Euclidean Minimum Spanning Trees and Bichromatic Closest Pairs*

Pankaj K. Agarwal†
Department of Computer Science, Duke University, Durham, NC 27706, USA

Herbert Edelsbrunner
Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois 61801, USA

Otfried Schwarzkopf
Emo Welzl
Institut für Informatik, Fachbereich Mathematik, Freie Universität Berlin, Arnimallee 2–6, D-1000 Berlin 33, West Germany

Abstract

We present an algorithm to compute a Euclidean minimum spanning tree of a given set S of n points in $\mathbb{E}^d$ in time $\mathcal{O}(\mathcal{T}_d(N,N)\log^d N)$, where $\mathcal{T}_d(n,m)$ is the time required to compute a bichromatic closest pair among n red and m blue points in $\mathbb{E}^d$. If $\mathcal{T}_d(N,N) = \Omega(N^{1+\varepsilon})$, for some fixed $\varepsilon > 0$, then the running time improves to $\mathcal{O}(\mathcal{T}_d(N,N))$. Furthermore, we describe a randomized algorithm to compute a bichromatic closest pair in expected time $\mathcal{O}((nm\log n\log m)^{2/3} + m\log^2 n + n\log^2 m)$ in $\mathbb{E}^3$, which yields an $\mathcal{O}(N^{4/3}\log^{4/3} N)$ expected time algorithm for computing a Euclidean minimum spanning tree of N points in $\mathbb{E}^3$.

1 Introduction

Given a set S of N points in Euclidean d-dimensional space $\mathbb{E}^d$, a *Euclidean minimum spanning tree* (EMST) is a spanning tree of S whose edges have a minimum total length among all spanning trees of S, where the length of an edge is the Euclidean distance between its vertices. For $d = 2$, an $\mathcal{O}(N\log N)$ algorithm for the computation of a EMST has been given by Shamos and Hoey [SH]. For $d \geq 3$, Yao [Ya] obtained $o(N^2)$ algorithms. In three dimensions, his algorithm runs in time $\mathcal{O}((N\log N)^{1.8})$, which can be reduced to $\mathcal{O}((N\log N)^{1.5})$ using results on the computation of Voronoi diagrams, see Section 5.1. Algorithms for computing approximate minimum spanning trees have been developed by Clarkson [Cl] and Vaidya [Va].

Our aim is to shed light on the relation between the EMST problem and the computation of *bichromatic closest pairs* (BCP). The latter problem can be formulated as follows: Given a set of n red and m blue points in $\mathbb{E}^d$, find a red point p and a blue point q such that the distance between p and q is minimum among all red-blue pairs.

It is not difficult to verify that a EMST of the union of the red and blue points contains at least one closest red-blue pair. It is thus possible to solve the BCP problem by computing a EMST. The first result of this paper is to show that the converse is also true. We present an algorithm that computes a EMST by solving several BCP problems. If we can find a BCP for n red and m blue points in $\mathbb{E}^d$ in time $\mathcal{T}_d(n,m)$, then we can compute a EMST in $\mathbb{E}^d$ in time $\mathcal{O}(\mathcal{T}_d(N,N)\log^d N)$. Moreover, if $\mathcal{T}_d(N,N) = \Omega(N^{1+\varepsilon})$ for some $\varepsilon > 0$, the time to compute a EMST is only $\mathcal{O}(\mathcal{T}_d(N,N))$.

Most current EMST algorithms start by computing a set of edges which can be shown to be a superset of the edge set of a EMST. In the two-dimensional case the set of edges of the Delaunay triangulation is a good choice for this superset. Unfortunately, already in three dimensions the edge set of the Delaunay triangulation can be the complete graph. Another possible choice for a suitable superset is due to Yao. He divides the set of all possible edges into a constant number of groups,

*The first, second, and fourth authors acknowledge support from the Center for Discrete Mathematics and Theoretical Computer Science (DIMACS), a National Science Foundation Science and Technology Center under NSF grant STC 88-09648. The second author's work was supported by the National Science Foundation under grant CCR-8714565. The third author's work was supported by the Deutsche Forschungsgemeinschaft under grant Al 253/1-2, Schwerpunktprogramm "Datenstrukturen und effiziente Algorithmen". The last two authors' work was also partially supported by the ESPRIT II Basic Research Actions Program of the EC under contract no. 3075 (project ALCOM).

†*current address:* DIMACS Center, Hill Center — Busch Campus, P.O. Box 1179, Rutgers University, Piscataway, NJ 08855–1179, USA.

according to the slope of the edges, and selects a linear number of edges from each group. Our algorithm is based on a similar idea. We classify all possible edges into $\mathcal{O}(N \log^{d-1} N)$ groups, each group forming a complete bipartite subgraph on two subsets, P and Q, of S. We are then able to show that for each group only a closest pair between P and Q can form an edge of any EMST.

We then turn our attention to the BCP problem. In three dimensions, we give a randomized algorithm that computes a bichromatic closest pair of n red and m blue points in expected time $\mathcal{O}((nm \log n \log m)^{2/3} + m \log^2 n + n \log^2 m)$. This implies an $\mathcal{O}(N^{4/3} \log^{4/3} N)$ randomized expected time algorithm for the EMST problem in $\mathbb{E}^3$.

2 Conventions

We do not distinguish between points and vectors in $\mathbb{E}^d$; hence, we can add and subtract points. The *scalar product* of $x = (x_1, \ldots, x_d)$ and $y = (y_1, \ldots, y_d)$ is $x^T y = \sum_{i=1}^d x_i y_i$, the *norm* of x is $||x|| = \sqrt{x^T x}$, and the *angle* between x and y is $\angle(x, y) = \cos^{-1} \frac{x^T y}{||x|| \cdot ||y||}$. Furthermore, the *angle* $\angle(xyz)$ defined by the three points x, y, and z is defined as $\angle(xyz) = \angle(z - y, x - y)$.

