UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* STEPS INTO COMPUTATIONAL GEOMETRY		5. TYPE OF REPORT & PERIOD COVERED Technical Report
		6. PERFORMING ORG. REPORT NUMBER R-760; UILU-ENG 77-2207
7. AUTHOR(s) F. P. Preparata, Editor		8. CONTRACT OR GRANT NUMBER(s) MCS76-17321 DAAB-07-72-C-0259
9. PERFORMING ORGANIZATION NAME AND ADDRESS Coordinated Science Laboratory University of Illinois at Urbana-Champaign Urbana, Illinois 61801		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Joint Services Electronics Program		12. REPORT DATE March, 1977 13. NUMBER OF PAGES 25
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Computational Geometry	Medial Axis
Analysis of Algorithms	Voronoi Diagrams
Computational Complexity	Diameter of Polygon
Point Location	Minimum Spanning Circle

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

This report is a collection of results in computational geometry which have been recently obtained by the Applied Computation Theory Group at the Coordinated Science Laboratory, University of Illinois at Urbana.

The format of this report is not uncommon in this branch of computational complexity; notable examples are "Problems in Computational Geometry" by M. I. Shamos [1], and "Excursions into Geometry" by Dobkin, Lipton, and Reiss [2]. One of the advantages of this anthological approach is that a large number of results can be timely disclosed; some of these results are

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

20. ABSTRACT (continued)

minor and yet may embody techniques which could prove very useful in this rapidly expanding area of research.

In this first "Notebook", results are presented on the problems of the minimum spanning circle and of the closest boundary point (medial axis) of a convex polygon, and the analogy between these problems and its common relation to sorting by selection are illustrated. In addition, a merge-type algorithm is illustrated for computing the medial axis of a convex polygon and its analogy to the Voronoi diagram problem is pointed out. It is also shown how the order-k Voronoi diagram construction can be profitably used to speed-up the solution of the "smallest bomb" problem discussed in [1].

It is also shown that the two closest vertices of a convex polygon with n vertices can be found in time $O(n)$ and that k points can be collectively located in a planar subdivision faster than they would be one at a time.

Finally, contrary to what had been hoped for sometime ago, we show that the construction of the Voronoi diagram on n points in three dimensions may require time $O(n^2)$.

UILU-ENG 77-2207

STEPS INTO COMPUTATIONAL GEOMETRY

by

F. P. Preparata, Editor

This work was supported in part by the National Science Foundation under Grant MCS76-17321 and by the Joint Services Electronics Program (U. S. Army, U. S. Navy and U. S. Air Force) under Contract DAAB-07-72-C-0259.

STEPS INTO COMPUTATIONAL GEOMETRY

F. P. Preparata[†], Editor

Foreword

This report is a collection of results in computational geometry which have been recently obtained by the Applied Computation Theory Group at the Coordinated Science Laboratory, University of Illinois at Urbana.

The format of this report is not uncommon in this branch of computational complexity; notable examples are "Problems in Computational Geometry" by M. I. Shamos [1], and "Excursions into Geometry" by Dobkin, Lipton, and Reiss [2]. One of the advantages of this anthological approach is that a large number of results can be timely disclosed; some of these results are minor and yet may embody techniques which could prove very useful in this rapidly expanding area of research.

In this first "Notebook", results are presented on the problems of the minimum spanning circle and of the closest boundary point (medial axis) of a convex polygon, and the analogy between these problems and its common relation to sorting by selection are illustrated. In addition, a merge-type algorithm is illustrated for computing the medial axis of a convex polygon and its analogy to the Voronoi diagram problem is pointed out. It is also shown how the order-k Voronoi diagram construction can be profitably used to speed-up the solution of the "smallest bomb" problem discussed in [1].

This work was supported in part by the National Science Foundation under Grant MCS76-17321 and by the Joint Services Electronics Program (U.S. Army, U.S. Navy and U.S. Air Force) under Contract DAAB-07-72-C-0259.

