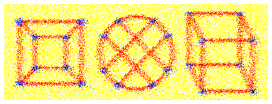
Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 8, no. 1, pp. 27–38 (2004)

Extreme Distances in Multicolored Point Sets

Adrian Dumitrescu

Computer Science
University of Wisconsin–Milwaukee
3200 N. Cramer Street
Milwaukee, WI 53211, USA
http://www.cs.uwm.edu/faculty/ad/
ad@cs.uwm.edu

Sumanta Guha

Computer Science & Information Management
Asian Institute of Technology
P.O. Box 4, Klong Luang
Pathumthani 12120, Thailand
guha@ait.ac.th

Abstract

Given a set of n colored points in some d-dimensional Euclidean space, a bichromatic closest (resp. farthest) pair is a closest (resp. farthest) pair of points of different colors. We present efficient algorithms to maintain both a bichromatic closest pair and a bichromatic farthest pair when the the points are fixed but they dynamically change color. We do this by solving the more general problem of maintaining a bichromatic edge of minimum (resp. maximum) weight in an undirected weighted graph with colored vertices, when vertices dynamically change color. We also give some combinatorial bounds on the maximum multiplicity of extreme bichromatic distances in the plane.

Article Type	Communicated by	Submitted	Revised
regular paper	M. T. Goodrich	January 2003	March 2004

A preliminary version of this paper appeared in the Proceedings of the Second International Conference on Computational Science (Workshop on Computational Geometry and Applications CGA '02), Amsterdam, April 2002. In Lecture Notes in Computer Science, Vol. 2331, 2002, 14–25.

1 Introduction

Our work is motivated by the problem of determining closest and farthest pairs from an input point set in $\mathbb{R}^d$. The classical (static, uncolored) version of this problem is to determine a closest or farthest pair from a set of n points in $\mathbb{R}^d$. Extensive work has been reported on this problem — see Eppstein [9] and Mitchell [13] for recent surveys, particularly of the closest pairs problem. Dynamizations of the uncolored closest/farthest pairs problem for points in $\mathbb{R}^d$, allowing for insertion and deletion of points, have been of considerable recent interest as well. For information about the dynamic problem the reader is referred to a recent paper by Bespamyatnikh [4] and the bibliography therein.

The colored version of the problem is, typically, given a set of n points in $\mathbb{R}^d$ each colored either red or blue, to find a closest pair of points of different colors (i.e., the *bichromatic closest pair* or BCP problem) or a farthest pair of different colors (i.e., the *bichromatic farthest pair* or BFP problem). There is extensive literature on the BCP and BFP problems; e.g., see [1, 5, 6, 12, 17, 19, 23].

Dynamic versions of the BCP and BFP problems have been studied too [7, 8], again, as in the uncolored version, from the point of view of inserting into and deleting from the point set. The best update times are polynomial in the size of the point set.

In this paper we consider the dynamic bichromatic closest and farthest pairs problem — in a *multicolor* setting — where the point set itself is fixed, but colors of points change. To our knowledge, ours is the first paper to consider this restricted dynamism and, not surprisingly, when points are from an Euclidean space, our update times are superior to and our algorithms less complicated than the best-known ones for the more general dynamic problems mentioned above, where points themselves may be inserted and deleted. In fact, we first consider the more general problem of maintaining extreme pairs in an undirected weighted graph where vertices dynamically change color, and provide efficient algorithms. This sort of dynamic graph algorithm is so far uncommon; it appears most closely related to work on dynamically switching vertices on and off in graphs [10].

In addition to its obvious relevance in the context of closest/farthest pairs problems, a scenario when the problem under consideration arises, is when a set of processes is competing for control of each of a fixed set of resources and, consequently, at any instant, each process controls a group of resources. Assuming a situation where control changes rapidly, it may be useful to quantify and maintain information about the relative disposition of the different groups of resources.