For $A, B \subseteq \mathbb{E}^d$ define $A + B = \{x + y \mid x \in A, y \in B\}$, and let $x + B = \{x\} + B$. The *Euclidean distance* between x and y is $d(x, y) = ||y - x||$. For a finite point set A, define $d(x, A) = \min_{y \in A} d(x, y)$, and for two finite point sets A and B, define $d(A, B) = \min_{x \in A} d(x, B)$. For two points x and y we let xy be the line segment connecting them; the *length* of xy is $d(x, y)$.

A *closest* (A, B)*-pair* is a pair (x, y) with $x \in A$, $y \in B$ and $d(x, y) = d(A, B)$. We define $diam(A) = \max_{x, y \in A} d(x, y)$.

3 Geometric results

As mentioned above, we intend to classify the set of pairs of points in S into several groups. From each group we will select only one pair as a possible edge for a EMST. In this section, we define the groups and show the geometric result.

First, we introduce some notation. Let $d \in \mathbb{E}^d$ be a vector of unit length, $||d|| = 1$, indicating a direction in d-dimensional space, and let $\alpha < 90^\circ$ be an angle. We define the cone $Cone(d, \alpha) = \{x \in \mathbb{E}^d \mid \angle(x, d) \leq \alpha\}$.

Let α_0 be the largest angle so that for any $0 < \alpha < \alpha_0$ we have

$$\tan 2\alpha < \cos 2\alpha$$

($\alpha_0 = (\arcsin \frac{\sqrt{5}-1}{2})/2$ which is about 19.08°). In the following we will assume that α is fixed with $0 < \alpha < \alpha_0$.

Let S be a finite set of points in $\mathbb{E}^d$ for which we want to compute a EMST. For two disjoint subsets P and Q of S we call (P, Q) a *strongly separated* pair if

$$\max\{diam(P), diam(Q)\} < d(P, Q).$$

Furthermore, we call (P, Q) *α-separated* if there exists a point z and a direction vector d such that $P \subseteq z + Cone(-d, \alpha)$ and $Q \subseteq z + Cone(d, \alpha)$. When d, the orientation of the cones, is important we will call (P, Q) α-separated *in direction d*.

The strongly separated pairs are our first means to reduce the number of candidate edges for a EMST.

Lemma 1 *If (P, Q), $P, Q \subseteq S$, is a strongly separated pair, then any EMST of S contains at most one edge between P and Q.*

Proof. Indeed if lemma were not true, then there is exists a EMST in which we can find two edges $\{p, q\}$ and $\{p', q'\}$, for $p, p' \in P$ and $q, q' \in Q$, between P and Q, such that the points q, q' lie on the path from p to p' in the EMST, or vice versa. But, in that case we can obtain a shorter spanning tree of S by adding the edge $\{p, p'\}$ and removing the edge $\{p, q\}$, a contradiction. Hence the lemma is true. ◻

We obtain strongly separated pairs from α-separated pairs. Let (P, Q), $P, Q \subseteq S$, be α-separated in direction d. We call $p \in P$ *extremal* if $p + Cone(d, \frac{\pi}{2} - \alpha)$ contains no element of P. Analogously, we call $q \in Q$ *extremal* if $q + Cone(-d, \frac{\pi}{2} - \alpha)$ contains no element of Q. We denote the subsets of extremal elements by P' and Q'.

Lemma 2 *Let (P, Q), $P, Q \subseteq S$, be an α-separated pair, and let P', Q' be the subsets of extremal elements. If $\{p, q\}$, $p \in P$, $q \in Q$, is an edge of some EMST of S, then $p \in P'$, $q \in Q'$ and $\{p, q\}$ is a bichromatic closest pair.*

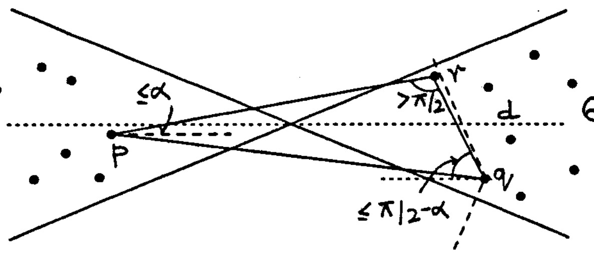

Figure 1: Illustration of Lemma 2

Proof. Here we use a result of Yao [Ya] who proves that if $\{p, q\}$ is an edge of a EMST of S and $q \in p +$

$Cone(d,\beta)$, for $\beta \leq \frac{\pi}{6}$, then q is a nearest neighbor of p in the cone $p + Cone(d,\beta)$. Since $\alpha < \frac{\pi}{6}$, q is the nearest neighbor of p in Q. Thus it suffices to show that $q \in Q'$. Indeed if q were not extremal, then there exists a point $r \in q + Cone(-d, \frac{\pi}{2} - \alpha)$. Consider the triangle $\triangle prq$. We have $\angle(prq) = \pi - \angle(r-p, q-r)$, $\angle(d, r-p) \leq \alpha$ and $\angle(d, q-r) \leq \frac{\pi}{2} - \alpha$. Since

$$\angle(r-p, q-r) \leq \angle(d, r-p) + \angle(d, q-r) \leq \frac{\pi}{2},$$

$\angle(prq) \geq \frac{\pi}{2}$. Hence, pq is the longest side of prq which implies $d(p,q) > d(p,r)$, a contradiction.

We conclude that q must be extremal in Q. By symmetry, we have $p \in P'$ and p is the closest neighbor of q in P. ◻

So we can content ourselves with the pair (P', Q'), and fortunately we have the following result.