[†]Coordinated Science Laboratory and Department of Electrical Engineering, University of Illinois, Urbana, Illinois.

It is also shown that the two closest vertices of a convex polygon with n vertices can be found in time $O(n)$ and that k points can be collectively located in a planar subdivision faster than they would be one at a time.

Finally, contrary to what had been hoped for sometime ago, we show that the construction of the Voronoi diagram on n points in three dimensions may require time $O(n^2)$.

1. MINIMUM SPANNING CIRCLE (F. P. Preparata)

The minimum spanning circle (MSC) of a set S of n points in the plane is the circle of smallest radius containing all of the points. Algorithms for solving this problem have been proposed by Shamos [1].

One of these algorithms is based on first finding in time O(nlogn) the convex hull of S, and then in constructing the MSC of the resulting convex polygon. The latter operation is carried out by eliminating one vertex at a time, for a total running time $O(n^2)$.

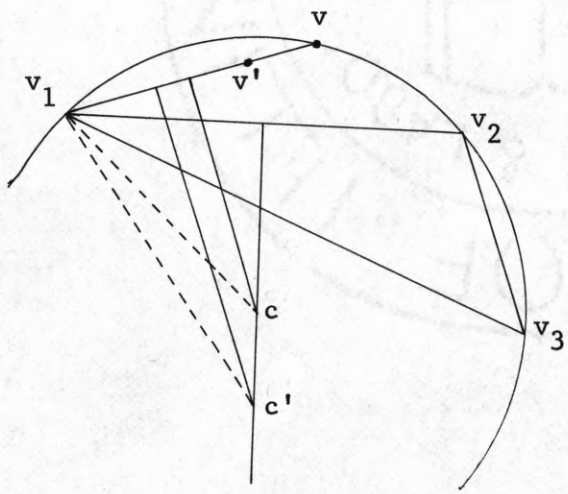
The second algorithm is based on the remark that the center of the MSC is a vertex of the farthest-point Voronoi diagram $V_{n-1}(S)$. In fact, a region K_i $(i=1,2,\ldots,m \leq n)$ of the subdivision induced by $V_{n-1}(S)$ is the locus of the points whose farthest member of S is some $P_i \in S$: therefore, a circle with center $C \in R_i$ passing through P_i contains all of the points in S. Thus, if the MSC is a 2-point circle, its center lies on an edge of $V_{n-1}(S)$, else it is a vertex of $V_{n-1}(S)$ (when three edges of $V_{n-1}(S)$ meet). Based on this idea, Shamos suggests to compute $V_{n-1}(S)$, which can be done in time O(nlogn), and to check the resulting O(n) Voronoi points; if one such point lies inside the triangle formed by its three determiners, that point is the center of the MSC, otherwise the diametral circle is minimum.

These two approaches can be combined into a single algorithm, which can be thought of as constructing either the MSC of S or $V_{n-1}(S)$. Moreover, the algorithm can be viewed as an application of sorting by selection, a technique which appears quite attuned to a number of geometric problems.

Assume, without loss of generality, that S is a convex polygon $\mathcal{P}$ with $n \geq 5$ vertices. A circle which passes through (at least) three vertices of $\mathcal{P}$ is called a <u>determined circle</u> of $\mathcal{P}$. We can now prove:

<u>Lemma</u>. The largest determined circle of $\mathcal{P}$ passes through three consecutive vertices of $\mathcal{P}$.

