Approximating the Geometric Minimum-Diameter Spanning Tree

Joachim Gudmundsson*
Dept. of Comp. Science
Utrecht University
The Netherlands
joachim@cs.uu.nl

Herman Haverkort†
Dept. of Comp. Science
Utrecht University
The Netherlands
herman@cs.uu.nl

Sang-Min Park
Dept. of Comp. Science
KAIST
Korea
smpark@jupiter.kaist.ac.kr

Chan-Su Shin
School of Electr. and Inform. Engineering
Hankuk University of Foreign Studies
Korea
cssin@hufs.ac.kr

Alexander Wolff
Institut für Mathematik und Informatik
Universität Greifswald
Germany
awolff@uni-greifswald.de

ABSTRACT

Let P be a set of n points in the plane. The geometric minimum-diameter spanning tree (MDST) of P is a tree that spans P and minimizes the Euclidian length of the longest path. It is known that there is always a mono- or a dipolar MDST, i.e. a MDST whose longest path consists of two or three edges, respectively. The more difficult dipolar case can so far only be solved in $O(n^3)$ time. In this paper we give an $O(n \log n)$-time approximation scheme for the MDST.

1. INTRODUCTION

The geometric minimum-diameter spanning tree (MDST) can be seen as a network without cycles that minimizes the maximum travel time between any two sites connected by the network. This is of importance e.g. in communication systems where the maximum delay in delivering a message is to be minimized. Ho et al. showed that there always is a mono- or a dipolar MDST [4]. For a different proof, see [3]. Ho et al. gave an $O(n \log n)$-time algorithm for the monopolar and an $O(n^3)$-time algorithm for the dipolar case [4]. In the more difficult dipolar case the objective is to find a minimum-diameter dipolar spanning tree (MDdST), i.e. a tree with two roots $x, y \in P$ such that the function $r_x + |xy| + r_y$ is minimized, where $|xy|$ is the Euclidean distance of x and y, and r_x and r_y are the radii of two disks centered at x and y that cover P.

We show that there is a fast approximation scheme for the MDdST. More precisely, given a set P of n points and some $\varepsilon > 0$ we show how to compute in $O(n \log n)$ time and space a dipolar tree whose diameter is at most $(1+\varepsilon)$ times as long as the diameter of a MDdST. Our approximation scheme is based on a well-separated pair decomposition [1] of P, a very powerful tool that we will briefly review. This decomposition makes it possible to consider only a linear number of pairs of points on the search for the two poles of an approximate MDdST. The main novelty of our scheme is a discretization of the orientations of line segments to a constant number of orientations. We only consider projections of P onto those directions, which gives us fast access to farthest neighbors. For the full paper, see [2].

2. THE WELL-SEPARATED PAIR DECOMPOSITION

Our algorithm uses the well-separated pair decomposition (WSPD) of Callahan and Kosaraju [1] that we now review.

DEFINITION 1. *Let $\tau > 0$ be a real number, and let A and B be two finite sets of points in $\mathbb{R}^d$. We say that A and B are* well-separated w.r.t. τ, *if there are two disjoint d-dimensional balls C_A and C_B both of radius r such that $A \subset C_A$, $B \subset C_B$, and the distance between C_A and C_B is at least equal to τr.*

The parameter τ will be referred to as the *separation constant*. The following lemma follows easily from Definition 1.

LEMMA 1. *Let A and B be two finite sets of points that are well-separated w.r.t. τ, let x and p be points of A, and let y and q be points of B. Then (i) $|xy| \leq (1 + 2/\tau) \cdot |xq|$, (ii) $|xy| \leq (1 + 4/\tau) \cdot |pq|$, (iii) $|px| \leq (2/\tau) \cdot |pq|$, and (iv) the angle between the line segments pq and py is at most* $\arcsin(2/\tau)$.

*Supported by the Swedish Foundation for International Cooperation in Research and Higher Education.

†Supported by the Netherlands' Organization for Scientific Research.

