SERIE B — INFORMATIK

Weighted Closest Pairs *

Michael Formann◇

B 92-04
February 1992

Abstract

In this paper we study the following *weighted closest pair problem*: Given a set of planar objects with centerpoints, determine the *maximal scaling factor* δ_{max}, such that the objects scaled by δ_{max} are pairwise disjoint.
We describe a method to compute the maximal scaling factor in optimal $\mathcal{O}(n \log n)$ time for a wide class of objects, including disks generated by L_p-norms ($1 \leq p \leq \infty$).

*This work was partially supported by the ESPRIT II Basic Research Action of the European Community under contract No. 3075 (project ALCOM I).
◇Institut für Informatik, Fachbereich Mathematik, Freie Universität Berlin, Arnimallee 2–6, W1000 BERLIN 33, Germany; e-mail: formann@tcs.fu-berlin.de

1 Introduction

In this paper we study the following *weighted closest pair problem*: Given a set of planar objects with centerpoints, determine the *maximal scaling factor* δ_{max}, such that the objects scaled by δ_{max} are pairwise disjoint. Clearly δ_{max} can be computed in $\mathcal{O}(n^2)$ time, by taking the minimum of all $\binom{n}{2}$ pairwise maximal scaling factors. (It is assumed, that the computation of the maximal scaling factor of two objects is a primitive operation that can be done in $\mathcal{O}(1)$ time.) The goal of this article is to beat the $\mathcal{O}(n^2)$ time bound.

If all the objects considered are unit-disks then we are faced with the ordinary *closest pair problem*, that is to determine the closest distance between any pair of points. This is a well-studied fundamental problem of computational geometry (cf. [HNS90], [HS75], [BS76]). Another related problem is finding the closest pair among a set of objects (cf. [BH91], [For87], [Sha85], [Yap87]).

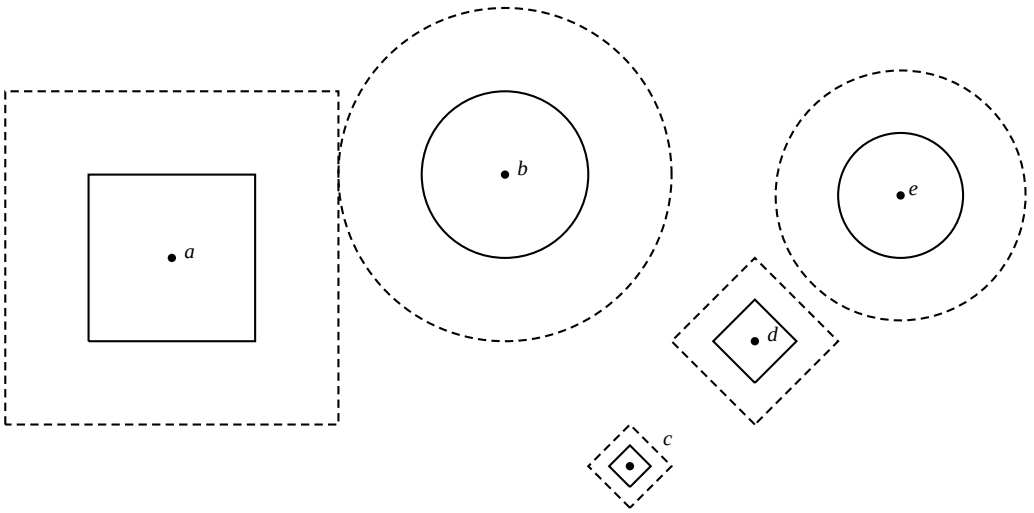

Figure 1: five objects (solid) scaled by δ_{max} (dashed)

Figure 1 illustrates the aforementioned closeness concepts. Five objects are drawn in solid lines in their original size. The δ_{max}-scaled objects are shown in dashed lines ($\delta_{max} = 2$). The square around a and the disk around b are the closest weighted pair, points c and d are the closest pair (in the Euclidean metric) and the shortest distance between any pair of objects (the Hausdorff-distance) occurs between the square around d and the circle around e.