<u>Proof</u>: Let $\mathcal{C}$ be the largest determined circle of $\mathcal{P}$ and let v_1, v_2, and v_3 be its three determining vertices. The chords (v_1v_2), (v_2v_3), and (v_3v_1) determine three circular segments. Obviously $\mathcal{C}$ contains all the vertices of $\mathcal{P}$. If all of the remaining $n-3 \geq 2$ vertices of $\mathcal{P}$ lie in the same segment, then v_1, v_2, and v_3 are consecutive. Therefore, assume that the remaining vertices of $\mathcal{P}$ belong to at least two circular segments; clearly one of these two segments does not contain the center of $\mathcal{C}$. We will now show that if a circular segment not containing the center contains vertices of $\mathcal{P}$, then there is a determined circle of $\mathcal{P}$ larger than $\mathcal{C}$. Let (v_1v_2) be the chord of a circular segment A not containing the center c of $\mathcal{C}$ and let $v' \in A$ be a vertex of $\mathcal{P}$. Let v be the intersection of $\mathcal{C}$ and of the prolongation of (v_1v'), and let c (the center of $\mathcal{C}$) and c' be the intersections of the perpendicular bisector of (v_1v_2) and of the perpendicular bisectors of (v_1v) and (v_1v'), respectively. Obviously length (v_1c') > length (v_1c), i.e., the circle passing through v_1, v_2, and v' is a determined circle of $\mathcal{P}$ and is larger than $\mathcal{C}$. □



Therefore, we shall start by obtaining the largest determined circle $\mathcal{C}$ of the polygon $\mathcal{P}$. The center of this circle is the intersection of the perpendicular bisectors of two adjacent edges of $\mathcal{P}$; this intersection is a point of $V_{n-1}(\mathcal{P})$. Unless the triangle which determines $\mathcal{C}$ contains the center of $\mathcal{C}$, we eliminate the unique vertex which lies opposite to the center with respect to the chord formed by the other two vertices. Thus we obtain a polygon $\mathcal{P}'$ with one less vertex and, by a simple argument due to Shamos, it is guaranteed that the largest determined circle of $\mathcal{P}'$ contains all of the points of $\mathcal{P}$.

The data structure to be used is a tournament tree, each leaf of which is associated with a vertex of the polygon. In turn, the key associated with each vertex is the radius of the circle determined by v and by its two adjacent vertices. The tournament selects the vertex with the largest associated radius. Thus at each iteration three leaves are deleted and two new leaves are inserted: specifically, if $(v_0,v_1,\ldots,v_m)$ is the current vertex sequence and v_i is the winner of the tournament, the leaves associated with radius (v_{i-2},v_{i-1},v_i), radius (v_{i+1},v_i,v_{i+1}), and radius (v_i,v_{i+1},v_{i+2}) are deleted, while the new leaves radius $(v_{i-2},v_{i-1},v_{i+1})$ and radius $(v_{i-1}, v_{i+1}, v_{i+2})$ are inserted. Clearly, each iteration requires O(logn) operations and the algorithm terminates when there are only three or two vertices, for a total work O(nlogn) at most.

2. MEDIAL AXIS OF A CONVEX POLYGON (F. P. Preparata)

The medial axis M(G) of an arbitrary simple polygon G is the set of points internal to G which have more than one closest point on the boundary of G.

The medial axis of a convex polygon G is a tree which partitions the interior of G into regions. Each region is associated with an edge of G and is the locus of the points internal to G whose closest point on the boundary of G lies on that edge. For this reason the construction of the medial axis has been appropriately called by M. I. Shamos ([1], probl. POL9) the solution of the "closest boundary point" problem.

We shall now show that if G has n vertices, then M(G) is constructible in time O(nlogn). The algorithm to be considered is related to "sorting by selection."

Let (u,v) be an edge of G and let B(u) be the bisector of the angle at vertex u. We shall call C(u,v) the intersection of lines B(u) and B(v), and r(u,v) the distance of C(u,v) from (u,v). Notice that C(u,v) is the center of the circle tangent to (u,v) and its two adjacent edges.

Assume that $n \geq 4$ and $v_1v_2v_3v_4$ be four consecutive vertices of G. By removal of edge (v_2v_3) we define the operation of replacing G by the polygon G' constructed as follows:

(i) find the intersection v_{23} of the prolongations of (v_1v_2) and (v_3v_4);

(ii) replace vertices v_2 and v_3 by the single vertex v_{23}. Clearly G' is a convex polygon with one less vertex than G. We now prove the following lemma:

Lemma: $\min(r(v_1,v_{23}),\ r(v_{23},v_4)) \geq \min(r(v_1,v_2),\ r(v_2,v_3),\ r(v_3,v_4))$.