DEFINITION 2. *Let P be a set of n points in $\mathbb{R}^d$, and $\tau > 0$ a real number. A* well-separated pair decomposition *for P w.r.t. τ is a sequence $(A_1, B_1), (A_2, B_2), \ldots, (A_\ell, B_\ell)$ of pairs of non-empty subsets of P such that each pair is well-separated w.r.t. τ and for any $p \neq q \in P$, there is exactly one pair (A_i, B_i) with $p \in A_i$ and $q \in B_i$ or vice versa.*

Callahan and Kosaraju show that a WSPD with $\ell = O(\tau^2 n)$ can be computed using $O(n \log n + \tau^2 n)$ time and space.

3. A STRAIGHT-FORWARD APPROXIMATION SCHEME

The approximation algorithm consists of two subalgorithms: the first algorithm computes a minimum-diameter monopolar spanning tree and the second computes an approximation of the MDdST. When these two trees have been computed we output the one with smaller diameter. According to [4] there exists a MDST that is either a monopolar or a dipolar tree. The minimum-diameter monopolar tree can be computed in time $O(n \log n)$, hence we will focus on the problem of computing a MDdST. Let $d_{\min}$ be the diameter of a MDdST and let $\mathcal{T}_{pq}$ denote a spanning tree with dipole $\{p, q\}$ whose diameter is minimum among all such trees. For any dipolar spanning Tree $\mathcal{T}$ with dipole $\{u, v\}$ let $r_u(\mathcal{T})$ ($r_v(\mathcal{T})$) be the length of the longest edge of $\mathcal{T}$ incident to u (v) without taking into account the edge uv. When it is clear which tree we refer to, we use r_u and r_v.

OBSERVATION 1. *Let $(A_1, B_1), \ldots, (A_\ell, B_\ell)$ be a well-separated pair decomposition (WSPD) of P with separation constant τ, and let p and q be any two points in P. Then there is a pair (A_i, B_i) such that for every point $u \in A_i$ and every point $v \in B_i$ the inequality* diam $\mathcal{T}_{uv} \leq (1 + 8/\tau) \cdot$ diam $\mathcal{T}_{pq}$ *holds.*

PROOF. According to Definition 2 there is a pair (A_i, B_i) in the WSPD such that $p \in A_i$ and $q \in B_i$. If u is any point in A_i and v is any point in B_i, then let $\mathcal{T}$ be the tree with poles u and v where u is connected to v, p and each neighbor of p in $\mathcal{T}_{pq}$ except q is connected to u, and q and each neighbor of q in $\mathcal{T}_{pq}$ except p is connected to v. By Lemma 1(ii) $|uv| \leq (1 + 4/\tau)|pq|$ and by Lemma 1(iii) $r_u \leq |up| + r_p \leq 2|pq|/\tau + r_p$. Since diam $\mathcal{T} = r_u + |uv| + r_v$ we have

$$\begin{aligned}\text{diam } \mathcal{T} &\leq \left(r_p + 2\frac{|pq|}{\tau}\right) + \left(|pq| + 4\frac{|pq|}{\tau}\right) + \left(r_q + 2\frac{|pq|}{\tau}\right) \\ &< \left(1 + \frac{8}{\tau}\right) \cdot \text{diam } \mathcal{T}_{pq}.\end{aligned}$$

The lemma follows since $\mathcal{T}_{uv}$ has minimum diameter among all dipolar spanning trees with dipole $\{u, v\}$. □

A first algorithm is now obvious. For each pair (A_i, B_i) in the WSPD of P pick *any* vertex $p \in A_i$ and *any* vertex $q \in B_i$, and compute $\mathcal{T}_{pq}$. This is done by checking every possible radius of a disk centered at p as in [4] in $O(n \log n)$ time. Setting τ to $8/\varepsilon$ yields:

LEMMA 2. *A dipolar tree $\mathcal{T}$ with* diam $\mathcal{T} \leq (1 + \varepsilon) \cdot d_{\min}$ *can be computed in $O(\frac{n}{\varepsilon^2} + n^2 \log n)$ time using $O(\frac{n}{\varepsilon^2} + n \log n)$ space.*

4. A FAST APPROXIMATION SCHEME

Now we describe a faster algorithm and show its time complexity; we will prove its correctness in Section 5.