Summary of Our Results. In this paper, we obtain the following results on the theme of computing extreme distances in multicolored point sets, including:

(1) We show that the bichromatic closest (resp. farthest) pair of points in a multicolored point set in $\mathbb{R}^d$ can be maintained under dynamic color changes in sub-logarithmic time and linear space after suitable prepro-

cessing. We do this by solving the more general problem of maintaining in logarithmic time a bichromatic edge of minimum (resp. maximum) weight in an undirected weighted graph with colored vertices, when vertices dynamically change color.

(2) We present combinatorial bounds on the maximum number of extreme distances in multicolored planar point sets. Our bounds are tight up to multiplicative constant factors.

2 Dynamic Color Changes

2.1 Weighted graphs

Let $G = (V, E)$ be an undirected graph with colored vertices and weighted edges (i.e., a coloring $c : V \to \mathbb{N}$ and a weight function $w : E \to \mathbb{R}$). We further assume that G is connected, as the extension to the general case is straightforward. A *bichromatic edge* of G is one joining vertices of different colors. A *minimum* (resp. *maximum*) *bichromatic edge* is a bichromatic edge of least (resp. greatest) weight, if exists. Subsequent input is a sequence of color updates, where one update consists of a (vertex, color) pair — note that a color is any non-negative integer. We provide algorithms to maintain both minimum and maximum bichromatic edges after each update. When there is no bichromatic edge in G we report accordingly. After suitable preprocessing, our algorithms run in linear space and logarithmic time per update.

We begin with a simple observation, which was previously made for the two color case and a geometric setting [1, 5]:

Observation 1 *Let G be an undirected graph with colored vertices and weighted edges. Then a minimum spanning tree (MST) of G contains at least one minimum bichromatic edge, if exists. Similarly, a maximum spanning tree (XST) of G contains at least one maximum bichromatic edge, if exists.*

Proof: Assume that a minimum bichromatic edge pq of G has smaller weight than each bichromatic edge of an MST T of G. Consider the unique path in T between p and q. Since p and q are of different colors there exists a bichromatic edge rs on this path. Exchanging edge rs for pq would reduce the total weight of T, which is a contradiction.

The proof that an XST of G contains at least one maximum bichromatic edge is analogous. □

Let $T^{MST}(n, m)$ be the time complexity of a minimum spanning tree computation on a (connected) graph with n vertices and m edges [20]; note that this is the same as the time complexity of a maximum spanning tree computation on such a graph.

Theorem 1 *Let $G = (V, E)$ be an undirected connected graph with colored vertices and weighted edges, where $|V| = n$ and $|E| = m$. Then a minimum (resp.*

maximum) bichromatic edge can be maintained under dynamic color changes in $O(\log n)$ *update time, after* $O(T^{MST}(n,m))$ *time preprocessing, and using* $O(n)$ *space.*

Proof: We only describe the algorithm for minimum bichromatic edge maintenance, as the maximum bichromatic edge maintenance is analogous.

In the preprocessing step, compute T, a MST of G. View T as a rooted tree, such that for any non-root node v, $p(v)$ is its parent in T. Conceptually, we are identifying each edge $(v, p(v))$ of T with node v of T. The algorithm maintains the following data structures:

- For each node $v \in T$, a balanced binary search tree C_v, called the *color tree at* v, with node keys the set of colors of children of v in T. C_v is accessible by a pointer at v. For example if node v has 10 children colored by $3, 3, 3, 3, 5, 8, 8, 9, 9, 9$, C_v has 4 nodes with keys $3, 5, 8, 9$.
- For each node $v \in T$ and for each color class c of the children of v, a min-heap $H_{v,c}$ containing edges (keyed by weight) to those children of v colored c. In the above example, these heaps are $H_{v,3}, H_{v,5}, H_{v,8}, H_{v,9}$. $H_{v,c}$ is accessible via a pointer at node c in C_v.
- A min-heap H containing a subset of bichromatic edges of T (keyed by weight). Specifically, for each node v and for each color class c, *distinct* from that of v of the children of v, H contains one edge of minimum weight from v to children of color c. If edge $(v, p(v))$ is in H, there is a pointer to it from v.