In principle, we could determine the maximal scaling factor via a Voronoi diagram approach. For example we are given a set of disks with different radii. From

the center p of a disk we measure the weighted distance from p as

$$d_p(x) := \frac{\| p - x \|}{r_p},$$

where r_p denotes the radius of the disk around p. The bisector of two disks with centers p and q will be the set of points in the plane that have equal weighted distance to p and to q. Then we could define a pair p, q as *closest pair* if it minimizes the shortest weighted distance to its bisector among all pairs. Exactly the closest pairs of δ_{max}-scaled disks will touch. This approach doesn't lead to an efficient solution, because the Voronoi diagram might have quadratic complexity (cf. [AE84]).

We describe a method to compute the maximal scaling factor in optimal $\mathcal{O}(n \log n)$ time for a wide class of objects. The basic idea is as follows: In a preprocessing step we compute an over-estimate $\delta \geq \delta_{max}$ for δ_{max}. Clearly, if $\delta > \delta_{max}$ then there are intersections in the set of δ-scaled objects. We ensure, that after the preprocessing step only a linear number of pairs of the δ-scaled objects will intersect. We detect these intersections in $\mathcal{O}(n \log n)$ time by a standard sweep-technique and compute the related pairwise maximal scaling factors. The minimum of δ and these $\mathcal{O}(n)$ numbers will determine δ_{max}.

Potential applications arise in computer graphics (simultaneously resizing windows on a screen such that there is no overlap), robotics (maximizing the minimal workspace of stationary rotating robots), computational cartography (e.g. maximizing rectangular labels attached to certain geographical sites) etc.

The paper proceeds as follows. In Section 2 we show how to compute maximal scaling factors for a set of disks with possibly different weights. Section 3 discusses generalizations of this result for disks generated by different L_p-norms with different weights. Section 4 addresses further generalizations.

2 Weighted Closest Pairs for Disks

In this section we will describe how to compute the maximal scaling factor for a set of (topologically open) disks with (possibly) different radii. Firstly we define a certain property, the *halfmoon property* for disks. Then we show that if that property holds for a set of disks, then the number of intersecting pairs of disks is only linear. We will then present an algorithm to preprocess a set of disks — we scale them — such that the halfmoon property is fulfilled. On the way we describe how we glue everything together to an algorithm for computing the maximal scaling factor.

Let us now start with a definition.

Definition 1 *Let $\mathcal{D} = \{D_1, D_2, \ldots, D_n\}$ be a set of disks in $\mathbb{R}^2$. We say, that $\mathcal{D}$ fulfills the* halfmoon-property *if no disk in $\mathcal{D}$ cuts the vertical diameter of another disk of $\mathcal{D}$. For a disk $D_i \in \mathcal{D}$ we will call the two parts of D_i to the left and to the right of the vertical diameter* left *resp.* right halfmoon *of D_i and denote them by D_i^- resp. D_i^+.*

We sum up some simple facts about a set of disks that fulfills the halfmoon-property.

Observation 1 *Let $\mathcal{D} = \{D_1, D_2, \ldots, D_n\}$ be a set of disks with the halfmoon property.*

(1) No two left and no two right halfmoons intersect.

(2) No disk is completely contained in another.

(3) No point in the plane is covered by more than 2 disks of $\mathcal{D}$.

Proof: (1) and (2) are immediately clear. Note, that Fact (1) is equivalent to the halfmoon property. If (3) is violated, then we have a point in two left halfmoons or in two right halfmoons — a contradiction to (1). □

We are now ready to present our main lemma:

Lemma 1 *The number of intersecting pairs in set of disks that fulfills the halfmoon property is at most $3n - 6$.*

Proof: Let $\mathcal{D} = \{D_1, D_2, \ldots, D_n\}$ be a set of disks with the halfmoon property. We will show, that the graph G formed by putting vertices at the centers of the disks and drawing edges by straight line segments between centers of intersecting disks is plane. Then the claim follows.