THEOREM 1. *A dipolar tree $\mathcal{T}$ with* diam $\mathcal{T} \leq (1+\varepsilon) \cdot d_{\min}$ *can be computed in $O(\frac{n}{\varepsilon^3} + \frac{n}{\varepsilon} \log n)$ time using $O(\frac{n}{\varepsilon^2} + n \log n)$ space.*

The idea of the algorithm is again to check a linear number of pairs of points, using the WSPD, but to speed up the computation of the disks around the two poles. Note that we need to find a close approximation of the diameters of the disks to be able to guarantee a $(1 + \varepsilon)$-approximation of the MDdST. Obviously we cannot afford to try all possible disks for all possible pairs of poles. Instead of checking the disks we will show in the analysis that it suffices to check a constant number of partitions of the points among the poles. The partition of points is done by cuts that are orthogonal to the line through the poles. We cannot afford to do this for each possible pair. Instead we select a constant number of orientations and use a constant number of orthogonal cuts for each orientation. For each cut we calculate for each point in P the approximate distance to the farthest point on each side of the cut. Below we give a more detailed description of the algorithm. For its pseudocode refer to Algorithm 1.

Phase 1: Initializing. Choose an auxiliary positive constant $\kappa < \min\{0.9\epsilon, 1/2\}$. As will be clear later, this parameter can be used to fine-tune which part of the algorithm contributes how much to the uncertainty and to the running time. In phase 3 the choice of the separation constant τ will depend on the value of κ and ε.

DEFINITION 3. *A set of points P is said to be l-ordered if the points are ordered with respect to their orthogonal projection onto the line l.*

Let l_i be the line with angle $\frac{i\pi}{\gamma}$ to the horizontal line, where $\gamma = \lceil 4/\kappa \rceil$. This implies that for an arbitrary line l there exists a line l_i such that $\angle l_i l \leq \frac{\pi}{2\gamma}$. For each i, $1 \leq i \leq \gamma$, sort the input points with respect to the l_i-ordering. We obtain γ sorted lists $F = \{F_1, \ldots, F_\gamma\}$. Each point p in F_i has a pointer to itself in $F_{(i \bmod \gamma)+1}$. The time to construct these lists is $O(\gamma n \log n)$.

For each l_i, rotate P and l_i such that l_i is horizontal and consider the orthogonal projection of the points in P onto l_i. For simplicity we denote the points in P from left to right on l_i by $p_1, \ldots, p_n$. Let d_i denote the horizontal distance between p_1 and p_n. Let b_{ij}, $1 \leq j \leq \gamma$, be the point on l_i at distance $\frac{j d_i}{\gamma+1}$ to the right of p_1. Let L_{ij} and R_{ij} be the set of points to the left and to the right of b_{ij} respectively.

For each point b_{ij} on l_i we construct γ pairs of lists, denoted L'_{ijk} and R'_{ijk}, where $1 \leq k \leq \gamma$. A list L'_{ijk} (R'_{ijk}) contains the set of points in L_{ij} (R_{ij}) sorted according to the l_k-ordering. Such a list can be constructed in linear time since the ordering is given by the list F_k. (Actually it is not necessary to store the lists L'_{ijk} and R'_{ijk}: we only need to store the first and the last point in each list.) Hence the total time complexity needed to construct the lists is

Algorithm 1 Approx-MDdST(P, ε)