Lemma 3 *If (P', Q') be an α-separated pair of extremal elements, then (P', Q') is strongly separated.*

Proof. Let $z \in \mathbb{E}^d$ be such that $Q' \subseteq z + Cone(d, \alpha)$. Let $q, r \in Q'$ and consider the triangle zqr. Define $\phi = \angle(qzr)$ and $\psi = \angle(rqz)$, $\omega = \angle(qrz)$ and let a, b, and c be the lengths of zq, zr, and qr. Next we derive bounds on the sine functions of the angles. It is easily seen that $\phi \leq 2\alpha$ and $\frac{\pi}{2} - 2\alpha \leq \psi, \omega \leq \frac{\pi}{2} + 2\alpha$, and therefore $\sin\phi \leq \sin 2\alpha$, $\sin\psi \geq \cos 2\alpha$ and $\sin\omega \geq \cos 2\alpha$.

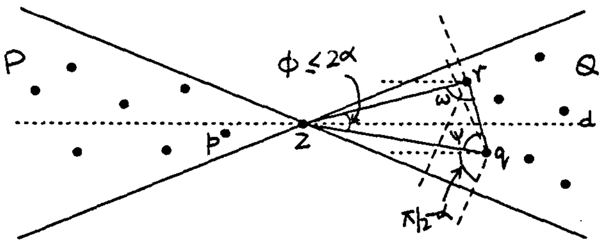

Figure 2: Illustration of Lemma 3

The equality

$$\frac{a}{\sin\omega} = \frac{b}{\sin\psi} = \frac{c}{\sin\phi}$$

for the triangle zqr implies

$$c = b\frac{\sin\phi}{\sin\psi} \leq b\frac{\sin 2\alpha}{\cos 2\alpha} = b\tan 2\alpha \qquad (1)$$

$$b = a\frac{\sin\psi}{\sin\omega} \leq a\frac{1}{\sin\omega} \leq a\frac{1}{\cos 2\alpha}. \qquad (2)$$

Let $q_0 \in Q'$ be a nearest neighbor of z, that is, $d(z, q_0) = d(z, Q')$. By (2) for every $q \in Q'$, we have

$$d(z,q) \leq d(z,q_0)\frac{1}{\cos 2\alpha} = d(z,Q')\frac{1}{\cos 2\alpha}. \qquad (3)$$

Now let q', q'' be any pair of points in Q'. It follows from (1) and (3) that

$$d(q',q'') \leq d(z,q')\tan 2\alpha \leq d(z,Q')\frac{\tan 2\alpha}{\cos 2\alpha} < d(z,Q')$$

as $\alpha < \alpha_0$. By symmetry, we have $d(p',p'') < d(z,P')$ for any two points $p', p'' \in P'$. We thus have $\max\{diam(P'), diam(Q')\} < \max\{d(z,P'), d(z,Q')\}$.

Consider now $p \in P$, $q \in Q$. The angle $\angle(pzq) \geq \pi - 2\alpha > \pi/2$, so pq is the longest side of the triangle pzq. This implies $d(p,q) > \max\{diam(P'), diam(Q')\}$, which is equivalent to saying that P and Q are strongly separated. ◻

Now we can give our central result which reduces the EMST problem to the BCP problem.

Lemma 4 *Let S be a set of points in $\mathbb{E}^d$ and let $\mathcal{B}$ be a set of α-separated pairs, for some $0 < \alpha < \alpha_0$, with the property that for any pair of points $p, q \in S$ there exists a pair $(P,Q) \in \mathcal{B}$ such that $p \in P$ and $q \in Q$. If a set $M \subset P \times Q$ contains a closest (P,Q)-pair for every $(P,Q) \in \mathcal{B}$, then M contains a EMST of S.*

Proof. Consider the set $\mathcal{B}' = \{(P',Q') \mid (P,Q) \in \mathcal{B}$ and P', Q' are the sets of extremal elements in P and $Q\}$. Let E be the set of edges $E = \{\{p',q'\} \mid p' \in P', q' \in Q', (P',Q') \in \mathcal{B}'\}$. By Lemma 2, E contains every EMST of S. Since P', Q' are strongly separated (cf. Lemma 3), there is at most one edge between P' and Q' in any EMST of S, namely the one connecting the closest pair of points (cf. Lemma 1 and 2). Furthermore, using the same argument as in Lemma 2, one can show that a closest pair of (P,Q) is also a closest pair of (P',Q'). Hence, M contains a EMST of S. ◻

4 An algorithm to reduce EMST to BCP

We now describe an algorithm that solves the EMST problem in d dimensions by solving several instances of the BCP problem, assuming we are given an algorithm for the BCP problem. Let this algorithm take $\mathcal{T}_d(n,m)$ time for a set of n red and m blue points. As usual we let S be a set of N points and we wish to compute a EMST of S.

We borrow some notation from Yao [Ya]. Let $B = \{b_1, \ldots, b_d\}$ be a basis of $\mathbb{E}^d$. The *convex cone* of B is $Conv(B) = \{\sum_{i=1}^{d} \lambda_i b_i \mid \lambda_i \geq 0 \ \forall i\}$. We call $Conv(B)$ *narrow* if there exists a vector $d \in \mathbb{E}^d$, $||d|| = 1$, such that $Conv(B) \subseteq Cone(d,\alpha)$ with $\alpha < \alpha_0$. Let $\mathcal{F}$ be a finite family of bases of $\mathbb{E}^d$. We call $\mathcal{F}$ a *frame* of $\mathbb{E}^d$ if $\bigcup_{B\in\mathcal{F}}(Conv(B) \cup -Conv(B)) = \mathbb{E}^d$. A frame $\mathcal{F}$ is called *narrow* if every $B \in \mathcal{F}$ is narrow. By Lemma 4.2 of Yao [Ya], for every dimension d one can construct a narrow frame in a finite number of steps.

We compute such a narrow frame $\mathcal{F}$ and successively consider the bases $B \in \mathcal{F}$. For each $B \in \mathcal{F}$, we compute a set $\mathcal{B}_B$ of α-separated pairs using an algorithm inspired by the range tree structure for d-dimensional point sets (see e.g. [PS]).