The preprocessing step computes C_v and $H_{v,c}$, for each $v \in T$ and color c, as well as H, in $O(n \log n)$ total time. All pointers are initialized in this step as well. The preprocessing time-complexity, clearly dominated by the MST computation, is $O(T^{MST}(n,m))$.

Next we discuss how, after a color change at a vertex v, the data structures are updated in $O(\log n)$ time. Without loss of generality assume that v's color changes from 1 to 2. Let $u = p(v)$ and let j be the color of u. Assume first that v is not the root of T.

Step 1. Search for colors 1 and 2 in C_u and locate $H_{u,1}$ and $H_{u,2}$ (the latter may not exist prior to the update, in which case color 2 is inserted into C_u, together with a pointer to an initially empty $H_{u,2}$). Edge (u, v) is deleted from $H_{u,1}$ and inserted into $H_{u,2}$ (if this leaves $H_{u,1}$ empty then it is deleted, and so is color 1 from C_u). The minimum is recomputed in $H_{u,1}$, if exists, and $H_{u,2}$. If $j = 1$, the minimum weight edge in $H_{u,2}$ updates the corresponding item in H (i.e., the item in H that is currently the edge of minimum weight from u to children colored 2). If $j = 2$, the minimum weight edge in $H_{u,1}$ (if exists) updates the corresponding item in H. If $j > 2$, both minimum weight edges in $H_{u,1}$ (if exists) and $H_{u,2}$ update the corresponding items in H.

In all the preceding cases the corresponding pointers to edges in H are updated as required.

Step 2. Search for colors 1 and 2 in C_v and locate $H_{v,1}$ and $H_{v,2}$. The minimum weight edge of $H_{v,1}$ (if exists) is inserted into H, the minimum weight edge of $H_{v,2}$ (if exists) is deleted from H, and the corresponding pointers to edges in H are updated as required.

Step 3. If H is not empty, the minimum bichromatic edge is recomputed as the minimum item in H and returned. If H is empty, it is reported that there are no bichromatic edges.

If v is the root of T, Step 1 in the above update sequence is simply omitted. One can see that the number of tree search and heap operations per update is bounded by a constant, thus the update time is $U(n) = O(\log n)$. The total space used by the data structure is clearly $O(n)$. □

2.2 Euclidean spaces

We next specialize to the situation where the vertex set of G consists of a set S of n points in $\mathbb{R}^d$ and G is the complete graph with the Euclidean metric (i.e., the straight-line distance between pairs of points). In this case a minimum spanning tree of G is called an *Euclidean minimum spanning tree* (EMST) of S and a maximum spanning tree of G is called an *Euclidean maximum spanning tree* (EXST) of S. Let $T_d^{EMST}(n)$ denote the best-known worst-case time to compute an EMST of n points lying in $\mathbb{R}^d$, and $T_d^{EXST}(n)$ denote the analogous time complexity to compute an EXST (see [1, 2, 9, 13, 14, 19]).

By simply applying the algorithm of Theorem 1 we get corresponding results for the BCP and BFP problems.

Corollary 1 *Given a set S of n points in $\mathbb{R}^d$, a bichromatic closest (resp. farthest) pair can be maintained under dynamic color changes in $O(\log n)$ update time, after $O(T_d^{EMST}(n))$ (resp. $O(T_d^{EXST}(n))$) time preprocessing, and using $O(n)$ space.*

For the BCP problem we can take advantage of a geometric property of EMSTs to provide a more efficient algorithm. In the preprocessing step, compute T, an EMST of the point set S. Sort the edge lengths of T and keep integer priority queues of their indices ($\in \{1, \ldots, n-1\}$), instead of binary heaps on edge weights. The time taken by the basic priority queue operations would then be reduced to only $O(\log \log n)$ instead of $O(\log n)$ [21]. For $d = 2$, the maximum degree of a vertex in T is at most 6, whereas in d dimensions, it is bounded by $c_d = 3^d$, a constant depending exponentially on d [18]. Observe that the size of any color tree is at most c_d, therefore each binary search tree operation takes only $O(\log c_d) = O(d)$ time. The net effect is a time bound of $O(d + \log \log n)$ per color change. We thus have:

