 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 $\epsilon > 0$. Consider the point set $S = \{p_1, p_2, p_3\} \subset \mathbb{R}^2$, where p_1, p_2 and p_3 are defined as follows. The points $p_1 = (0, 0)$ and $p_2 = (1, 0)$ are stationary. The point p_3 moves from $(-\epsilon, 0)$ at time $t = 0$ to $(1 + \epsilon, 0)$ at time $t = 1$ (refer to Figure 5). We now describe the spanning tree produced by our 2-approximation algorithm on S. In G_S, the edge p_1p_2 has weight 1, and the weights of p_1p_3 and p_2p_3 are $1 + \varepsilon$. Hence, when we compute an MST of G_S, we get the edges p_1p_2 and (due to symmetry) p_1p_3. The moving spanning tree of S defined by the edges p_1p_2 and p_1p_3 has weight $2 + \epsilon$.

Now consider the moving spanning tree of S defined by the edges p_1p_3 and p_2p_3. The weight of this moving spanning tree of S is $1 + 2\epsilon$. Hence, the approximation factor is at least

$$\frac{2 + \epsilon}{1 + 2\epsilon} = 2 - \frac{3\epsilon}{1 + 2\epsilon},$$

which can be made arbitrarily close to 2 by taking ϵ small enough. In other words, for every $\delta > 0$ there exists a point set for which our algorithm has an approximation ratio of $2 - \delta$.

4.4 Metric spaces and their doubling dimension

Let (V, dist) be a metric space. For any point u in V and any real number $\rho > 0$, the ball with center u and radius ρ is the set

$$\mathsf{ball}_{\mathsf{dist}}(u, \rho) = \{v \in V : \mathsf{dist}(u, v) \leq \rho\}.$$

Let τ be the smallest integer such that for every real number $\rho > 0$, every ball of radius ρ can be covered by at most τ balls of radius $\rho/2$. Note that all balls must be centered at points of the set V. The *doubling dimension* of (V, dist) is defined to be $\log \tau$.

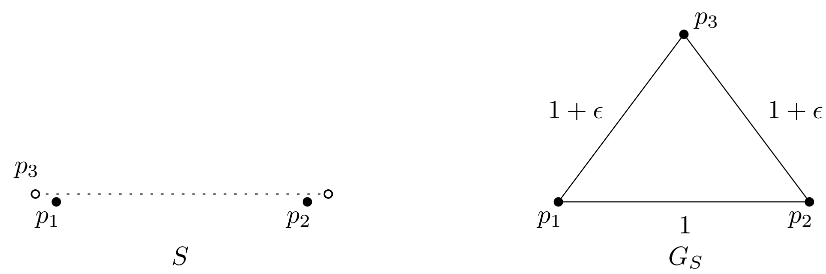

Figure 5: Tightness of the approximation ratio 2.

As an example, for any two points $u = (u_1, \ldots, u_{\text{dim}})$ and $v = (v_1, \ldots, v_{\text{dim}})$ in $\mathbb{R}^{\text{dim}}$, their L_∞-distance is defined as

$$\|uv\|_\infty = \max_{1 \le i \le \text{dim}} |u_i - v_i|.$$

The following lemma gives an upper bound on the doubling dimension of the metric space induced by this metric; this lemma will be used in Section 4.5.2.

Lemma 8. *Let V be a finite set of points in $\mathbb{R}^{\text{dim}}$. Then, the doubling dimension of the metric space $(V, \|\cdot\|_\infty)$ is at most* $\text{dim} \cdot \log 5$.

Proof. Let u be a point in V and let $\rho > 0$ be a real number. Let $B = \mathsf{ball}_\infty(u, \rho)$ be the L_∞-ball of radius ρ that is centered at u. To prove the claim, we have to show that B can be covered by at most 5^{dim} L_∞-balls of radius $\rho/2$. Note that these balls must be centered at points of V.

Consider the following algorithm:

1. Set $X = B$. (Note that $X \subseteq V$.)
2. Set $k = 0$.
3. While $X \neq \emptyset$:
   (a) Set $k = k + 1$.
   (b) Let v_k be an arbitrary point in X.
   (c) Set $X = X \setminus \mathsf{ball}_\infty(v_k, \rho/2)$.