Ensure: diam $\mathcal{T} \leq (1+\varepsilon)\, d_{\min}$
Phase 1: initializing
1: choose $\kappa \in (0, \min\{0.9\varepsilon, 1/2\})$
2: $\gamma \leftarrow \lceil 4/\kappa \rceil$
3: **for** $i \leftarrow 1$ **to** γ **do**
4: $\quad l_i \leftarrow$ line with angle $i\frac{\pi}{\gamma}$ to the horizontal
5: $\quad F_i \leftarrow l_i$-ordering of P
6: **end for**
7: **for** $i \leftarrow$ **to** γ **do**
8: $\quad$ rotate P and l_i such that l_i is horizontal
9: $\quad$ let $p_1, \ldots, p_n$ be the points in F_i from left to right
10: $\quad d_i \leftarrow |p_1.x - p_n.x|$
11: $\quad$ **for** $j \leftarrow 1$ **to** γ **do**
12: $\quad\quad b_{ij} \leftarrow$ point on l_i at dist. $j\frac{d_i}{\gamma+1}$ to the right of p_1
13: $\quad\quad$ **for** $k \leftarrow 1$ **to** γ **do**
14: $\quad\quad\quad L'_{ijk} \leftarrow l_k$-ordered subset of F_k to the left of b_{ij}
15: $\quad\quad\quad R'_{ijk} \leftarrow l_k$-ordered subset of F_k to the right of b_{ij}
16: $\quad\quad$ **end for**
17: $\quad$ **end for**
18: **end for**
Phase 2: computing approximate farthest neighbors
19: **for** $i \leftarrow 1$ **to** γ **do**
20: $\quad$ **for** $j \leftarrow 1$ **to** γ **do**
21: $\quad\quad$ **for** $k \leftarrow 1$ **to** n **do**
22: $\quad\quad\quad N(p_k, i, j, L) \leftarrow p_k$ (dummy)
23: $\quad\quad\quad$ **for** $l \leftarrow 1$ **to** γ **do**
24: $\quad\quad\quad\quad p_{\min} \leftarrow$ first point in L'_{ijl}
25: $\quad\quad\quad\quad p_{\max} \leftarrow$ last point in L'_{ijl}
26: $\quad\quad\quad\quad$ **if** $|p_k p_{\min}| > |p_k p_{\max}|$ **then**
27: $\quad\quad\quad\quad\quad f \leftarrow p_{\min}$
28: $\quad\quad\quad\quad$ **else**
29: $\quad\quad\quad\quad\quad f \leftarrow p_{\max}$
30: $\quad\quad\quad\quad$ **end if**
31: $\quad\quad\quad\quad$ **if** $|p_k f| > |p_k N(p_k, i, j, L)|$ **then**
32: $\quad\quad\quad\quad\quad N(p_k, i, j, L) \leftarrow f$
33: $\quad\quad\quad\quad$ **end if**
34: $\quad\quad\quad$ **end for**
35: $\quad\quad$ **end for**
36: $\quad\quad$ repeat lines 21–35 with R instead of L
37: $\quad$ **end for**
38: **end for**
Phase 3: testing pole candidates
39: $\tau = 8(\frac{1+\varepsilon}{(1+\varepsilon-(1+\kappa)(1+\kappa/24)} - 1)$
40: build WSPD for P with separation constant τ
41: $d \leftarrow \infty$ {smallest diameter so far}
42: **for** each pair (A, B) in WSPD **do**
43: $\quad$ choose any two points $u \in A$ and $v \in B$
44: $\quad D \leftarrow \infty$ {approx. diam. of tree with poles u and v}
45: $\quad l_{(u,v)} \leftarrow$ the line through u and v
46: $\quad$ find l_i such that $\angle l_i l_{(u,v)}$ is minimized
47: $\quad$ **for** $j \leftarrow 1$ **to** γ **do**
48: $\quad\quad D \leftarrow \min\{D,$
$\quad\quad\quad\quad |N(u,i,j,L)u| + |uv| + |vN(v,i,j,R)|,$
$\quad\quad\quad\quad |N(u,i,j,R)u| + |uv| + |vN(v,i,j,L)|\}$
49: $\quad$ **end for**
50: $\quad$ **if** $D < d$ **then**
51: $\quad\quad u' \leftarrow u$ and $v' \leftarrow v$
52: $\quad\quad d \leftarrow D$
53: $\quad$ **end if**
54: **end for**
55: compute $\mathcal{T} \leftarrow \mathcal{T}_{u'v'}$
56: **return** $\mathcal{T}$

$O(\gamma^3 n + \gamma n \log n)$, see lines 1–18 in Algorithm 1. These lists will help us to compute an approximate farthest neighbor in L_{ij} and R_{ij} for each point $p \in P$ in time $O(\gamma)$, as will be described below.