Corollary 2 *Given a set S of n points in $\mathbb{R}^d$, a bichromatic closest pair can be maintained under dynamic color changes in $O(d + \log \log n)$ update time, after $O(T_d^{EMST}(n))$ time preprocessing, and using $O(n)$ space.*

The integer-priority-queue idea appears less applicable for the general graph problem and for farthest pairs unless the number of colors is small.

Remarks. In the static case, the only attempt (that we know of) to extend to the multicolor version, algorithms for the bichromatic version, appears in [3]. The authors present algorithms based on Voronoi diagrams computation, for the bichromatic closest pair (BCP) problem in the plane — in the multicolor setting — that run in optimal $O(n \log n)$ time. In fact, within this time, they solve the more general *all bichromatic closest pairs problem* in the plane, where for each point, a closest point of different color is found. However the multicolor version of the BFP problem does not seem to have been investigated.

Let us first notice a different algorithm to solve the BCP problem within the same time bound, based on Observation 1. The algorithm first computes an EMST of the point set, and then performs a linear scan of its edges to extract a bichromatic closest pair. The same approach solves the BFP problem, and these algorithms generalize to higher dimensions. Their running times are dominated by EMST (resp. EXST) computations. (We refer the reader to [1, 2] for algorithms to compute EMST (resp. EXST) in sub-quadratic time.) This answers questions posed by Bhattacharya and Toussaint on solving BCP in the multicolor version (the "k-color distance problem, as they call it) [5], as well as on solving BCP and BFP in higher dimensions in sub-quadratic time [5, 6].

A natural related algorithmic question is the following. Given a multicolored set of n points in $\mathbb{R}^d$, a *bichromatic Euclidean spanning tree* is an Euclidean spanning tree where each edge joins points of different colors. Design an efficient algorithm to maintain a minimum bichromatic Euclidean spanning tree when colors change dynamically. Note that it may be the case that all its edges change after a small number of color flips.

3 Combinatorial Bounds in the Plane

In this section, we present some combinatorial bounds on the number of extreme distances in multicolored planar point sets. Let $f^{\min}(k, n)$ be the maximum multiplicity of the minimum distance between two points of different colors, taken over all sets of n points in $\mathbb{R}^2$ colored by exactly k colors. Similarly, let $f^{\max}(k, n)$ be the maximum multiplicity of the maximum distance between two points of different colors, taken over all sets of n points in $\mathbb{R}^2$ colored by exactly k colors. For simplicity, in the monochromatic case, the argument which specifies the number of colors will be omitted.

3.1 Minimum distance

It is well known that in the monochromatic case, $f^{\min}(n) = 3n(1 - o(1))$, and more precisely $f^{\min}(n) = \lfloor 3n - \sqrt{12n - 3} \rfloor$, cf. [11] (or see [16]). In the multicolored version, we have

Theorem 2 *The maximum multiplicity of a bichromatic minimum distance in multicolored point sets ($k \geq 2$) in the plane satisfies*

(i) $2n - O(\sqrt{n}) \leq f^{\min}(2, n) \leq 2n - 4$.

(ii) For $k \geq 3$, $3n - O(\sqrt{n}) \leq f^{\min}(k, n) \leq 3n - 6$.

Proof: We start with the upper bounds. Consider a set P of n points such that the minimum distance between two points of different colors is 1. Connect two points in P by a straight line segment, if they are of different colors and if their distance is exactly 1. We obtain a graph G embedded in the plane. It is easy to see that no pair of edges in G can cross: if there were such a crossing, the resulting convex quadrilateral would have a pair of bichromatic opposite sides with total length strictly smaller than of the two diagonals which create the crossing; one of these sides would then have length strictly smaller than 1, which is a contradiction. Therefore G is planar, which yields the upper bound in (ii). Since in (i), G is also bipartite, the upper bound in (i) follows.