Look at two embedded edges $\overline{ab}$ and $\overline{rs}$ with disjoint endpoints. There must be a point c on $\overline{ab}$ that is contained in the disk around a and also in the disk around b by definition of the edges. Similarly, there is a point t on $\overline{rs}$ that is contained in the disks around r and s. The points c and t must be distinct, otherwise we have found a point $c = t$ that is contained in four disks, a contradiction to Observation 1(3). Now draw the perpendicular bisector g between c and t. s and c must lie on different sides of g, otherwise the disk around s contains c and therefore c is contained in three disks. Similarly r and c must lie on different sides of g. Therefore s, t and r lie on the same side of g. By mirrorsymmetric arguments a, b and c lie on the other side of g. Therefore the segments $\overline{ab}$ and $\overline{rs}$ do not cross. □

In the proof of the Observation above, we have shown that the "intersection graph" of a set of disks with the halfmoon property is planar. The converse is also true, Koebe [Koe36] showed that any planar graph can be realized as the "contact graph" of a set of nonoverlapping disks in the plane. An elementary proof of this result, as well as two generalizations are given in [PR]. An alternative proof of Lemma 1 may be obtained via power diagrams (see [Aur87] for the apparatus of power diagrams). Note, that the regions of two intersecting disks are neighbours in the power diagram because of Observation 1(3). Therefore the edges of G are a subgraph of the dual graph of the power diagram, which is planar.

Let us now turn our attention to the description of our main algorithm. Similarly to the halfmoon property we could define two properties as follows.

Definition 2 *Let $\mathcal{D} = \{D_1, D_2, \ldots, D_n\}$ be a set of disks in $\mathbb{R}^2$. We say, that $\mathcal{D}$ fulfills the* left *(resp.* right*)* halfmoon-property *if no two left (resp. right) halfmoons in the set $D_1^-, D_2^-, \ldots, D_n^-$ (resp. $D_1^+, D_2^+, \ldots, D_n^+$) intersect.*

Similarly to the definition of the maximal scaling factor δ_{max} we define the *maximal left scaling factor* δ_{max}^- of a set of disks as the maximal number such that in the set of δ_{max}^--scaled disks no two left halfmoons intersect, δ_{max}^+ is analogously defined as the *maximal right scaling factor*. Clearly $\delta_{max} \leq \min\{\delta_{max}^-, \delta_{max}^+\}$ and if we scale the disks by $\min\{\delta_{max}^-, \delta_{max}^+\}$ then the left and the right halfmoon properties and therefore also the halfmoon property itself holds for the scaled disks. After that scaling process we can run a standard intersection detecting algorithm (cf. [BO79]) that will output at most $3n-6$ intersecting pairs of disks. Then we take the minimum of those pairwise scaling factors and $\min\{\delta_{max}^-, \delta_{max}^+\}$. This number clearly is δ_{max} the output of our algorithm.

So far we have reduced our original problem to that of computing the maximal left and right scaling factors. Without loss of generality we describe only the algorithm for computing the maximal right scaling factor δ_{max}^+. The basic idea of that algorithm is as follows: We start with an initial assumption about δ_{max}^+, say we set $\delta := +\infty$. Then we sweep in the plane with a vertical line from left to right. Whenever we meet a centerpoint of a disk D_i, we insert that disk into the vertical structure. If the right halfmoon of the newly inserted disk has intersections with right halfmoons of disks inserted before, we decrease δ so that the intersections disappear. When all disks have been processed the algorithm stops and outputs $\delta_{max}^+ = \delta$.

In order to work correctly our algorithm will maintain the following two invariants:

Invariant 1: All δ-scaled disks whose right halfmoon cuts the sweepline are stored in the vertical structure (sweepline-status) ordered by the y-coordinates of their centerpoints. All δ-scaled disks whose right halfmoon is to the right of the sweepline are not stored in the vertical structure.