Let $B = \{b_1, \ldots, b_d\}$ be the basis for which we compute α-separated pairs, and let $(x_1, \ldots, x_d)$ be the coordinates of a point x in this basis, that is, $x = x_1 b_1 + \ldots + x_d b_d$. We assume that B is such that no two points of S share a coordinate. The algorithm that computes the α-separated pairs is recursive, and each recursive call either reduces the number of points considered or the *dimensionality* of the problem. The α-separated pairs are output when the dimensionality k is 0. The input parameters of the algorithm are k, the dimensionality, P, a set of red points, and Q, a set of blue points in $\mathbb{E}^d$. Initially, $k = d$ and $P = Q = S$.

```
1 If $k = 0$ then output $(P, Q)$ as an $\alpha$-separated pair.
2 Otherwise (if $k \geq 1$) execute the following steps.
    2.1 Compute $x_k$, the median of the $k$th coordinate of points in $P \cup Q$.
    2.2 Set $P_\ell = \{p \in P \mid p_k \leq x_k\}$, $P_r = \{p \in P \mid p_k > x_k\}$, $Q_\ell = \{q \in Q \mid q_k \leq x_k\}$, and $Q_r = \{q \in Q \mid q_k > x_k\}$.
    2.3 If $P_\ell \neq \emptyset$ and $Q_r \neq \emptyset$ then recurse with parameters $k-1$, $P_\ell$, and $Q_r$.
    2.4 If $P_\ell \neq \emptyset$ and $Q_\ell \neq \emptyset$ then recurse with $k$, $P_\ell$, and $Q_\ell$.
    2.5 If $P_r \neq \emptyset$ and $Q_r \neq \emptyset$ then recurse with $k$, $P_r$, and $Q_r$.
```

Since B is narrow, every pair (P, Q) returned by the algorithm is α-separated. We need to show that if $q \in p + Conv(B)$ then the algorithm outputs a pair (P, Q) with $p \in P$ and $q \in Q$. Assume inductively that this is true for dimension $d-1$. To prove it for dimension d note that $q \in p + Conv(B)$ is equivalent to $p_i < q_i$ for $1 \leq i \leq d$. In particular, it implies $p_d < q_d$. Thus, there will be a call of the algorithm so that $k = d$, $p \in P_\ell$, and $q \in Q_r$. Step 2 calls the algorithm for $k = d-1$, $P = P_\ell$, and $Q = Q_r$, and by inductive assumption this call produces the desired α-separated pair. Let $\mathcal{B}_B$ be the set of α-separated pairs produced by the algorithm. From what we just said it follows that $\mathcal{B} = \bigcup_{B \in \mathcal{F}} \mathcal{B}_B$ fulfills the requirements of Lemma 4.

It therefore suffices to compute for every pair $(P, Q) \in \mathcal{B}$ a closest (P, Q)-pair to obtain a suitable set M. We claim that the size of M is $\mathcal{O}(N \log^{d-1} N)$ and verify this by counting the number of α-separated pairs generated by the algorithm when it is called for $k = d$, $P = Q = S$, and basis B. Let this number be $t_k(|P| + |Q|)$; so we are interested in $t_d(2N)$. Clearly, $t_0(n + m) = 1$. For higher indices k we have

$$t_k(n+m) \leq 2t_k\left(\frac{n+m}{2}\right) + t_{k-1}(n+m)$$

which solves to $t_k(n+m) = \mathcal{O}((n+m)\log^{d-1}(n+m))$. It follows that $t_d(2N) = \mathcal{O}(N \log^{d-1} N)$ as claimed. From M we can compute a EMST of S in time $\mathcal{O}(N \log^{d-1} N)$.

It remains to analyze the computation of closest pairs. In order to get closest pairs as output we just replace step 1 with

```
1' If $k = 0$ then find a closest $(P, Q)$-pair $\{p, q\}$ and output it.
```

Recall that $\mathcal{T}_d(n, m)$ is an upper bound on the time it takes to compute $\{p, q\}$ if $n = |P|$ and $m = |Q|$. Let $T_d^k(n+m)$ be the running time of the above algorithm for dimensionality k, and for sets P of size n and Q of size m in $\mathbb{E}^d$. We have $T_d^0(n+m) = \mathcal{T}_d(n, m)$, and assuming $\mathcal{T}_d(n, m) = \Omega(n+m)$ we have

$$T_d^k(n+m) = 2T_d^k\left(\frac{n+m}{2}\right) + T_d^{k-1}(n+m)$$

for $k \geq 1$. Without further assumptions we get $T_d^d(2N) = \mathcal{O}(\mathcal{T}_d(N, N) \log^d N)$. However, if $\mathcal{T}_d(n, m) = \Omega((n+m)^{1+\varepsilon})$, for some fixed $\varepsilon > 0$, then $T_d^d(2N) = \mathcal{O}(\mathcal{T}_d(N, N))$. We summarize the results of this section.

Theorem 5 *Let $\mathcal{T}_d(n, m)$ be the time required to compute a bichromatic closest pair for n red and m blue points in $\mathbb{E}^d$. If $\mathcal{T}_d(n, m) = \Omega(n+m)$ then a Euclidean minimum spanning tree of N points in $\mathbb{E}^d$ can be computed in time $\mathcal{O}(\mathcal{T}_d(N, N) \log^d N)$. If furthermore $\mathcal{T}_d(n, m) = \Omega((n+m)^{1+\varepsilon})$, for $\varepsilon > 0$, then $\mathcal{O}(\mathcal{T}_d(N, N))$ time suffices to compute a Euclidean minimum spanning tree.*

5 Computing bichromatic closest pair

In this section we present a fast randomized algorithm for the three-dimensional BCP problem: Given a set P of n red points in $\mathbb{E}^3$ and another set Q of m blue points in $\mathbb{E}^3$, determine a pair of points $p \in P$ and $q \in Q$ such that $d(p, q) = d(P, Q)$.