The points $v_1, v_2, \ldots, v_k$ that are computed by this algorithm have the properties that

$$\|v_i v_j\|_\infty > \rho/2 \text{ for all } i \neq j$$

and

$$B \subseteq \bigcup_{i=1}^{k} \mathsf{ball}_\infty(v_k, \rho/2).$$

For each i with $1 \le i \le k$, let

$$H_i = \{z \in \mathbb{R}^{\text{dim}} : \|v_i z\|_\infty \le \rho/4\}.$$

Note that H_i is a hypercube centered at v_i with sides of length $\rho/2$. These hypercubes are pairwise disjoint. Finally, let

$$H = \{z \in \mathbb{R}^{\dim} : \|uz\|_\infty \leq 5\rho/4\}.$$

Then all hypercubes H_i, for $1 \leq i \leq k$, are contained in the hypercube H. By considering their volumes, it follows that

$$k \leq \frac{(10\rho/4)^{\dim}}{(\rho/2)^{\dim}} = 5^{\dim}.$$

□

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

Let S be a set of n linearly moving points in $\mathbb{R}^{\dim}$. Section 4.2 showed that the weight of a minimum spanning tree of the upper bound graph G_S gives a 2-approximation to the MMST. Since G_S has $\Theta(n^2)$ edges, it takes $\Theta(n^2)$ time to compute its minimum spanning tree.

In this section, we prove that a $(1+\varepsilon)$-approximation to the minimum spanning tree of G_S 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 an MMST of a set S of linearly moving points.

4.5.1 The fatness of the convex set C

Recall that C is a compact and convex set in $\mathbb{R}^{\dim}$ that contains the origin in its interior. We assume that C is centrally symmetric with respect to the origin. Let

$$H = \{z \in \mathbb{R}^{\dim} : \|oz\|_\infty \leq 1\}$$

denote the L_∞-ball of radius one that is centered at the origin o. Note that H is a hypercube with sides of length two.

We define the real numbers

$$d_\infty^- = \min\{\|oz\|_\infty : z \in \partial C\}$$

and

$$d_\infty^+ = \max\{\|oz\|_\infty : z \in \partial C\}.$$

Let H^- and H^+ denote the L_∞-balls centered at the origin with radii d_∞^- and d_∞^+, respectively. Note that H^- is the largest hypercube centered at the origin that is contained in C, and H^+ is the smallest hypercube centered at the origin that contains C. In particular,

$$H^- \subseteq C \subseteq H^+.$$

Moreover, we have

$$H^- = d_\infty^- H$$

and

$$H^+ = d_\infty^+ H.$$

We define the *fatness* of C to be

$$f(C) = \frac{d_\infty^+}{d_\infty^-}.$$

The following lemma states that the convex distance function dist_C and the L_∞-metric are related by the fatness $f(C)$.

Lemma 9. *Let p and q be two points in $\mathbb{R}^{\dim}$. Then*

$$d_\infty^- \cdot \mathsf{dist}_C(p,q) \le \|pq\|_\infty \le d_\infty^+ \cdot \mathsf{dist}_C(p,q).$$

Proof. Since

$$\{\lambda \ge 0 : q - p \in \lambda C\} \subseteq \{\lambda \ge 0 : q - p \in \lambda H^+\} = \{\lambda \ge 0 : q - p \in \lambda d_\infty^+ H\},$$

we have

$$\min\{\lambda \ge 0 : q - p \in \lambda d_\infty^+ H\} \le \min\{\lambda \ge 0 : q - p \in \lambda C\}.$$

Observe that the minimum on the right-hand side is equal to $\mathsf{dist}_C(p,q)$, whereas the minimum on the left-hand side is equal to

$$\min\{\mu/d_\infty^+ \ge 0 : q - p \in \mu H\} = \frac{1}{d_\infty^+} \min\{\mu \ge 0 : q - p \in \mu H\} = \frac{1}{d_\infty^+} \|pq\|_\infty.$$

This proves the second inequality in the lemma. The proof of the first inequality follows by a symmetric argument. □

4.5.2 The approximation algorithm

Let S be a set of n linearly moving points in $\mathbb{R}^{\dim}$. For any point p in S, define the point

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

in $\mathbb{R}^{2\cdot\dim}$. Doing this for all points in S, we obtain a set S' of n points in $\mathbb{R}^{2\cdot\dim}$. For any two points P and Q in S', define their distance to be

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

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

Corollary 11 below states that dist satisfies the properties of a metric. Its proof uses the following lemma.