Invariant 2: The right halfmoons of the δ-scaled disks already processed do not intersect.

It is clear that we have to insert a disk into the vertical structure exactly when the sweepline passes the centerpoint of that disk. Invariant 1 allows, that disks whose right halfmoon does no longer cut the sweepline, remain in the sweepline. Note that neither Invariant 1 nor Invariant 2 is destroyed if we decrease δ.

The only problem appears when a new disk D_i is inserted into the sweepline. Its right halfmoon may intersect other right halfmoons already processed and we have to decrease δ. We detect such situations and handle them as follows:
Look at the upper neighbour D_j of D_i in the vertical structure.

1. D_j lies completely behind the sweepline. Then it is clear that $D_j^+ \cap D_i^+ = \emptyset$ and we may delete D_j from the vertical structure.

2. $D_j^+ \cap D_i^+ = \emptyset$. Then it is clear, that D_i also does not cut any right halfmoon which is in the vertical structure and whose centerpoint is above the centerpoint of D_i.

3. $D_j^+ \cap D_i^+ \neq \emptyset$. Then we compute the pairwise maximal right scaling factor of D_j^+ and D_i^+. This number δ_{ij} must be lower than the current δ and we set $\delta := \delta_{ij}$. Two cases may appear:

   (a) D_j^+ now lies completely behind the sweepline. We proceed as in Case 1.

   (b) D_j^+ still cuts the sweepline. We proceed as in Case 2.

After that process D_i might have a new upper neighbour (in Cases 1 and 3a) and we have to repeat this process until either Case 2 occurs or D_i does no longer have an upper neighbour. A similar processing is done for the lower neighbour of D_i. Finally we will have again established Invariant 2.

As already said we start the sweeping process with $\delta := +\infty$. If the sweepline is to the left of the leftmost centerpoint the vertical structure is empty and we can scale the disks by $+\infty$ without violating Invariant 2. During the algorithm δ will only be decreased when required and only so much that still two right halfmoons will touch. Then it is clear that the final value of δ equals δ_{max}^+. Note, that for our purpose it is not necessary to compute the pairwise maximal *right* scaling factor in Case 3. In an actual implementation we could compute the pairwise maximal scaling factor instead, since the maximal scaling factor is really what we are interested in.

Lemma 2 *The maximal right scaling factor δ_{max}^+ of a set of n disks in the plane can be computed in $\mathcal{O}(n \log n)$ time.*

Proof: By the discussion above our algorithm computes δ_{max}^+. We only have to analyze the runtime. If we use some appropriate balanced tree scheme for the vertical structure then an insertion or deletion of a disk can be done in $\mathcal{O}(\log n)$ time. But when we process an inserted disk in order to maintain the invariants, we may have to look at many other disks.

The amount of work done for one of these disks is $\mathcal{O}(\log n)$ (neighbour-finding and computing the pairwise maximal right scaling factor). If the disk is deleted we charge that work to the deletion. At most two disks — the topmost and bottommost processed — are not deleted. That work is charged to the insertion.

Now insertions and deletions cost at most $\mathcal{O}(\log n)$ time. Note that every disk is inserted exactly once and deleted at most once. Therefore the total runtime of our algorithm is $\mathcal{O}(n \log n)$. □

We have now described all ingredients to solve the weighted closest pair problem and sum up in the following Theorem.

Theorem 1 *The maximal scaling factor of a set of disks can be computed in $\mathcal{O}(n \log n)$ time.*

Since our problem is a generalization of the closest pair problem, the $\Omega(n \log n)$ lower bound of this problem (see [PS85]) also applies in our more general setting. It is easy to see, that the space requirement of our algorithm is $\mathcal{O}(n)$.