This problem can be obviously solved in time $\mathcal{O}(mn)$ by trying all red-blue pairs of points. The goal of this section is to develop a significantly faster algorithm. The main result is a randomized algorithm whose expected running time is $\mathcal{O}((nm \log n \log m)^{2/3} + m \log^2 n + n \log^2 m)$.

5.1 BCP for unbalanced point sets

We start by considering the case when the number of red points is much smaller than the number of blue points. Our algorithm uses ideas from [Se, CS, ESh]. We incrementally construct a triangulation of the Voronoi diagram of the red points. More specifically, we use what we call the *bottom-vertex-triangulation (bv-triangulation)* of the Voronoi diagram defined as follows. For every *bounded* face of the Voronoi diagram choose the (lexicographically) smallest vertex v and form triangles with v and every edge of the face that is not incident to v; we *do not* triangulate unbounded faces and cells. In the same manner, choose the (lexicographically) smallest vertex v for every bounded cell and form a tetrahedron with v and every triangle on the cell's faces (which are now triangulated, since they are all bounded).

Let $Vor(S)$ denote the Voronoi diagram of S and let $bv\text{-}Vor(S)$ be its bv-triangulation. The bv-triangulation has the nice property that it is unique and it is completely determined locally. In fact, four vertices of a bounded Voronoi cell of $Vor(S)$ form a tetrahedron Δ of $bv\text{-}Vor(S)$ if and only if Δ is a tetrahedron of $bv\text{-}Vor(T)$, where $T \subseteq S$ is the set of at most 10 points defining the four vertices of Δ. (We arrive at 10 points because four points in S define a vertex of the Voronoi diagram, but six of the 16 points are duplicates.)

To make the above remark more formal we introduce a few definitions. For a finite point set $S \subseteq \mathbb{E}^3$ let $\mathcal{T}_S$ be the set of tetrahedra Δ in $bv\text{-}Vor(T)$, for all $T \subseteq S$ with $|T| \leq 10$. Because we assume that S is in general position, the subset T defining a tetrahedron Δ is unique and denoted by T_Δ. For $\Delta \in \mathcal{T}_S$, let p_Δ be the point in T_Δ that generates the Voronoi cell containing Δ, and let

$$reg(\Delta) = \{x \in \mathbb{E}^3 \mid \exists y \in \Delta,\ d(x,y) < d(y, p_\Delta)\}.$$

We have the following result which we state without proof.

Lemma 6 *A tetrahedron $\Delta \in bv\text{-}Vor(S)$ if and only if $\Delta \in \mathcal{T}_S$ and $reg(\Delta) \cap S = \emptyset$.*

Before we proceed to the algorithm we need to discuss the unbounded cells of $Vor(S)$, as they pose a slight problem when it comes to triangulating $Vor(S)$. We decided not to triangulate an unbounded cell because its bottom vertex may not be defined. To cope with the thus arising difficulties, we introduce a set U of four points forming a sufficiently large tetrahedron, where *sufficiently large* means that

(i) the convex hull of $P \cup Q \cup U$ is the tetrahedron U, and

(ii) for any two points $x, y \in P \cup Q$ we have $d(x,y) < d(x,U)$.

Property (i) ensures that for any $P' \subseteq P$, the only unbounded cells of $Vor(P' \cup U)$ are those generated by the four points in U. On the other hand Property (ii) implies that if $x \in P'$ and $y \in P \cup Q$ then y lies in a bounded cell of $Vor(P' \cup U)$. This will be convenient later when we perform point location queries in $Vor(P' \cup U)$.

We now have the tools ready to give the algorithm that proves the following result.

Lemma 7 *The BCP problem for a set P of n red and a set Q of m blue points in $\mathbb{E}^3$ can be solved in randomized expected time $O(n^2 + m \log^2 n)$.*

Proof. As in [CS], we construct $bv\text{-}Vor(P)$ by incrementally adding one point after the other in some random order $p_1, \ldots, p_n$. However, we maintain not only the Voronoi diagram, but also its bv-triangulation during the process. We start with $bv\text{-}Vor(U \cup \{p_1\})$. When a new point p_i is added to $bv\text{-}Vor(U \cup \{p_1, \ldots, p_{i-1}\})$, we first find, in $O(i)$ time, the nearest neighbor of p_i in $p_1, \ldots, p_{i-1}$ by exhaustive search; p_i is contained in the Voronoi cell of this nearest point. By spending another $O(i)$ time we can find the tetrahedron Δ in this cell that contains p_i. The total time for this step over all points is thus $\sum_{i=1}^{n} O(i) = O(n^2)$.

We then compute the set $\mathcal{D}$ of all tetrahedra that have to be deleted when we add p_i. Recall that a tetrahedron Δ has to be deleted from $bv\text{-}Vor(U \cup \{p_1, \ldots, p_{i-1}\})$ if and only if $p_i \in reg(\Delta)$, that is, Δ intersects the half-space formed by the bisecting plane of p_i and p_Δ, and containing the point p_i. Notice that it suffices to check the four vertices of Δ. Clearly, $\mathcal{D}$ is exactly the set of tetrahedra intersecting the Voronoi cell of p_i in $bv\text{-}Vor(U \cup \{p_1, \ldots, p_i\})$. Hence, $\mathcal{D}$ is connected in the sense that for any two tetrahedra Δ_1 and Δ_2 in $\mathcal{D}$ there is a sequence of tetrahedra in $\mathcal{D}$ starting with Δ_1 and ending with Δ_2 so that any two adjacent tetrahedra share a face. It follows that $\mathcal{D}$ can be found using a graph search algorithm (such as depth first search) starting at the tetrahedron that contains p_i; this takes time $O(|\mathcal{D}|)$.

