Computing Euclidean Bottleneck Matchings in Higher Dimensions

Alon Efrat* Matthew J. Katz†

October 6, 1998

Abstract

We extend the planar results of Chang et al. [5] to higher dimensions, and show that given a set A of $2n$ points in d-space it is possible to compute a Euclidean bottleneck matching of A in roughly $O(n^{1.5})$ time, for $d \leq 6$, and in subquadratic time, for any constant $d > 6$. If the underlying norm is L_∞, then it is possible to compute a bottleneck matching of A in $O(n^{1.5} \log^{0.5} n)$ time, for any constant $d \geq 2$.

1 Introduction

Let $G = (V, E)$ be a graph. A *matching* M in G is a subset of E such that each $v \in V$ is adjacent to at most one edge of M. M is *maximum* if $|M| \geq |M'|$, for any other matching M' in G. M is *perfect* if each $v \in V$ belongs to an edge of M. (Obviously, if a perfect matching exists then $|V|$ is even.) The problem of computing a perfect matching in G has been studied extensively. The best known solution is due to Micali and Vazirani [13]; it computes a perfect matching in G in time $O(|E|\sqrt{|V|})$. (See also [9].)

When weights are associated with the edges of G, it is often desirable to compute a perfect matching that is optimal with respect to some criterion. A *minimum weight matching* minimizes the sum of the weights of the edges of the matching, a *bottleneck matching* minimizes the maximum weight of an edge of the matching, a *most uniform matching* minimizes the difference

*School of Mathematical Sciences, Tel Aviv University, Tel-Aviv 69978, Israel. *alone@cs.tau.ac.il*

†Department of Mathematics and Computer Science, Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel. *matya@cs.bgu.ac.il*

between the maximum weight and the minimum weight, and a *minimum deviation matching* minimizes the difference between the average weight and the minimum weight (alternatively, minimizes the difference between the maximum weight and the average weight). Much work has been done on the problems of finding efficient algorithms for computing these matchings; see e.g. [7, 8, 10, 11, 12].

The Euclidean versions of these problems have also been studied. They are special important cases of these problems that admit more efficient solutions that exploit Geometry. In the Euclidean versions, the set of vertices V is a set of points in $\mathbb{R}^d$, and G is the complete graph over V. The weight associated with an edge (a, b) is the Euclidean distance between a and b. See [2, 5, 6, 14, 16] for a sample of results concerning Euclidean matching.

In this paper we consider the Euclidean version of the bottleneck matching problem. Let A be a set of $2n$ points in $\mathbb{R}^d$, and let G be the complete graph over A. The weight of an edge (a, b) is simply the Euclidean distance between a and b, and we assume that all weights are distinct. We wish to compute a perfect matching M in G such that the weight of the 'heaviest' edge in M is minimal, that is, the maximum distance between a matched pair of points is minimal. The best known algorithm for computing a bottleneck matching in a general graph is due to Gabow and Tarjan [7]; it runs in time $O(m\sqrt{n \log n})$, where n is the number of vertices and m is the number of edges. Chang et al. [5] have shown for $d = 2$ that a subgraph of G of size $O(n)$, called the *17 relative neighborhood graph* of A and denoted $17\mathrm{RNG}(A)$ (see definition in Section 2), already contains a bottleneck matching of G. Thus, by applying the algorithm of Gabow and Tarjan to $17\mathrm{RNG}(A)$, a bottleneck matching in the plane can be computed in $O(n^{1.5} \log^{0.5} n)$ time, after computing $17RNG(A)$, which can be done within the same time bound. See also [14].