3 Weighted Closest Pairs for Generalized Disks

In this section we will briefly indicate how the method developed for disks in Section 2 can be used for other objects shapes. Although the method works for more general shapes, we only demonstrate it for unit-disks generated by L_p-norms ($1 \leq p \leq \infty$) in order to keep the presentation simple.

So our setting is as follows:
Given a set of n convex planar figures (henceforth called *bodies*), each of them being a disk of some L_p-norm ($1 \leq p \leq \infty$) scaled by some positive number (weight), determine the *maximal scaling factor* δ_{max}, such that the bodies scaled by δ_{max} are pairwise disjoint.

Note that different bodies are allowed to be disks from different L_p-norms and with different weights. In a typical robot environment for example, some robots could be abstracted as ordinary disks (L_2), others could be squares (L_1, L_∞) (see Figure 1).

In the sequel we again define a *halfmoon property* for bodies as we did it for disks in Section 2. Unfortunately it is no longer true, that the straight line embedding of the "intersection graph" for a set of bodies with the halfmoon property is plane, but it is still planar, i.e. there is some other embedding of the graph which is crossing-free, but the straight line embedding has crossings. So again, the number of intersecting pairs of bodies is only linear. These intersections can be determined by a plane-sweep algorithm, the minimum of only linearly many pairwise scaling factors could be computed etc. similarly as demonstrated for disks in Section 2. This section shows only the planarity result. The actual computation of δ_{max} is done similar as for ordinary disks and therefore not described.

By the *vertical diameter* of a body we understand the intersection of the vertical line through the centerpoint of the body with the body. Analogously as done for disks in Section 2 we define the halfmoon-property for bodies.

Definition 3 *Let $\mathcal{B} = \{B_1, B_2, \ldots, B_n\}$ be a set of bodies in $\mathbb{R}^2$. We say, that $\mathcal{B}$ fulfills the* halfmoon-property *if no body in $\mathcal{B}$ cuts the vertical diameter of another body of $\mathcal{B}$. For a body $B_i \in \mathcal{B}$ we will call the two parts of B_i to the left and to the right of the vertical diameter* left *resp.* right halfmoon *of B_i and denote them by B_i^- resp. B_i^+.*

The main common feature of bodies can be stated as follows:

Observation 2 *Any body has horizontal tangents in the endpoints of its vertical diameter.*

This simple observation enables us to prove the following facts:

Observation 3 *Let $\mathcal{B} = \{B_1, B_2, \ldots, B_n\}$ be a set of bodies with the halfmoon property.*

(1) No two left and no two right halfmoons intersect.

(2) No disk is completely contained in another.

(3) No point in the plane is covered by more than 2 disks of $\mathcal{B}$.

Proof: By Observation 2 it is clear that Facts (1) and (2) hold. By Fact (1) a point must lie in different halfmoons of different bodies and therefore Fact (3) holds. □

Let us now state the main lemma of this section.

Lemma 3 *Let $\mathcal{B} = \{B_1, B_2, \ldots, B_n\}$ be a set of bodies with the halfmoon property. The graph G with the vertex set $\mathcal{B}$ and edges between any pair of intersecting bodies is planar.*

Proof: We place the vertices at the centerpoints of the bodies. Let us now take a look at some right (w.l.o.g.) halfmoon B_i^+ of some body B_i. By Observation 2 and the fact that all bodies are convex it follows that the intersections of B_i^+ with left halfmoons of other bodies all lie in disjoint open horizontal stripes and to the right of the vertical diameter. A similar argument holds for the left halfmoon B_i^-. If we now look at the relative complement of $\{B_j \,|\, 1 \leq j \leq n, i \neq j\}$ with respect to B_i

$$\text{residue}(B_i) := B_i \setminus \{B_j \,|\, 1 \leq j \leq n, i \neq j\}$$

we observe that this is a simply connected area containing the vertical diameter.