Phase 2: Computing approximate farthest neighbors. Compute, for each point p, an approximate farthest neighbor in L_{ij} and an approximate farthest neighbor in R_{ij}, denoted $N(p, i, j, L)$ and $N(p, i, j, R)$ respectively. This can be done in time $O(\gamma)$ by using the lists L'_{ijk} and R'_{ijk}: just compute the distance between p and the first respectively the last point in each list. There are γ lists for each pair (i, j) and given that at most two entries in each list have to be checked, an approximate farthest neighbor can be computed in time $O(\gamma)$. Hence the total time complexity of this phase is $O(\gamma^3 n)$, as there are $O(\gamma^2 n)$ triples of type (p, i, j). The error we make by using approximate farthest neighbors is small:

Observation 2. *If p is any point in P, p_L the point in L_{ij} farthest from p and p_R the point in R_{ij} farthest from p, then (a) $|pp_L| \leq (1+\kappa/24) \cdot |pN(p,i,j,L)|$ and (b) $|pp_R| \leq (1+\kappa/24) \cdot |pN(p,i,j,R)|$.*

Proof. Due to symmetry it suffices to check (a). If the algorithm did not select p_L as farthest neighbor it holds that for each of the l_i-orderings there is a point further from p than p_L. Hence p_L must lie within a symmetric 2γ-gon whose edges are at distance $|pN(p,i,j,L)|$ from p. This implies that $|pN(p,i,j,L)| \geq |pp_L| \cos(\pi/(2\gamma)) \geq |pp_L|/(1+\kappa/24)$, using some basic calculus and $\kappa \leq 1/2$. □

Phase 3: Testing pole candidates. Compute the WSPD for P with separation constant τ. To be able to guarantee a $(1+\varepsilon)$-approximation algorithm the value of τ will depend on ε and κ as follows:

$$\tau = 8\left(\frac{1+\epsilon}{1+\epsilon-(1+\kappa)(1+\kappa/24)} - 1\right).$$

Note that the above formula implies that there is a trade-off between the values τ and κ, which can be used to fine-tune which part of the algorithm contributes how much to the uncertainty and to the running time. Setting for instance κ to 0.9ε yields for ε small $16/\varepsilon + 15 < \tau/8 < 32/\varepsilon + 31$, i.e. $\tau = \Theta(\frac{1}{\varepsilon})$. For each pair (A, B) in the decomposition we select two arbitrary points $u \in A$ and $v \in B$. Let $l_{(u,v)}$ be the line through u and v. Find the line l_i that minimizes the angle between l_i and $l_{(u,v)}$. That is, the line l_i is a close approximation of the direction of the line through u and v. From above we have that l_i is divided into $\gamma+1$ intervals of length $\frac{d_i}{\gamma+1}$. For each value of j, $1 \leq j \leq \gamma$, compute $\min(|N(u,i,j,L)u| + |uv| + |vN(v,i,j,R)|, |N(u,i,j,R)u| + |uv| + |vN(v,i,j,L)|)$. The smallest of these $O(\gamma)$ values is saved, and is a close approximation of the diameter of $\mathcal{T}_{uv}$, which will be shown below.

The number of pairs in the WSPD is $O(\tau^2 n)$, which implies that the total running time of the central loop of this phase (lines 42–54 in Algorithm 1) is $O(\gamma \cdot \tau^2 n)$. Building the WSPD and computing $\mathcal{T}_{u'v'}$ takes an extra $O(\tau^2 n + n \log n)$ time. Thus the whole algorithm runs in $O(\gamma^3 n + \gamma\tau^2 n + \gamma n \log n)$ time and uses $O(n \log n + \gamma^2 n + \tau^2 n)$ space. Setting