We extend the elegant planar results of Chang et al. [5] to higher dimensions, and show that for any fixed dimension d there exists a constant $k = k(d)$ such that $k\mathrm{RNG}(A)$ contains a bottleneck matching of A. Since the size of $k\mathrm{RNG}(A)$ is linear in n in any dimension, and since $k\mathrm{RNG}(A)$ (or related linear-size supergraphs of it) can be computed efficiently, we obtain subquadratic algorithms for computing a bottleneck matching in any fixed dimension. In particular, for $d \leq 6$ a bottleneck matching of A can be computed in roughly $O(n^{1.5})$ time. Moreover, if the underlying norm is L_∞, then for any fixed dimension a bottleneck matching of A can be computed in time $O(n^{1.5} \log^{0.5} n)$.

2 Bottleneck Matching in Higher Dimensions

Let A be a set of $2n$ points in $\mathbb{R}^d$, $d \geq 3$. We show how to compute a bottleneck matching M^* in the (complete) Euclidean graph over A. We assume that the $\binom{n}{2}$ distances between pairs of points in A are distinct. Let $B_r(p)$ denote the ball of radius r centered at point p. For two points $p, q \in \mathbb{R}^d$, let *lune*(p,q) denote the region $B_{|p-q|}(p) \cap B_{|p-q|}(q)$, where $|p-q|$ is the distance between p and q. The k *relative neighborhood graph* of A, denoted kRNG(A), is a graph $G(A,E)$, whose nodes are the points of A, and for $p, q \in A$, the edge (p,q) is in the set of edges E if and only if the number of points of A other than p and q that lie in *lune*(p,q) is less than k. In the plane, Chang et al. [5] have proven that 17RNG(A) contains a bottleneck matching. Thus, in order to compute M^* in this case, it is enough to consider 17RNG(A) whose size is only $O(n)$. We then apply to it the general $O(m\sqrt{n \log n})$-algorithm of Gabow and Tarjan [7]. Chang et al. compute the 17RNG in time $O(n^2)$, but it can be computed in time $O(n \text{ polylog } n)$. Su and Chang [14] describe how to construct in $O(n \log n)$ time another linear-size graph, called the 17-Gabriel graph, that contains the 17RNG. Thus in both cases M^* can be computed in total time $O(n^{1.5} \log^{0.5} n)$.

We show below that for any fixed dimension d, there exists a constant $k = k(d)$ such that kRNG(A) contains a bottleneck matching. Since the size of a kRNG (for constant k) remains linear also in higher dimensions (assuming the distances between pairs of points are distinct [1]), and since it can be computed in subquadratic time, M^* can also be computed in subquadratic time even in higher dimensions. We also show that if the underlying norm is L_∞, then M^* can be computed in time $O(n^{1.5} \log^{0.5} n)$ in any fixed dimension.

We will need the following definition. A perfect matching M_1 is lexicographically smaller than another perfect matching M_2, if v_{M_1} is lexicographically smaller that v_{M_2}, where v_M is the decreasing sequence consisting of the distances corresponding to the edges of the matching M.

In order to prove that 17RNG(A) contains an optimal matching, Chang et al. [5] show how to transform an arbitrary bottleneck matching M_1^* to a bottleneck matching M_2^* that is contained in 17RNG(A), by repeatedly applying one of four basic transformations to the current matching (initially M_1^*). An application of a *basic transformation* rematches the points adjacent to two or three edges of the current matching, so that the resulting matching

[1] This size might grow in degenerate situations; see [3].

is still optimal but is lexicographically smaller. After a finite sequence of basic transformations, a matching that is contained in 17RNG(A) is obtained. The proof of Chang et al. [5] holds also in higher dimensions (where 17 is replaced by an appropriate constant $k(d)$), provided that Lemma 2.1 below, which is proven in [5] for $d = 2$, remains correct for higher dimensions. Next we show that this is indeed the case.

Let M be a bottleneck matching. We may assume that none of the edges (u, v) of M is contained in the lune of another edge (p, q) of M. Otherwise, we could replace these two edges of M by the edges (p, u) and (q, v), and obtain a lexicographically smaller bottleneck matching, since $\max\{|p-q|, |u-v|\} > \max\{|p-u|, |q-v|\}$. Thus, after a finite number of such transformations we will end up with a bottleneck matching that satisfies this assumption. Let (p, q) be an edge of M, and let S be the subset of A consisting of the points that are matched with points lying in the interior of $lune(p, q)$. According to our assumption $S \cap lune(p, q) = \emptyset$. For a point $u \in S$, let n_u be its nearest point on the boundary of $lune(p, q)$ and put $r = |p - q|$.