Proof: Assume, without loss of generality, that $r(v_1,v_{23}) \leq r(v_{23},v_4)$. Clearly, $C(v_2,v_3)$ lies on $B(v_{23})$ (see figure 1). We now distinguish two cases depending upon the location of $C(v_2,v_3)$. Let $r \triangleq \min(r(v_1,v_2),\ r(v_2,v_3),\ r(v_3,v_4))$ and $r' \triangleq \min(r(v_1,v_{23}),\ r(v_{23},v_4))$.

(1) $C(v_2,v_3) \in [v_{23},\ C(v_1,v_{23})]$. In this case $r = r(v_2,v_3) \leq r' = r(v_1,v_{23})$, trivially (figure 1a);

(2) $C(v_2,v_3)$ belongs to the half line $[C(v_1,v_{23}),\infty)$. In this case $r = r(v_1,v_2)$ and $r(v_1,v_2) \leq r(v_1,v_{23}) = r'$ (figure 1b). □

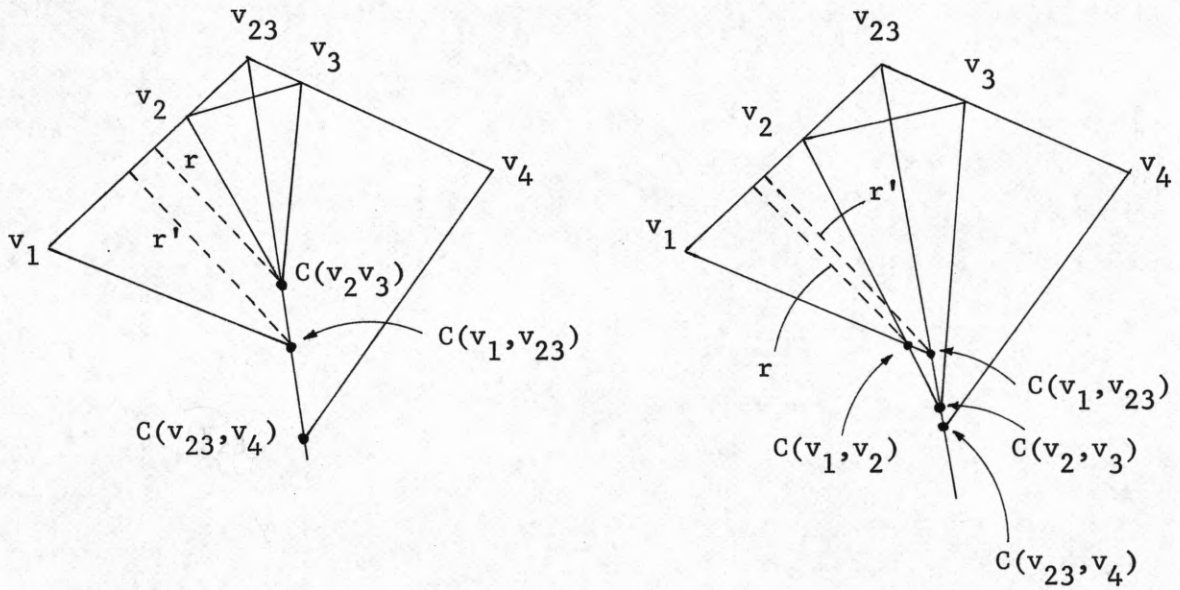


Figure 1 - Illustration of the proof of the lemma.

If we define r(G) as the min r(u,v) over all edges (u,v) in G, the previous lemma has the obvious consequence:

Corollary: If G' is obtained by edge removal from G, then $r(G) \leq r(G')$.