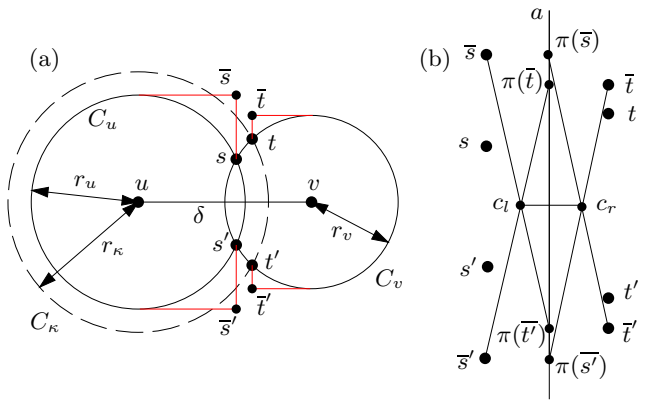

Figure 1: A valid cut.

$\kappa = 0.9\varepsilon$ yields $\gamma = O(\frac{1}{\varepsilon})$ and $\tau = O(\frac{1}{\varepsilon})$ and thus the time and space complexities we claimed. It remains to prove that the diameter of the dipolar tree that we compute is indeed at most $(1+\varepsilon) \cdot d_{\min}$.

5. THE PROOF OF CORRECTNESS

From Observation 1 we know that we will test a pair of poles u and v for which diam $\mathcal{T}_{uv} \leq (1 + 8/\tau)\ d_{\min} = \frac{1+\epsilon}{(1+\kappa)(1+\kappa/24)}\ d_{\min}$. The equality actually explains our choice of τ. In this section we will prove that our algorithm always computes a dipolar tree whose diameter is at most $(1+\kappa)(1+\kappa/24)$ diam $\mathcal{T}_{uv}$ and thus at most $(1+\epsilon)\ d_{\min}$.

Consider the tree $\mathcal{T}_{uv}$. For simplicity we rotate P such that the line l through u and v is horizontal and u lies to the left of v, as illustrated in Figure 1a. Let $\delta = |uv|$. Our aim is to prove that there exists an orthogonal cut that splits the point set P into two sets such that the tree obtained by connecting u to all points to the left of the cut and connecting v to all points to the right of the cut will give a tree whose diameter is a $(1+\kappa)$-approximation of diam $\mathcal{T}_{uv}$. Since the error introduced by approximating the farthest neighbor distances is not more than a factor of $(1+\kappa/24)$ according to Observation 2, this will prove the claim in the previous paragraph.

Denote by C_u and C_κ the circles with center at u and with radius r_u and $r_\kappa = r_u + \kappa z$ respectively, where $z =$ diam $\mathcal{T}_{uv} = \delta + r_u + r_v$. Denote by C_v the circle with center at v and with radius r_v. Let s and s' (t and t') be two points on C_u (C_v) such that if C_u (C_κ) and C_v intersect then s and s' (t and t') are the two intersection points, where s (t) lies above s' (t'). Otherwise, if C_u (C_κ) and C_v do not intersect, then $s = s'$ ($t = t'$) is the intersection of the line segment (u, v) and C_u (C_κ), see Figure 1a.

We say that a cut with a line l_κ is *valid* iff all points in P to the left of l_κ are contained in C_κ and all points of P to the right of l_κ are contained in C_v. A valid cut guarantees a dipolar tree whose diameter is at most $\delta + r_\kappa + r_v = (1+\kappa) \cdot$ diam $\mathcal{T}_{uv}$.

We will prove that the algorithm above always considers a valid cut. For simplicity we assume that $r_u \geq r_v$. We will show that there always exists a point b_{ij} on l_i such that cutting l_i orthogonally through b_{ij} is valid. Actually it is enough to show that the two requirements below are valid for any $\mathcal{T}_{uv}$. For a point p, denote the x-coordinate and the y-coordinate of p by $p.x$ and $p.y$, respectively. For simplicity we set $u = (0, 0)$.

$$\text{(i)} \qquad \frac{z}{\gamma+1} \cdot \frac{1}{\cos\frac{\pi}{2\gamma}} \leq \frac{1}{2}(t.x - s.x), \quad \text{and}$$

$$\text{(ii)} \qquad \tan\frac{\pi}{2\gamma} \leq \frac{t.x - s.x}{2(r_u + r_v)}.$$