We insert the Voronoi cell of p_i, triangulate it and complete the triangulations of all adjacent cells. All this can be accomplished in time $O(|\mathcal{D}|)$ because a Voronoi cell gets a new bottom vertex only if all tetrahedra sharing the old bottom vertex are in $\mathcal{D}$. Clearly, the total number of tetrahedra deleted in the course of the algorithm is not larger than the total number of tetrahedra created. Using Seidel's backwards-analysis [Se2], we find that the average number of tetrahedra created when inserting p_i is $O(i)$. To see this consider a tetrahedron Δ that exists in *bv-Vor*$(U \cup \{p_1, \ldots, p_i\})$ and notice that it ceases to exist in *bv-Vor*$(U \cup \{p_1, \ldots, p_{i-1}\})$ if and only if the removed point p_i is one of the $|T_\Delta|$ points defining Δ. The probability that this happens is at most $\frac{10}{i}$. Since there are $\mathcal{O}(i^2)$ tetrahedra in *bv-Vor*$(U \cup \{p_1, \ldots, p_i\})$, the expected number of tetrahedra ceasing to exist is $\mathcal{O}(i)$.

We can thus bound the expected total number of tetrahedra created, and thus the expected running time of the algorithm, by $\mathcal{O}(n^2)$.

The set of tetrahedra created during the algorithm is used to create a point location structure as follows. For each Δ, we store the point p_i that causes the deletion of Δ. In addition, for every Voronoi cell changed by p_i we store a two-dimensional point location structure [EGS, ST] so that if a point lies in a newly created tetrahedron we can find it in time $\mathcal{O}(\log n)$. These additional structures can be computed within the above time bounds.

Now we locate every point $q \in Q$ in *bv-Vor*$(U \cup P)$. We thus find for every blue point $q \in Q$ a nearest red point. This suffices to solve the BCP problem.

The point location is done by following the pointers established before, starting at the Voronoi cell of p_1 in *bv-Vor*$(U \cup \{p_1\})$, which happens to be a tetrahedron. For every Δ with $q \in \Delta$ we look up the point p_i causing the deletion of Δ and determine whether or not q falls into the cell of p_i. In either case, we find the appropriate new tetrahedron Δ' in time $\mathcal{O}(\log n)$ using the established point location structures. It remains to bound the expected number of tetrahedra visited during the search. Again, we use backwards-analysis. The probability that the tetrahedron $\Delta \in$ *bv-Vor*$(U \cup \{p_1, \ldots, p_i\})$ containing q ceases to exist in *bv-Vor*$(U \cup \{p_1, \ldots, p_{i-1}\})$ is $\mathcal{O}(\frac{1}{i})$ by the argument above. This implies that the expected length of the search chain is $\mathcal{O}(\sum_{i=1}^{n} \frac{1}{i}) = \mathcal{O}(\log n)$. We thus obtain a search time of $\mathcal{O}(\log^2 n)$ per point in Q, which completes the proof of the lemma. □

Remark: Actually the above BCP problem can be solved deterministically in time $O(n^2 + m \log^2 n)$, using the convex hull algorithm of Seidel [Se] and the spatial point location algorithm of Preparata and Tamassia [PT].

The running time of the above algorithm can be improved by dividing P into $t = \lceil n/(\sqrt{m} \log n) \rceil$ subsets $P_1, \ldots, P_t$ of size at most $\lceil n/t \rceil$ each and then solving the problem with the above method for every pair (P_i, Q). This results in an expected running time of $\mathcal{O}\left(t \cdot (\frac{n}{t})^2 + t \cdot m \log^2 n\right) = \mathcal{O}\left(n\sqrt{m} \log n + m \log^2 n\right)$. We thus have the following result.

Corollary 8 *The BCP problem for a set of n red and a set of m blue points in $\mathbb{E}^3$ can be solved in randomized expected time $\mathcal{O}(n\sqrt{m} \log n + m \log^2 n)$.*

5.2 BCP for balanced point sets

When n and m are of about the same size we use a technique similar to that of [CEGSW], combined with ideas of [CS]. We take a random sample of the blue points and use it to decompose the problem into many small problems. The idea relies on the fact that the expected size of the subproblems will be unbalanced enough so that we can use the algorithm of the previous section.

The description of the algorithm follows. Let again P be a set of n red points and Q a set of m blue points in $\mathbb{E}^3$. We assume that $n \leq m$; otherwise, we swap colors.

If $n/\log n \leq \sqrt{m}$, a bichromatic closest pair can be computed, in time $\mathcal{O}(n^2 + m \log^2 n) = \mathcal{O}(m \log^2 n)$, using the algorithm described in the proof of Lemma 7.

Otherwise, i.e. $\frac{n}{\log n} > \sqrt{m}$, take a random sample R of r blue points (r will be specified below). We compute, in time $O(r^2)$, the Voronoi diagram of R and its bv-triangulation $\mathcal{T} =$ *bv-Vor*$(U \cup R)$, where U satisfies the conditions (i) and (ii) specified above. $\mathcal{T}$ is a cell complex of $\mathcal{O}(r^2)$ tetrahedra. For every $\Delta \in \mathcal{T}$, we determine a set P_Δ of red points and a set Q_Δ of blue points defined as follows. $P_\Delta = P \cap \Delta$, that is, P_Δ is the set of all red points $p \in P$ contained in Δ, and $Q_\Delta = Q \cap reg(\Delta)$, that is, a blue point $q \in Q$ is in Q_Δ if and only if there exists a point $y \in \Delta$ with $d(y, q) \leq d(y, q_\Delta)$.

It is easy to compute P_Δ for all $\Delta \in \mathcal{T}$ in time $O(n \log^2 r)$ by performing a point location for every red point. If n_Δ is the number of points in P_Δ, we have $\sum_{\Delta \in \mathcal{T}} n_\Delta = n$.

Now observe that the set of tetrahedra Δ with Q_Δ containing a fixed blue point q is exactly the set of all tetrahedra that would be deleted if q were inserted into *bv-Vor*$(U \cup R)$. As in the last section, we can therefore use a point location query and a graph search to identify all $\Delta \in \mathcal{T}$ with $q \in Q_\Delta$. The total running time for this procedure is $\mathcal{O}(\sum_{\Delta \in \mathcal{T}} m_\Delta)$, where m_Δ is the cardinality of Q_Δ.