To show the lower bound in (i), place about $n/2$ red points in a $\sqrt{n/2}$ by $\sqrt{n/2}$ square grid, and about $n/2$ blue points in the centers of the squares of the above red grid. The degree of all but $O(\sqrt{n})$ of the points is 4 as desired. To show the lower bound in (ii), it is enough to do so for $k = 3$ (for $k > 3$, recolor $k - 3$ of the points using a new color for each of them). Consider a hexagonal portion of the hexagonal grid, in which we color consecutive points in each row with red, blue and green, red, blue, green, etc., such that the (at most six) neighbors of each point are colored by different colors. The degree of all but $O(\sqrt{n})$ of the points is six, as desired. This is in fact a construction with the maximum multiplicity of the minimum distance — in the monochromatic case — modulo the coloring (see [16]). □

Remark. The previously mentioned tight bound on the maximum multiplicity of the minimum distance due to Harborth [11], namely $\lfloor 3n - \sqrt{12n - 3} \rfloor$, does not apply here directly, since the bichromatic closest pair may be not an uncolored closest pair. Is it possible to prove a $2n - \Omega(\sqrt{n})$ upper bound for $k = 2$, or a $3n - \Omega(\sqrt{n})$ upper bound for $k \geq 3$, or even exact bounds similar to Harborth's result?

3.2 Maximum distance

It is well known that in the monochromatic case, the maximum multiplicity of the diameter is $f^{\max}(n) = n$, (see [16]). In the multicolored version, we have

Theorem 3 *The maximum multiplicity of a bichromatic maximum distance in multicolored point sets ($k \geq 2$) in the plane satisfies*

(i) $f^{\max}(2, n) = n$, *for* $n \geq 4$.

(ii) For $k \geq 3$, $n - 1 \leq f^{\max}(k, n) < 2n$.

Proof: (i) We first prove the upper bound. Consider a set S of n points, colored red or blue, such that the maximum distance between a red and a blue point is 1. Let $b \geq 1$ and $r \geq 1$ stand for the number of blue and red points, respectively, and denote by m the number of red/blue pairs of points at unit distance. If either $b = 1$ or $r = 1$, clearly $m \leq n-1$. The red points all lie in the intersection P of the family F of b distinct unit disks centered at the blue points. It is easy to see that P is either a single point or a curvilinear polygon, whose sides are circular arcs, each of which is an arc of a distinct disk. In the first case, we are done by the previous remark, so consider the latter. Let $s \geq 2$ be the number of sides of P. It is clear that for each disk $D \in F$, either (1) D contains P in its interior, or (2) a side of P is an arc of the boundary of D, or (3) one vertex of P lies on the boundary of D, and the rest of P in the interior of D.

To show that $m \leq n$, we charge each maximum (bichromatic) pair to one of the points, such that no point gets charged more than once. A red point in the interior of P does not generate maximum pairs, since all blue points are at a distance less than 1 from it. A red point in the interior of a side of P generates exactly one maximum pair, with the blue point at the center of the corresponding arc. This unique pair is charged to the red point itself.

Assume that j vertices of P are red. Each such point generates at least two maximum pairs, that we consider first, with the blue points at the centers of the arcs intersecting at the vertex, for a total of $2j$ pairs for all such red points. Since $j \leq s$, so that $2j \leq j + s$, these maximum pairs can be charged to the j red points at the vertices of P, and to the blue points at the centers of intersecting arcs, so that each blue point is charged at most once: charge to the blue point for which when scanning the arc along the polygon in clockwise order, the red point is the first endpoint of the arc. The only maximum pairs that are unaccounted for, are those formed by red points at the vertices of P with (blue) centers of disks intersecting P at precisely those red points (item (3) above). Such a pair can be safely charged to the blue center point, since the rest of P (except the red vertex point) is at distance less than 1 from the blue point.