The reason for this will now be explained. First we need to define some additional points. The reader is encouraged to study Figure 1 for a visual description. Let $\overline{s} = (s.x, r_u)$, $\overline{s}' = (s'.x, -r_u)$, $\overline{t} = (t.x, r_v)$ and $\overline{t}' = (t'.x, -r_v)$. Let a be the perpendicular bisector of the projections of s and t on the x-axis and let π be the orthogonal projection of the plane on a. Now we can define c_l to be the intersection point of the lines $(\overline{s}, \pi(\overline{t}'))$ and $(\overline{s}', \pi(\overline{t}))$, and c_r to be the intersection point of the lines $(\overline{t}, \pi(\overline{s}'))$ and $(\overline{t}', \pi(\overline{s}))$.

It now follows that any bisector l' that intersects the three line segments $(\overline{s}, \overline{t})$, (c_l, c_r) and $(\overline{s}', \overline{t}')$, will be a valid cut. This follows since all points to the left of l' will be connected to u and all points to the right of l' will be connected to v, and the diameter of that tree will, obviously, be bounded by $\delta + (r_u + \kappa z) + r_u$ which is a $(1+\kappa)$-approximation of diam $\mathcal{T}_{uv}$.

From the algorithm we know that (a) there is a line l_i such that $\angle(l_i, l_{(u,v)}) \leq \pi/(2\gamma)$, and that (b) there are γ orthogonal cuts of l_i that define equally many partitions of P. The distance between two adjacent orthogonal cuts of l_i is at most $z/(\gamma+1)$. This implies that the length of the largest interval on $l_{(u,v)}$ that is not intersected by any of these orthogonal cuts is at most

$$\frac{1}{\cos\frac{\pi}{2\gamma}} \cdot \frac{z}{\gamma+1}.$$

Hence requirement (i) ensures that for every $\mathcal{T}_{uv}$ the distance $|c_l c_r| = (t.x - s.x)/2$ must be large enough to guarantee that there is an orthogonal cut of l_i that intersects it.

An orthogonal cut of l_i has an angle of at least $\pi/2 - \pi/(2\gamma)$ to $l_{(u,v)}$. To ensure that an orthogonal cut of l_i that intersects the line segment $\overline{c_l c_r}$ also passes between $\overline{s}$ and $\overline{t}$ and between $\overline{s}'$ and $\overline{t}'$ it suffices to add requirement (ii).

It remains to prove the following lemma which implies that for every $\mathcal{T}_{uv}$ there is a valid orthogonal cut.

LEMMA 3. *For any $u, v \in P$ ($u \neq v$) the tree $\mathcal{T}_{uv}$ fulfills the requirements (i) and (ii).*

PROOF. The tree $\mathcal{T}_{uv}$ can be characterized by the relationship of the two ratios

$$\alpha := \frac{\delta}{r_u + r_v} \quad \text{and} \quad E := \frac{1 + \kappa/2}{1 - \kappa/2}.$$

We distinguish three cases: (1) $\alpha < 1$, (2) $1 \leq \alpha \leq E$, and (3) $\alpha > E$. For each of these three cases we will show that $\mathcal{T}_{uv}$ fulfills the two requirements.

Case 1: Using the following two straight-forward equalities, $s.x^2 + s.y^2 = r_u^2$ and $(\delta - s.x)^2 + s.y^2 = r_v^2$, we obtain that $s.x = (\delta^2 + r_u^2 - r_v^2)/(2\delta)$. A similar calculation for $t.x$ yields $t.x = (\delta^2 + r_\kappa^2 - r_v^2)/(2\delta)$. Inserting these values gives $t.x - s.x = (\kappa^2 z^2 + 2\kappa z r_u)/(2\delta)$. The fact that $\alpha \leq E$ allows us to further simplify the expression for $t.x - s.x$ by using the following two expressions:

$$\frac{z}{\delta} = \frac{\delta + r_u + r_v}{\delta} = 1 + \frac{r_u + r_v}{\delta} \geq \frac{2}{1 + \kappa/2}, \quad \text{and}$$

$$\frac{r_u}{\delta} \geq \frac{1 - \kappa/2}{2(1 + \kappa/2)}.$$

From this we obtain that

$$t.x - s.x = \frac{\kappa z}{2}\left(\frac{\kappa z}{\delta} + \frac{2r_u}{\delta}\right) > \frac{\kappa z}{2}.$$