Lemma 2.1 *If for every $u, v \in S$*

(i) $|u - p| > r$ *and* $|u - q| > r$,

(ii) $|u - v| > r$, *and*

(iii) $|u - v| > |u - n_u|$ *and* $|u - v| > |v - n_v|$,

then $|S|$ is a constant depending only on the dimension d. More precisely,

$$|S| \leq \lfloor \frac{11^d}{\omega_d} \rfloor + d2^d(\lceil \sqrt{d-1} \rceil)^{d-1} ,$$

where ω_d is the volume of a d-dimensional unit ball.

Proof: Assume p and q lie on the x_d-axis at heights $r/2$ and $-r/2$, respectively, and let O denote the origin. Let C be the (d-dimensional) axis-parallel cube whose center is O and whose edge length is $9r$. Using condition (ii) above, we can easily bound the number of points of S that lie inside C; it is less than $\lfloor \frac{(11r)^d}{\omega_d r^d} \rfloor = \lfloor \frac{11^d}{\omega_d} \rfloor$.

We now show that the number of points of S that lie in $\mathbb{R}^d - C$ is also bounded by some constant depending on d. Divide each facet f of ∂C into a disjoint collection of $(d-1)$-dimensional cubes with edge length at most

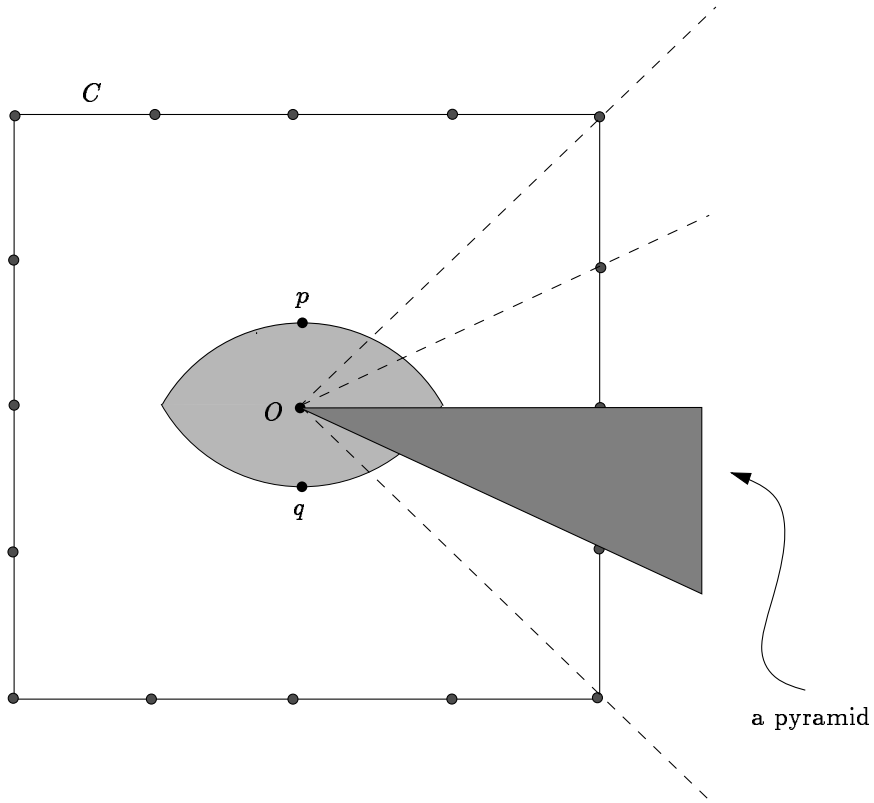

Figure 1: In the plane the 'pyramids' are wedges

γ, where $\gamma = \gamma(d)$ is such that for any two points $a, b \in f$ that lie in the same cube, the angle $\angle aOb \leq 45$. Let $\mathcal{C}$ denote the collection of cubes that is obtained from this decomposition of ∂C. It is easy to see that if we divide f so that its center point does not lie in the interior of a cube, then we may choose $\gamma = 4.5r/\sqrt{d-1}$ (since the shortest segment on f that is contained in a single 'quadrant' of f and rests on an angle of 45 degrees is of length $4.5r$), and then $|\mathcal{C}| \leq 2d(2\lceil\sqrt{d-1}\rceil)^{d-1} = d2^d(\lceil\sqrt{d-1}\rceil)^{d-1}$. We partition $\mathbb{R}^d$ into a collection of $|\mathcal{C}|$ '*pyramids*' by drawing all rays emanating from O and passing through a point on the boundary of a cube in $\mathcal{C}$ (see Figure 1).