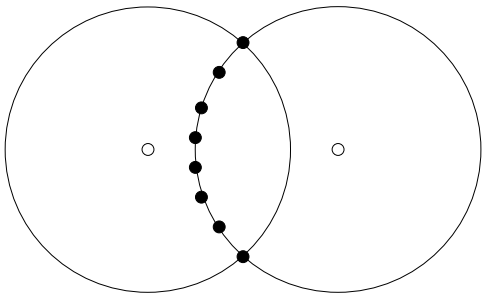

Figure 1: Maximum distance: a tight lower bound for two colors.

The point set in Figure 1, with $n - 2$ black points and two white points, shows that the bound is tight.

(*ii*) For the lower bound, place $n-1$ points at distance 1 from a point p in a small circular arc centered at p. Color p with color 1 and the rest of the points arbitrarily using up all the colors in $\{2,\ldots,k\}$. The maximum bichromatic distance occurs $n-1$ times in this configuration. (Better constructions exist, see the remark at the end of this section.)

We now prove the upper bound. It is based on an argument due to Pach and Tóth [15, 22]. Let E denote the set of maximum (bichromatic) pairs over a point set S. Suppose that $S = S_1 \cup S_2 \cup \ldots \cup S_k$ is the partition of S into monochromatic sets of colors $1, 2, \ldots, k$, and consider the complete graph K on vertex set $V = \{S_1, S_2, \ldots, S_k\}$. By the averaging principle, there exists a balanced bipartition of V, which contains a fraction of at least

$$\frac{\lfloor \frac{k}{2} \rfloor \lceil \frac{k}{2} \rceil}{\binom{k}{2}}$$

of the edges in E. Looking at this bipartition as a two-coloring of the point set, and using the upper bound for two colors in Theorem 3(i), one gets

$$\frac{\lfloor \frac{k}{2} \rfloor \lceil \frac{k}{2} \rceil}{\binom{k}{2}} |E| \leq n.$$

Depending on the parity of k, this implies the following upper bounds on $f^{\max}(k,n)$.
For even k,

$$f^{\max}(k,n) \leq \frac{2(k-1)}{k} n,$$

and for odd k,

$$f^{\max}(k,n) \leq \frac{2k}{k+1} n.$$

□

Remarks.

1. A *geometric graph* $G = (V, E)$ [16] is a graph drawn in the plane so that the vertex set V consists of points in the plane, no three of which are collinear, and the edge set E consists of straight line segments between points of V. Two edges of a geometric graph are said to be *parallel*, if they are opposite sides of a convex quadrilateral.

   Theorem 4 (Valtr [24]) *Let $l \geq 2$ be a fixed positive integer. Then any geometric graph on n vertices with no l pairwise parallel edges has at most $O(n)$ edges.*

Our original proof of the upper bound in Theorem 3 gives a weaker bound of $O(n)$, with a larger multiplicative constant, and under the assumption that no three points are collinear. Consider a set P of n points such that the maximum distance between two points of different colors is 1. Connect two points in P by a straight line segment, if they are of different colors and if their distance is exactly 1. We obtain a geometric graph $G = (V, E)$. We can show that G has no 4 pairwise parallel edges. The result then follows by Theorem 4 above.

2. Observe that $f^{\max}(n, n) = f^{\max}(1, n) = n$, for $n \geq 3$, since all points having different colors is equivalent in fact, to the well known monochromatic case. A lower bound of $\frac{3}{2}n - O(1)$ can be obtained for certain values of k. However, determining a tight bound for the entire range $2 \leq k \leq n$, remained elusive to us. It is interesting to note that as k gets close to n, the best lower bound we have is roughly n, while the upper bound is essentially $2n$.