Finally, we use the algorithm of Corollary 8 to find the closest (P_Δ, Q_Δ)-pair for every tetrahedron Δ. We output the pair with the shortest distance as a closest (P, Q)-pair.

If we set $r = \left\lceil n^{2/3}/(m^{1/3} \log^{2/3} m) \right\rceil$, we can prove the following result.

Theorem 9 *The algorithm above computes a bichromatic closest pair of a set of n red and a set of m blue* *points in $\mathbb{E}^3$ in randomized expected time*

$$\mathcal{T}_d(n,m) = \mathcal{O}\left((nm\log nm)^{2/3} + m\log^2 n + n\log^2 m\right).$$

Proof. The correctness of the algorithm is based on the observation that q is in Q_Δ if $\{p,q\}$ is a closest (P,Q)-pair and Δ is the tetrahedron containing p.

The running time of the algorithm is $\mathcal{O}(m\log^2 n)$ if $\frac{n}{\log n} \le \sqrt{m}$. Otherwise, the running time is

$$\mathcal{O}\left(\sum_{\Delta\in\mathcal{T}} (n_\Delta\sqrt{m_\Delta}\log n_\Delta + m_\Delta \log n_\Delta) + r^2 + n\log^2 r\right),$$

which is bounded by

$$\mathcal{O}\left(\log n \sum_{\Delta\in\mathcal{T}} n_\Delta\sqrt{m_\Delta} + \log^2 n \sum_{\Delta\in\mathcal{T}} m_\Delta + m\log^2 n\right).$$

We show below that the expected value of $\sum_{\mathcal{T}} n_\Delta\sqrt{m_\Delta}$ is $\mathcal{O}(n\sqrt{\frac{m}{r}})$ and the expected value of $\sum_{\mathcal{T}} m_\Delta$ is $\mathcal{O}(mr)$. For the given choice for r, the expected running time is thus

$$\mathcal{O}(n^{2/3}m^{2/3}\log^{4/3} n + m\log^2 n).$$

Without the assumption $n \le m$ we obtain the symmetric bound on the running time given in the theorem. ⊡

It is interesting to compare the time bound of the three-dimensional BCP algorithm with the currently best upper bound on the number of bichromatic minimum distance pairs which is $\mathcal{O}(n^{2/3}m^{2/3} + n + m)$ [ESh]. To complete the proof of the theorem we still need to establish the claimed expectations. Following [CEGSW] and [CS] we start with an elementary lemma.

Lemma 10 *At most 10 tetrahedra Δ in $\mathcal{T}_{U\cup R}$ contain some fixed point $p \in P$ and have $|reg(\Delta)\cap R| = 1$.*

Proof. For every set $R' \subseteq R$ there is a unique tetrahedron $\Delta_{R'} \in \mathcal{T}_{U\cup R'}$ with $p \in \Delta_{R'}$ and $reg(\Delta_{R'})\cap R' = \emptyset$. Now consider a tetrahedron $\Delta \in \mathcal{T}_{U\cup R}$ that contains p and $|reg(\Delta)\cap R| = 1$. Let $q = reg(\Delta)\cap R$. Clearly, $\Delta = \Delta_{R\setminus\{q\}}$. The result follows since $\Delta_{R\setminus\{q\}} = \Delta_R$ if $q \notin T_{\Delta_R}$ (that is, q is not one of the points defining Δ_R), and since $|T_{\Delta_R}| \le 10$. ⊡

With this result we can now go ahead and prove the claimed expectations.

Lemma 11 *If $R \subset Q$ is chosen at random, we have*

$$\begin{aligned}\mathsf{E}[\sum_{t\in\mathcal{T}} n_t\sqrt{m_t}] &= \mathcal{O}\left(n\sqrt{\frac{m}{r}}\right) \quad \textit{and}\\ \mathsf{E}[\sum_{t\in\mathcal{T}} m_t] &= \mathcal{O}(mr).\end{aligned}$$

Proof. Consider the first equation. We observe that the sum is the same as $\sum_{j=1}^{n}\sqrt{q_j}$, where q_j is the cardinality of Q_Δ for the tetrahedron Δ containing the point $p_j \in P$. Since the expectation is additive, and since by Jensen's inequality $\mathsf{E}[\sqrt{q_j}] \le \sqrt{\mathsf{E}[q_j]}$, we can concentrate on showing that $\mathsf{E}[q_j] = \mathcal{O}(\frac{m}{r})$.

Let $p \in P$ be fixed and let $q = |Q_{\Delta_0}|$ for the tetrahedron $\Delta_0 \in bv\text{-}Vor(U\cup R)$ with $p \in \Delta_0$. We want to show $\mathsf{E}[q] = \mathcal{O}(\frac{m}{r})$. Define $\mathcal{T}'_S = \{\Delta \in \mathcal{T}_S \mid p \in \Delta\}$, let $|\Delta| = |reg(\Delta)\cap Q|$, and let $\Pr_\Delta$ denote the probability that $\Delta = \Delta_0$ under the assumption that $p \in \Delta$. With these definitions, $\mathsf{E}[q] = \sum_{\Delta\in\mathcal{T}'_{U\cup Q}} |\Delta|\cdot\Pr_\Delta$.

By Lemma 6, we can bound $\Pr_\Delta$ as follows. Clearly, $\Delta = \Delta_0$ for $\Delta \in \mathcal{T}'_{U\cup Q}$ if and only if $\Delta \in \mathcal{T}'_{U\cup R}$ and $reg(\Delta)\cap R = \emptyset$. Now recall that there is a unique set $T_\Delta \subseteq U\cup Q$ with $|T_\Delta| \le 10$ such that $\Delta \in \mathcal{T}_{U\cup R}$ if and only if $T_\Delta \subseteq U\cup R$ (this is true because of our general position assumption).