As already stated, the straight-line embedding of the edges of G is not plane, i.e there may be intersections. So we will give another layout for the edges. Any edge will be laid out completely in the two bodies, whose intersection defines it. So we split an edge going from B_i to B_j into three parts, the part which will be laid out in residue(B_i), the part in $B_i \cap B_j$ and the part in residue(B_j). The first parts of all edges with one endpoint in B_i can now be easily laid out star-like in residue(B_i) since this is a simply connected area and no intersections with other edges will appear. By symmetry, the third part of an edge can be laid out like its first part. The second part of an edge going from B_i to B_j is the only edge that must be laid out in $B_i \cap B_j$ and is therefore crossing-free. It can easily be laid out to link the two other parts. □

4 Further Extensions and Discussion

In the previous section we have extended our original ideas for circular disks to more general disks, i.e. for unit disks generated by L_p-norms. There the most important

feature was the existence of horizontal tangents in the endpoints of the vertical diameter of bodies (Observation 2). Of course we could also use our ideas for other convex shapes with the property cited above, for example convex polygons with that property. Also the choice of the vertical diameter with horizontal tangents is in some sense arbitrary. For other shapes other directions can, or have to be chosen. Furthermore also the convexity assumption can be relaxed.

Recall that it is assumed, that the computation of the maximal scaling factor of two objects is a $\mathcal{O}(1)$ time operation, therefore we have to restrict ourself to simple classes of objects to attain the promised runtime of $\mathcal{O}(n \log n)$. So for example in Section 3 we have to restrict ourself to some finite class of norms.

Furthermore, we have used multiplicative weights. Our methods also work for additive weights, but here we could also compute the Voronoi diagram (cf. [For87]) of the boundary of the figures and extract the closest pair from the set of neighbouring regions. Our approach may be more practicable.

References

[AE84] Franz Aurenhammer and Herbert Edelsbrunner. An optimal algorithm for constructing the weighted Voronoi diagram in the plane. *Pattern Recognition*, 17:251-257, 1984.

[Aur87] Franz Aurenhammer. Power diagrams: properties, algorithms, and applications. *SIAM J. Comput.*, 16:78-96, 1987.

[BH91] Frank Bartling and Klaus Hinrichs. A plane-sweep algorithm for finding a closest pair among cnvex planar objects. Technical Report 91-03, Gesamthochschule Siegen, Institut für Informatik, 1991.

[BO79] J. Bentley and Thomas Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Trans. Comput.*, 28:643-647, 1979.

[BS76] J. Bentley and M. Shamos. Divide and conquer in multidimensional space. In *Proceedings 8th Annual Symp. Theory Comput.*, pages 220-230, 1976.

[For87] S. Fortune. A sweepline algorithm for Voronoi diagrams. *Algorithmica*, 2:153-174, 1987.

[HNS90] Klaus Hinrichs, J. Nievergelt, and Paul Shorn. Plane-sweep solves the closest pair problem elegantly. *Information Processing Letters*, pages 337-342, 1990.

[HS75] D. Hoey and M. Shamos. Closest-point problems. In *Proceedings 17th IEEE Annu. Symp. Found. Comput. Sci.*, volume 26, pages 151-162, 1975.

[Koe36] Paul Koebe. Kontaktprobleme der konformen Abbildung. *Berichte der Verhandlungen der Sächsischen Akademie der Wissenschaften zu Leipzig*, pages 141-164, 1936. Math.-Phys. Klasse 88.

[PR] William Pulleyblank and Günter Rote. Disk packings, planar graphs and combinatorial optimization. Unpublished.

[PS85] Franco Preparata and M. Shamos. *Computational Geometry: An introduction*. Springer-Verlag, New York, 1985.

[Sha85] Micha Sharir. Intersection and closest pair problems for a set of planar discs. *SIAM Journal on Computing*, 14:448-468, 1985.

[Yap87] Chee Yap. An $O(n \log n)$ algorithm for the Voronoi diagram of a set of simple curve segments. *Discrete Comput. Geom.*, 2:365-393, 1987.