Lemma 10. *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 $a \ne b$, 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

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

Using the inequality

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

it follows that

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

□

Corollary 11. *The pair (S', dist) is a metric space.*

Proof. The proof follows from Lemma 10 and the definition of dist. □

In Lemma 13, we will prove that the doubling dimension of (S', dist) is bounded from above by a function of the dimension dim and the fatness $f(C)$ of the convex set C. Our strategy will be as follows. First, in Lemma 12, we use Lemma 9 to obtain upper and lower bounds on the ratio $\|PQ\|_\infty / \mathsf{dist}(P,Q)$. Then we use these bounds, together with Lemma 8, to obtain an upper bound on the doubling dimension of (S', dist).

Lemma 12. *Let P and Q be two points in S'. Then*

$$d_\infty^- \cdot \mathsf{dist}(P,Q) \leq \|PQ\|_\infty \leq d_\infty^+ \cdot \mathsf{dist}(P,Q).$$

Proof. Observe that

$$\|PQ\|_\infty = \max(\|p(0)q(0)\|_\infty, \|p(1)q(1)\|_\infty)$$

and

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

The claim follows from Lemma 9. □

Lemma 13. *The doubling dimension of the metric space (S', dist) is $O(\dim \cdot \log(f(C)))$.*

Proof. Let P be a point in S', let $\rho > 0$ be a real number, and let $B_{\mathsf{dist}} = \mathsf{ball}_{\mathsf{dist}}(P, \rho)$. We will prove that B_{dist} can be covered by $f(C)^{O(\dim)}$ dist-balls of radius $\rho/2$.

Let H be the L_∞-ball with center P and radius $d_\infty^+ \rho$. It follows from Lemma 12 that

$$B_{\mathsf{dist}} \subseteq H.$$

Using Lemma 8, with dimension $2 \cdot \dim$, by applying the definition of doubling dimension

$$\ell := 1 + \lceil \log(f(C)) \rceil$$

times, we can cover H by $k := 5^{2\ell \dim}$ L_∞-balls, each of radius

$$d_\infty^+ \rho / 2^\ell \leq d_\infty^- \rho / 2.$$

Let these balls have centers $C_1, \ldots, C_k$. For each i with $1 \leq i \leq k$, define $B_{i,\mathsf{dist}} = \mathsf{ball}_{\mathsf{dist}}(C_i, \rho/2)$. It follows from Lemma 12 and our choice of ℓ that

$$\mathsf{ball}_\infty(C_i, d_\infty^+ \rho / 2^\ell) \subseteq B_{i,\mathsf{dist}}.$$

Thus,

$$B_{\mathsf{dist}} \subseteq H \subseteq \bigcup_{i=1}^{k} \mathsf{ball}_\infty(C_i, d_\infty^+ \rho/2^\ell) \subseteq \bigcup_{i=1}^{k} B_{i,\mathsf{dist}},$$

i.e., we have covered the ball B_{dist} by

$$k = 5^{2\ell \dim} = f(C)^{O(\dim)}$$

dist-balls of radius $\rho/2$. ☐

Lemma 14. *Let $0 < \varepsilon < 1$ be a real number. In $(1/\varepsilon)^{O(\dim \cdot \log(f(C)))} n \log n$ expected time, we can compute a $(1+\varepsilon)$-approximation to the minimum spanning tree of the metric space (S', dist).*

Proof. Har-Peled and Mendel [21] have shown that for any n-point metric space of doubling dimension ddim, a $(1+\varepsilon)$-spanner with $(1/\varepsilon)^{O(\mathsf{ddim})} n$ edges can be computed in $2^{O(\mathsf{ddim})} n \log n + (1/\varepsilon)^{O(\mathsf{ddim})} n$ 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., [30, Theorem 1.3.1].)

Since the spanner has $(1/\varepsilon)^{O(\mathsf{ddim})} n$ edges, its minimum spanning tree can be computed in $(1/\varepsilon)^{O(\mathsf{ddim})} n \log n$ time using Prim's MST algorithm combined with a binary min-heap.

Thus, the total expected running time is

$$2^{O(\mathsf{ddim})} n \log n + (1/\varepsilon)^{O(\mathsf{ddim})} n + (1/\varepsilon)^{O(\mathsf{ddim})} n \log n = (1/\varepsilon)^{O(\mathsf{ddim})} n \log n.$$

By Lemma 13, in our case, we have $\mathsf{ddim} = O(\dim \cdot \log(f(C)))$. ☐

As a consequence of Lemma 14 and the fact that $\mathsf{dist}(P, Q) = w(pq)$, we have the following theorem.