The reverse operation of edge removal will be called vertex cutting and, obviously, if G' is obtained from G by vertex cutting, then $r(G) \geq r(G')$. We can now prove the following theorem:

Theorem. If $r(u,v) = r(G)$, then $C(u,v)$ belongs to $M(G)$.

Proof: By contradiction. Assume that $C(u,v)$ does not belong to $M(G)$. Then there is an edge (u',v') which is closer to $C(u,v)$ than (u,v) (see figure 2). We now prolong (u,v) and (u',v') until they meet in a point w. Without loss of generality, assume that w is closer to v than to u and is also closer to v' than u'. Let G_0 be the polygon obtained by replacing the vertex sequence uv...v'u' with the vertex sequence uwu'. Clearly, by our original assumption,

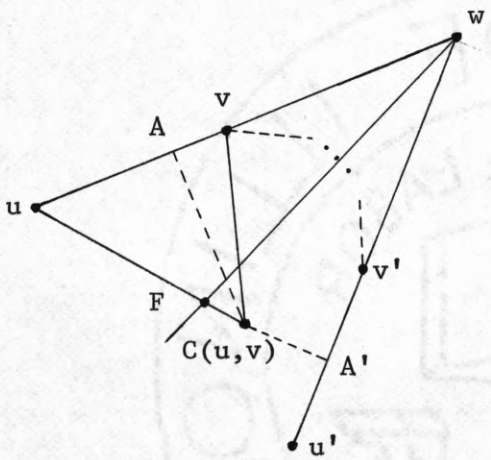


Figure 2.

length $(C(u,v)A) = r(u,v) >$ dist $(C(u,v),(u',v')) =$ length $(C(u,v)A')$, whence the bisector of angle $\sphericalangle$ uwu' intersects the segment $(u,C(u,v))$ in a point F. Obviously, since length $(uF) <$ length $(u,C(u,v))$, we conclude that $r(u,w)$ in G is less than $r(u,v)$ in G, i.e., $r(G_0) \leq r(u,w) < r(u,v)$. But we may

think of obtaining G from G_0 through a sequence of polygons $G_0, G_1, \ldots, G_k = G$, where G_i is obtained from G_{i-1} $(1 \leq i \leq k)$ by vertex cutting. Thus, by the previous results

$$r(G_0) \geq (G_1) \geq \ldots \geq r(G_k),$$

i.e., $r(u,v) > r(G_k) = r(G)$, violating the theorem hypothesis that $r(u,v) = r(G)$. □

On the basis of this theorem we can now outline a recursive algorithm for constructing M(G) of G.

Algorithm M(G)

Input: G, sequence of vertices, and T(G), tournament tree of r(u,v), for every $(u,v) \in G$.

Output: M(G).

1. Find (u,v) such that r(u,v) is minimum.
2. $\{G - (u,v), T(G - (u,v))\} \leftarrow$ REDUCE (T(G),(u,v))
3. M(G) ← COMBINE ((u,v),M(G - (u,v))).

The initial preparation of T(G) clearly requires time O(n). Since T(G) is available, Step 1 requires constant time. Step 2 consists in updating both the polygon and its corresponding tournament tree; since the latter involves three updates, for each edge removed we have work O(logn). Finally, Step 3 consists of the insertion of C(u,v) in the medial axis M(G-(u,v)) of G - (u,v). We conclude that the bulk of the work is done in Step 2, for a total of O(nlogn) operations.

It is also interesting to note that the same technique is applicable to the problems of constructing the nearest and farthest points Voronoi diagrams (see Section 1 of this report).

3. AN ALTERNATE METHOD FOR FINDING THE MEDIAL AXIS OF A CONVEX POLYGON

(D. T. Lee)

We shall now present an alternate method for finding the medial axis M(G) of a convex polygon G using a "divide and conquer" technique similar to that used in the construction of the Voronoi diagram of n points [1],[3].