To put Δ in $\mathcal{T}'_{U\cup R}$, we must choose these $|T_\Delta|$ points. To satisfy the second condition, the remaining $r - |T_\Delta|$ points must be chosen from $Q\setminus(reg(\Delta)\cup T_\Delta)$. We thus have

$$\Pr_\Delta = \binom{m-|\Delta|-|T_\Delta|}{r-|T_\Delta|} \Big/ \binom{m}{r}.$$

Since

$$\binom{m-|\Delta|-|T_\Delta|}{r-|T_\Delta|} = \Theta(\tfrac{m}{r})\binom{m-|\Delta|-|T_\Delta|}{r-|T_\Delta|-1},$$

we have

$$\mathsf{E}[q] = \mathcal{O}(\frac{m}{r})\sum_{\Delta\in\mathcal{T}'_{U\cup Q}} \frac{\binom{|\Delta|}{1}\binom{m-|\Delta|-|T_\Delta|}{r-|T_\Delta|-1}}{\binom{m}{r}}.$$

Observe that the summand, $\binom{|\Delta|}{1}\binom{m-|\Delta|-|T_\Delta|}{r-|T_\Delta|-1}/\binom{m}{r}$, is the probability that Δ is a tetrahedron in $\mathcal{T}'_{U\cup R}$ with $|reg(\Delta)\cap R| = 1$. The sum is therefore the expected number of such tetrahedra. By the last lemma, this expectation is at most 10.

Now consider the second equation. We use a similar argument as above, substituting the set $\mathcal{T}$ for $\mathcal{T}'$. We thus obtain

$$\mathsf{E}[\sum m_\Delta] = \mathcal{O}(\frac{m}{r})\sum_{\Delta\in\mathcal{T}_{U\cup Q}} \frac{\binom{|\Delta|}{1}\binom{m-|\Delta|-|T_\Delta|}{r-|T_\Delta|-1}}{\binom{m}{r}}.$$

Here, the sum on the right side is the expected number of tetrahedra Δ with $|reg(\Delta)\cap R| = 1$. Theorem 3.2 of [CS] can be used to show that this expectation is $\mathcal{O}(r^2)$. ⊡

Putting Theorem 5 and 9 together, we obtain

Theorem 12 *A Euclidean minimum spanning tree of a set of N points in $\mathbb{E}^3$ can be computed in expected time $O(N^{4/3}\log^{4/3} N)$ by a randomized algorithm.*

6 Conclusion

We have shown a close relationship between the Euclidean minimum spanning tree problem and the bichromatic closest pair problem in $\mathbb{E}^d$. As a result we get an improved algorithm for the three-dimensional EMST problem.

It should be noted that the techniques developed in this paper permit the computation of minimum spanning trees for metrics with polyhedral unit ball, such as the L_1- and the L_∞-metric. The idea is to construct a sufficiently narrow frame as a *refinement* of the frame induced by the edges of the unit ball. The distances in every cone of this frame are then determined by only one dimension. We can thus apply range trees and Yao's result to compute a EMST in time $O(N \log^d N)$. This is better than the algorithms of [GBT] for the L_1-metric and dimension greater than 5, while for the L_∞-metric and for the L_1-metric with $d \leq 5$ their algorithms are better by one or two log-factors.

We have no reason to believe that our algorithm is optimal, so it remains an open question whether the running time can be further improved. Also, it would be interesting if we could determinize our algorithm without increasing the running time.

References

[Cl] K. Clarkson, Fast expected-time and approximate algorithms for geometric minimum spanning tree, *Proceedings 16th Annual ACM Symposium on Theory of Computing*, 1984, pp. 342-348.

[CEGSW] K. Clarkson, H. Edelsbrunner, L. Guibas, M. Sharir and E. Welzl, Combinatorial complexity bounds for arrangements of curves and spheres, *Discrete and Computational Geometry* 5 (1990), 99-160.

[CS] K. Clarkson and P. Shor, Applications of random sampling in computational geometry II, *Discrete and Computational Geometry* 4 (1989), 387-422.

[Ed] H. Edelsbrunner, *Algorithms in Combinatorial Geometry*, Springer-Verlag, Heidelberg, 1987.

[EGS] H. Edelsbrunner, L.J. Guibas, and J. Stolfi, Optimal point location in a monotone subdivision, *SIAM J. Computing* 15 (1986), 317–340.

[ESh] H. Edelsbrunner and M. Sharir, A hyperplane incidence problem with applications to counting distances, manuscript, 1990.

[GBT] H. N. Gabow, J. L. Bentley and R. E. Tarjan, Scaling and related techniques for geometry problems, *Proceedings 16th Annual ACM Symposium on Theory of Computing*, 1984, pp. 135–143.

[PS] F. Preparata and M. Shamos, *Computational Geometry - an Introduction*, Springer-Verlag, New York, 1985.

[PT] F. Preparata and R. Tamassia, Efficient spatial point location, *Lecture Notes in Computer Science* 382 (1989), 3-11.

[ST] N. Sarnak and R. Tarjan, Planar point location using persistent search trees, *Communications of ACM* 29 (1986), 669-679.

[Se] R. Seidel, A convex hull algorithm optimal for point sets in even dimensions, Technical Report 81-14, Dept. Computer Science, Univ. British Columbia, Vancouver, 1981.

[Se2] R. Seidel, Linear programming and convex hulls made easy, *Proceedings 6th Annual Symposium on Computational Geometry*, 1990, to appear.

[SH] M. Shamos and D. Hoey, Closest-point problems, *Proceedings 16th Annual IEEE Symposium on Foundations of Computer Science*, 1975, pp. 151–162.

[Va] P. Vaidya, A fast approximate algorithm for minimum spanning tree in k-dimensional space, *Proceedings 25th Annual IEEE Symposium on Foundations of Computer Science*, 1984, pp. 403-407.

[Ya] A. Yao, On constructing minimum spanning trees in k-dimensional spaces and related problems, *SIAM J. Computing* 11 (1982), 721–736.