This fulfills requirement (i) since

$$\frac{z}{\gamma + 1} \cdot \frac{1}{\cos\frac{\pi}{2\gamma}} \leq \frac{\kappa z}{4} \leq \frac{1}{2}(t.x - s.x). \tag{1}$$

For requirement (ii) note that $\tan\pi/(2\gamma) \leq 2\kappa \tan\pi/16 < 2\kappa/5$. Since $\kappa \leq 1/2$ we get that $z/\delta \geq 2/(1 + \kappa/2) \geq 8/5$. Combining this inequality, Equality 1, and our assumption that $r_u \geq r_v$ shows that requirement (ii) is also fulfilled:

$$\frac{t.x - s.x}{2(r_u + r_v)} \geq \frac{\kappa z}{4\delta}\left(\frac{2r_u + \kappa z}{r_u + r_v}\right) \geq \frac{\kappa z}{4\delta} \geq \frac{2\kappa}{5}.$$

Case 2: In this case we argue in the same manner as in the previous case. Using the fact that $s.x = r_u$ and $t.x = (\delta^2 + r_\kappa^2 - r_v^2)/(2\delta)$ yields

$$t.x - s.x \geq \frac{\kappa z}{2}\left(\frac{\kappa z}{\delta} + \frac{2r_u}{\delta}\right) > \frac{\kappa z}{2}.$$

The rest of the proof is exactly as in case 1.

Case 3: The first requirement is already shown to be fulfilled since $t.x - s.x \geq \delta - r_u - r_v \geq \kappa z/2$, hence it remains to show requirement (ii). We have

$$\frac{t.x - s.x}{2(r_u + r_v)} \geq \frac{\delta - (r_u + r_v)}{2(r_u + r_v)}$$

plugging in the values gives $\kappa/(2-\kappa)$, which is at least $2\kappa/5$. The lemma follows. □

The lemma says that for every dipole $\{u, v\}$ there exists a line a such that the dipolar tree obtained by connecting all the points on one side of a to u and all the points on the opposite side to v, is a $(1 + \kappa)$-approximation of $\mathcal{T}_{uv}$.

Conclusions

While it seems to be hard to reduce the running time for computing an exact MDST to $o(n^3)$, we have given a fast PTAS for approximating the MDST. It computes in $O(n \log n)$ time and space a tree whose diameter is at most $(1+\varepsilon)$ times that of a MDST. The runtime dependency on ε is moderate. The PTAS also works for higher dimensional point sets, but the running time increases exponentially with the dimension. A similar scheme yields an $O(n \log n)$-time PTAS for the discrete two-center problem.

The main novelty of our scheme is a discretization of the orientations of line segments to a constant number of orientations. We only consider projections of the input points onto those directions, which gives us fast access to farthest neighbors.

6. REFERENCES

[1] Paul B. Callahan and S. Rao Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *Journal of the ACM*, 42(1):67–90, January 1995.

[2] Joachim Gudmundsson, Herman Haverkort, Sang-Min Park, Chan-Su Shin, and Alexander Wolff. Approximating the geometric minimum-diameter spanning tree. Technical Report 4/2002, Institut für Mathematik und Informatik, Universität Greifswald, February 2002. Available at `http://www.math-inf.uni-greifswald.de/~awolff/pub/ghpsw-agmds-02t.pdf`.

[3] Refael Hassin and Arie Tamir. On the minimum diameter spanning tree problem. *Information Processing Letters*, 53(2):109–111, 1995.

[4] Jan-Ming Ho, D. T. Lee, Chia-Hsiang Chang, and C. K. Wong. Minimum diameter spanning trees and related problems. *SIAM Journal on Computing*, 20(5):987–997, October 1991.