Acknowledgment. We wish to thank the anonymous referee for a thorough reading and useful suggestions that have lead to improved results in Section 2.2.

References

[1] P. K. Agarwal, H. Edelsbrunner, O. Schwarzkopf and Emo Welzl, Euclidean minimum spanning trees and bichromatic closest pairs, *Discrete & Computational Geometry*, 6:407–422, 1991.

[2] P. K. Agarwal, J. Matoušek and S. Suri, Farthest neighbors, maximum spanning trees and related problems in higher dimensions, *Computational Geometry: Theory and Applications*, 1:189–201, 1992.

[3] A. Aggarwal, H. Edelsbrunner, P. Raghavan and P. Tiwari, Optimal time bounds for some proximity problems in the plane, *Information Processing Letters*, 42(1):55–60, 1992.

[4] S. N. Bespamyatnikh, An Optimal Algorithm for Closest-Pair Maintenance, *Discrete & Computational Geometry*, 19:175–195, 1998.

[5] B. K. Bhattacharya and G. T. Toussaint, Optimal algorithms for computing the minimum distance between two finite planar sets, *Pattern Recognition Letters*, 2:79–82, 1983.

[6] B. K. Bhattacharya and G. T. Toussaint, Efficient algorithms for computing the maximum distance between two finite planar sets, *Journal of Algorithms*, 4:121–136, 1983.

[7] D. Dobkin and S. Suri, Maintenance of geometric extrema, *Journal of the ACM*, 38:275–298, 1991.

[8] D. Eppstein, Dynamic Euclidean minimum spanning trees and extrema of binary functions, *Discrete & Computational Geometry*, 13:111–122, 1995.

[9] D. Eppstein, Spanning trees and spanners, in J.-R. Sack and J. Urrutia (Editors), *Handbook of Computational Geometry*, pages 425–461, Elsevier, North-Holland, 2000.

[10] D. Frigioni and G. F. Italiano, Dynamically switching vertices in planar graphs, *Algorithmica*, 28(1):76–103, 2000.

[11] H. Harboth, Solution to problem 664A, *Elemente der Mathematik*, 29:14–15, 1974.

[12] D. Krznaric, C. Levcopoulos, Minimum spanning trees in d dimensions, *Proceedings of the 5th European Symposium on Algorithms*, LNCS vol. 1248, pages 341–349, Springer Verlag, 1997.

[13] J. S. B. Mitchell, Geometric shortest paths and geometric optimization, in J.-R. Sack and J. Urrutia (Editors), *Handbook of Computational Geometry*, pages 633–701, Elsevier, North-Holland, 2000.

[14] C. Monma, M. Paterson, S. Suri and F. Yao, Computing Euclidean maximum spanning trees, *Algorithmica*, 5:407–419, 1990.

[15] J. Pach, personal communication with the first author, June 2001.

[16] J. Pach and P.K. Agarwal, *Combinatorial Geometry*, John Wiley, New York, 1995.

[17] J. M. Robert, Maximum distance between two sets of points in $\mathbb{R}^d$, *Pattern Recognition Letters*, 14:733–735, 1993.

[18] G. Robins and J. S. Salowe, On the Maximum Degree of Minimum Spanning Trees, *Proceedings of the 10-th Annual ACM Symposium on Computational Geometry*, pages 250–258, 1994.

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

[20] R. E. Tarjan, *Data Structures and Network Algorithms*. Society for Industrial Mathematics, 1983.

[21] M. Thorup, On RAM priority queues, *SIAM Journal on Computing*, 30(1):86–109, 2000.

[22] G. Tóth, personal communication with the first author, June 2001.

[23] G. T. Toussaint and M. A. McAlear, A simple $O(n \log n)$ algorithm for finding the maximum distance between two finite planar sets, *Pattern Recognition Letters*, 1:21–24, 1982.

[24] P. Valtr, On geometric graphs with no k pairwise parallel edges, *Discrete & Computational Geometry*, 19(3):461–469, 1998.