Let the convex polygon G be given as a sequence of edges $(e_1,e_2,\ldots,e_n)$. Practically without loss of generality, we assume that n, the number of edges, is a power of 2. Divide the sequence of edges into two disjoint subsequences, L and R, each consisting of n/2 consecutive edges, i.e., $L = (e_1,e_2,\ldots,e_{n/2})$ and $R = (e_{n/2+1},\ldots,e_n)$. Let L_G be the boundary of the convex region formed by prolonging the first and last edges of the subsequence $(e_1,\ldots,e_{n/2})$ to infinity. R_G is defined similarly. Let B(i,j) be the bisector of the angle formed by the edges (or their prolongations) e_i and e_j, where $1 \le i, j \le n$. In other words, B(i,j) is the medial axis of the two edges e_i and e_j. Suppose the medial axes $M(L_G)$ and $M(R_G)$ have been obtained. If $M(L_G)$ and $M(R_G)$ can be merged in linear time to form M(G), then splitting the sequence of edges recursively will yield an O(n log n) algorithm.

We shall construct a polygonal line S, starting with B(n/2,n/2+1) and ending with B(1,n), with the property that any point to the left of S (oriented in the direction as we proceed) is closest to some edge in L_G and any point to the right of S is closest to some edge in R_G. Thus, after we have constructed S, the portion of $M(L_G)$ that is to the right of S and the portion of $M(R_G)$ that is to the left of S can be discarded and the resultant diagram is obtained.

Referring to Figure 1, where $L = (e_1,\ldots,e_6)$, $R = (e_7,\ldots,e_{11})$ and $M(L_G)$ and $M(R_G)$ are shown in dotted and dashed lines respectively, we start with B(6,7). When we meet B(5,6), we are closer to edge e_5 than to edge e_6. Therefore, we move off along B(5,7). Since at each step we are in two convex regions, one being associated with an edge in L_G and the other being associated with an edge in R_G we have to determine which of two candidate edges the polygonal line S intersects first, and then decide the direction in which we are to proceed. The process terminates when the direction of motion coincides with the angular bisector B(1,11). Since the total amount of work involved is proportional to the number of edges and the number of "turning" points on S, which is linear in n, we have obtained an O(n log n) algorithm. Detailed description of the merge process can be found in [3].

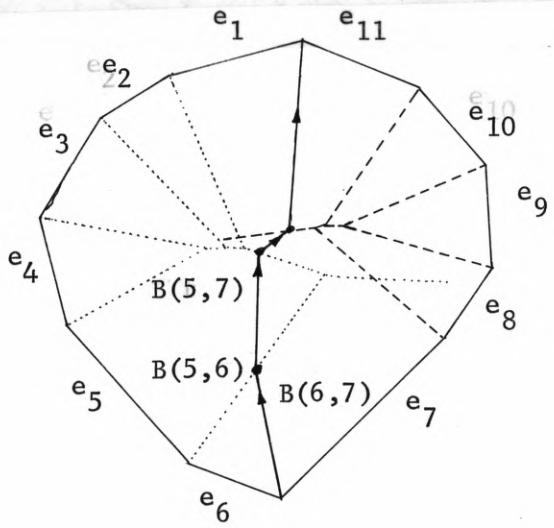


Figure 1. Merge of two medial axes.

4. THE SMALLEST BOMB PROBLEM (D. T. Lee)

Given n points in the plane, find the smallest circle that encloses at least k of the points. This also can be restated as: given n cities of equal strategic importance, determine the smallest bomb and the location to drop it so that it will destroy at least k of them [1].

Shamos [1] proposed an $O(n^4)$ algorithm that solves the problem for all values of k. But for a fixed value of k there is no better algorithm known to date. We shall present an $O(k^2 n \log n)$ algorithm for a fixed value of k. Furthermore, the same algorithm with some modification can solve the problem in at most $O(n^3 \log n)$ for all values of k.