Theorem 15. *Let $0 < \varepsilon < 1$ be a real number, and let dist_C be a convex distance function in $\mathbb{R}^{\dim}$, where* dim *is a constant. For any set S of n linearly moving points in $\mathbb{R}^{\dim}$, we can compute, in $(1/\varepsilon)^{O(\dim \cdot \log(f(C)))} n \log n$ expected time, a $(2+\varepsilon)$-approximation for the minimum moving spanning tree of S.*

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. [6]. We are grateful to anonymous reviewers who meticulously verified our proofs, and provided valuable feedback that improved the clarity of the paper, simplified our construction in Section 4.3, and lead to the generalization of our results to higher dimensions and to any convex distance function.

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] P. K. Agarwal, D. Eppstein, L. J. Guibas, and M. R. Henzinger. Parametric and kinetic minimum spanning trees. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 596–605, 1998.

[3] H. A. Akitaya, A. Biniaz, P. Bose, J.-L. D. Carufel, A. Maheshwari, L. F. S. X. d. Silveira, and M. Smid. The minimum moving spanning tree problem. In *Workshop on Algorithms and Data Structures*, pages 15–28. Springer, 2021.

[4] 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.

[5] H. Alt and M. Godau. Computing the fréchet distance between two polygonal curves. *International Journal of Computational Geometry & Applications*, 5(01n02):75–91, 1995.

[6] 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.

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

[8] T. Bonnesen and W. Fenchel. *Theory of Convex Bodies.* Translated from the German and edited by L. Boron, C. Christenson and B. Smith. BCS Associates, Moscow, Idaho, 1987.

[9] 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.

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

[11] T. M. Chan. Finding the shortest bottleneck edge in a parametric minimum spanning tree. In *Proceedings of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 917–918, 2005.

[12] L. P. Chew and R. L. Drysdale. Voronoi diagrams based on convex distance functions. In *Proceedings of the 1st ACM Symposium on Computational Geometry*, pages 235–244, 1985.

[13] 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.

[14] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms.* MIT press, 2022.

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

[16] D. Eppstein. A stronger lower bound on parametric minimum spanning trees. In *Proceedings of the 17th International Symposium on Algorithms and Data Structures (WADS)*, pages 343–356, 2021.

[17] 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.

[18] D. Fernández-Baca and G. Slutzki. Linear-time algorithms for parametric minimum spanning tree problems on planar graphs. *Theor. Comput. Sci.*, 181(1):57–74, 1997.

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

[20] D. Gusfield. Bounds for the parametric minimum spanning tree problem. In *Proceedings of the West Coast Conference on Combinatorics, Graph Theory and Computing (Humboldt State Univ., Arcata, Calif., 1979)*, pages 173–181, 1980.

[21] 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.

[22] 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.

[23] S. Jaffry, R. Hussain, X. Gui, and S. F. Hasan. A comprehensive survey on moving networks. *IEEE Communications Surveys & Tutorials*, 23(1):110–136, 2020.

[24] N. Katoh, T. Tokuyama, and K. Iwano. On minimum and maximum spanning trees of linearly moving points. *Discret. Comput. Geom.*, 13:161–176, 1995.

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

[26] 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.

[27] 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.

[28] H. Minkowski. *Geometrie der Zahlen.* B. G. Teubner, Leipzig und Berlin, 1896/1910; reprinted by Chelsea, 1953.

[29] C. Monma and S. Suri. Transitions in geometric minimum spanning trees. *Discrete & Computational Geometry*, 8(3):265–293, 1992.

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

[31] 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).

[32] H. Wang and Y. Zhao. Computing the Minimum Bottleneck Moving Spanning Tree. In *47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022)*, volume 241 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 82:1–82:15, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[33] J. Wulms, J. Buchmüller, W. Meulemans, K. Verbeek, and B. Speckmann. Stable visual summaries for trajectory collections. In *14th IEEE Pacific Visualization Symposium*, pages 61–70, 2021.

[34] D. Zhou and J. Gao. Maintaining approximate minimum Steiner tree and k-center for mobile agents in a sensor network. In *Proceedings of the 29th International Conference on Computer Communications (INFOCOM)*, pages 511–515, 2010.