Let P be such a pyramid. We now prove that the number of points of S in $P - C$ is at most one. Assume there are two points u, v of S in $P - C$, where $|v - O| \geq |u - O|$. Put $|u| = |u - O|$, $|v| = |v - O|$, and $\theta = \angle uOv$. Notice that $\theta \leq 45$, since both segments Ou and Ov intersect ∂C in the same cube of $\mathcal{C}$, i.e., in the cube defining P. Notice also that $|u|, |v| \geq 4.5r$, since u and v do not lie inside C. According to the cosine theorem

$$|u - v|^2 = |u|^2 + |v|^2 - 2|u||v|\cos\theta\,.$$

We show that the right side of the above equation is not greater than $|v - n_v|^2$, and therefore $|u - v| \leq |v - n_v|$ in contradiction with condition (iii) above. Indeed

$$\begin{aligned}
& |u|^2 + |v|^2 - 2|u||v|\cos\theta \\
\leq\ & |u|^2 + |v|^2 - \sqrt{2}|u||v| \\
\leq\ & |u||v| + |v|^2 - \sqrt{2}|u||v| = |v|^2 - (\sqrt{2}-1)|u||v| \\
\leq\ & |v|^2 - 4.5(\sqrt{2}-1)r|v| = |v|(|v| - 4.5(\sqrt{2}-1)r)\,.
\end{aligned}$$

Notice that $|v| \leq |v - n_v| + \frac{\sqrt{3}}{2}r$, since the distance between a point on the boundary of $lune(p, q)$ and the origin is at most $\frac{\sqrt{3}}{2}r$. Therefore we may continue the sequence of inequalities by replacing $|v|$ by $|v - n_v| + \frac{\sqrt{3}}{2}r$ to obtain, after some rearrangements, the following expression

$$\leq |v - n_v|^2 + (\sqrt{3} - 4.5(\sqrt{2}-1))r|v - n_v| + (3/4 - 4.5(\sqrt{2}-1)\sqrt{3}/2)r^2\,.$$

However, both the coefficient of $|v - n_v|$ and the free coefficient are negative, and thus the above expression is less than $|v - n_v|^2$. We conclude that the number of points of S that lie in $\mathbb{R}^d - C$ is at most $|\mathcal{C}|$.

Combining the two cases (inside C and in $\mathbb{R}^d - C$) we obtain

$$|S| \leq \lfloor \frac{11^d}{\omega_d} \rfloor + d2^d(\lceil \sqrt{d-1} \rceil)^{d-1} .$$

□

Using similar ideas it is easy to show that Lemma 2.1 is also true (with another constant depending on d) when the underlying norm is L_∞.

Lemma 2.1 together with the remarks preceding it lead to the conclusion that there exists a constant $k = k(d)$ such that kRNG(A) contains a bottleneck matching, and this conclusion is also true when the underlying norm is L_∞. Moreover, k is equal to the largest possible value for the expression $|S| + 1$ ([5]). However, if k' is a bound for $|S| + 1$, then clearly k'RNG(A) contains a bottleneck matching, since kRNG(A) $\subseteq k'$RNG(A). Our proof of Lemma 2.1 implies, for example, that in $\mathbb{R}^3$ a bottleneck matching is contained in 414RNG(A). (A slightly more careful calculation gives 358 instead of 414, which is probably still far from the optimum.)

Computing a bottleneck matching. Instead of computing kRNG(A), we prefer to compute a graph $G = (A, E)$, such that $|E| = O(n)$, and kRNG$(A) \subseteq G$. We first deal with the L_∞ case. Let $u \in A$, and let R^+ denote the region of $\mathbb{R}^d$ consisting of all points x such that all the components of the vector $x - u$ are non-negative (e.g., in the plane R^+ is the north-eastern quadrant of u). The idea is to find the k nearest neighbors of u in $A \cap R^+$ (under the L_∞ norm) and to connect each of them by an edge to u. This process is repeated for all $u \in A$, and for all the 2^d regions of $\mathbb{R}^d$ symmetric to R^+. In this way a graph G of size $O(kn)$ is obtained, and it is easy to verify that G contains kRNG(A). Indeed, if (u, v) is an edge of kRNG(A), then the number of points in *lune*(u,v) (see Figure 2) is less than k, and therefore v is among the k nearest neighbors of u in the region of u that contains v (the north-eastern quadrant of u in Figure 2), which implies that (u, v) is an edge of G.

We compute G as follows. Let B be the smallest axis-parallel cube such that (i) its most-negative corner coincides with u, and (ii) the number of points of A (other than u) lying in it is exactly k. In order to find the k nearest neighbors of u in $A \cap R^+$, we determine the edge length of B and then perform a reporting query with B. The edge length b of B is equal to the difference between the j'th component of some point x_u in $A \cap R^+$ and the j'th component of u, where $1 \leq j \leq d$. Thus, for each dimension j, we perform a binary search for b among the differences of the form $x.j - u.j$, where x is a point in $A \cap R^+$. For this, we use a standard d-dimensional

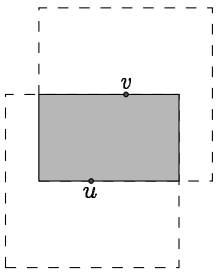

Figure 2: *lune*(u,v) under the L_∞ norm

orthogonal range tree that allows us to determine the number of points lying in a query range (i.e., an axis-parallel box) in time $O(\log^d n)$ and to report the k points lying in B in time $O(\log^d n + k)$. The number of counting queries that we perform in order to determine b is $O(\log n)$, and hence it is possible to compute G in time $O(n \log^{d+1} n)$. We now apply the algorithm of Gabow and Tarjan to the graph G (that contains kRNG(A)) to obtain a bottleneck matching of A in time $O(n^{1.5} \log^{0.5} n)$.

As to the L_2 case, Agarwal and Matoušek [3] show how to compute a kRNG, where k is some constant, in $O(n^{2(1-1/(d+1))+\varepsilon})$ time (assuming general position); see also [15].[2] They first construct a supergraph of the kRNG which is also of linear size, so we may take this supergraph as input to the algorithm of Gabow and Tarjan. The construction time of this supergraph is only $O(n^{4/3} \log^2 n)$ for $d = 3$, and $O(n^{2-2/(\lceil d/2 \rceil + 1)+\varepsilon})$ for $d \geq 4$; [1, 3, 4]. Thus we can compute M^* in $O(n^{1.5} \log^{0.5} n)$ time for $2 \leq d \leq 4$, and $O(n^{2-2/(\lceil d/2 \rceil + 1)+\varepsilon})$ time for $d \geq 5$. Hence we have:

Theorem 2.2 *Let A be a set of $2n$ points in d-space. A bottleneck matching of A can be computed in $O(n^{1.5} \log^{0.5} n)$ time for $2 \leq d \leq 4$, in $O(n^{1.5+\varepsilon})$ time for $5 \leq d \leq 6$, and in $O(n^{2-2/(\lceil d/2 \rceil + 1)+\varepsilon})$ time for any fixed $d > 6$. Under the L_∞ norm, a bottleneck matching of A can be computed in $O(n^{1.5} \log^{0.5} n)$ time for any fixed $d \geq 2$.*

Remark 2.3: The following observation is due to Pankaj Agarwal, it implies that finding a faster algorithm for computing a bottleneck matching

[2] In what follows, ε stands for a positive constant which can be chosen arbitrarily small with an appropriate choice of other constants of the algorithms.

in d-space, for $d > 6$, is probably a non-trivial problem. Assume that A consists of the two subsets A_1 and A_2, where $|A_1| = |A_2| = n/2$, and that $n/2$ is odd. Assume furthermore that the diameter of A_1 and the diameter of A_2 are very small compared to the diameter of A. In this case, a bottleneck matching of A must match the closest mixed pair a_1, a_2 where $a_1 \in A_1$ and $a_2 \in A_2$. But this is actually the famous *bichromatic closest pair problem in d-space* (see [4]), whose best known solution has the above bound.

Acknowledgment

We would like to thank Pankaj Agarwal and Micha Sharir for helpful discussions on the contents of this paper.

References

[1] P.K. Agarwal, personal communication.

[2] P.K. Agarwal, A. Efrat and M. Sharir, Vertical decomposition of shallow levels in 3-dimensional arrangements and its applications, *Proc. 11th ACM Symp. Comput. Geom.*, 1995, 39–50.

[3] P.K. Agarwal and J. Matoušek, Relative neighborhood graphs in three dimensions, *Computational Geometry: Theory and Applications* 2 (1992), 1–14.

[4] 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 (1992), 189–201.

[5] M.S. Chang, C.Y. Tang, and R.C.T. Lee, Solving the Euclidean bottleneck matching problem by k-relative neighborhood graphs, *Algorithmica* 8 (1992), 177–194.

[6] A. Efrat and A. Itai, Improvements on bottleneck matching and related problems using geometry, *Proc. 12th ACM Symp. Comput. Geom.*, 1996, 301–310. See also: A. Efrat, M.J. Katz and A. Itai, Improvements on bottleneck matching and related problems, using geometry, manuscript.

[7] H.N. Gabow and R.E. Tarjan, Algorithms for two bottleneck optimization problems, *J. Algorithms* 9 (1988), 411–417.

[8] S.K. Gupta and A.P. Punnen, Minimum deviation problems, *Oper. Res. Lett.* 7 (1988), 201–204.

[9] J. Hopcroft and R.M. Karp, An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs, *SIAM J. Computing* 2 (1973), 225–231.

[10] H. Kuhn, The Hungarian method for the assignment problem, *Naval Research Logistics Quarterly* 2 (1955), 83–97.

[11] E. Lawler, Combinatorial Optimization: Networks and Matroids, *Holt, Rinehart and Winston*, New-York, 1976.

[12] S. Martello, W.R. Pulleyblank, P. Toth and D. de Werra, Balanced optimization problems, *Operations Research Letters* 3 (1984), 275–278.

[13] S. Micali and V.V. Vazirani, An $O(\sqrt{|V|} \cdot |E|)$ algorithm for finding maximum matching in general graphs, *Proc. 21st IEEE Symp. on Foundations of Computer Science*, 1980, 17–27.

[14] T. Su and R. Chang, The k-Gabriel graphs and their applications, *1st Annual SIGAL International Symp. Algorithms*, LNCS 450, 1990, 66–75.

[15] T. Su and R. Chang, On constructing the relative neighborhood graphs in Euclidean k-dimensional spaces, *Computing* 46 (1991), 121–130.

[16] P.M. Vaidya, Geometry helps in matching, *SIAM J. Computing* 18 (1989), 1201–1225.