The fact that this problem is related to the well-known k-nearest neighbor problem in the Euclidean plane makes the improvement possible. The k-nearest neighbor problem consists in determining among a set of n points in the Euclidean plane the k nearest neighbors to a given test point. To illustrate this relationship we shall introduce the notion of a very useful geometric construct, namely, the Voronoi diagram of order k for a set of n points. The Voronoi diagram of order k (or order k diagram for short in the following discussion) is a generalization of the classical Voronoi diagram. A detailed description can be found in [3], [4]. Figure 1 shows the classical Voronoi diagram (of order 1) for a set of 8 points in which for example, the cross lined region is the locus of points closest to point p_3. Figure 2 is the order 2 diagram for the same set of points in which each region is associated with two points and is the locus of points closer to one of the associated points than to any other point. In general, in an order k diagram, each region is associated with some

subset, of cardinality k, of the given set of points and is the locus of points closer to one of the points in the subset than to any other point not in the subset. An iterative algorithm for constructing the order k diagram has been developed which runs in time at most $O(k^2 n \log n)$ [3].

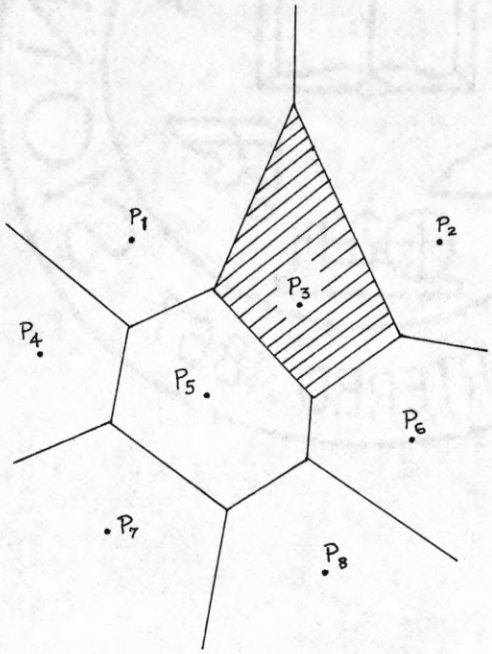


Figure 1. Voronoi diagram for a set of 8 points.

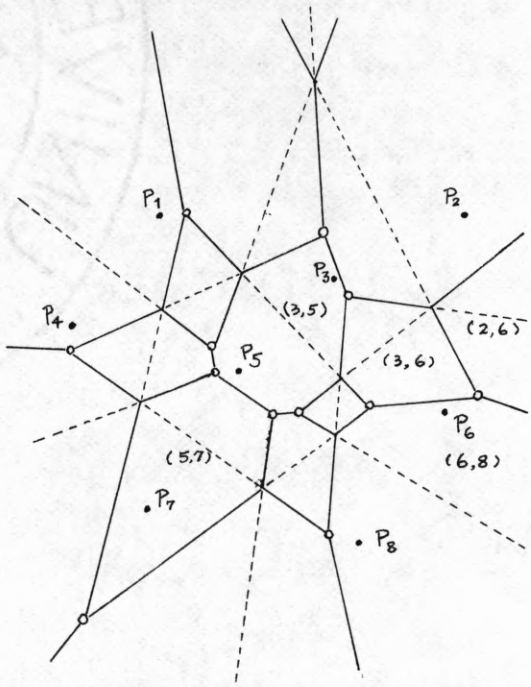


Figure 2. Order 2 diagram for the same set of 8 points in Fig. 1.

It can be seen that each Voronoi point in the order k diagram is a circumcenter of some three points (it is assumed that no more than 3 points are cocircular) and that the circumcircle thus determined contains either k-1 or k-2 points in its interior. The Voronoi point, whose

corresponding circumcircles contain k-1 points in the interior are called new Voronoi points. (Those points denoted by "$\circ$" in Figure 2.) The remaining are called old Voronoi points. The number of new Voronoi points in the order k diagram has been shown to be $O(k\,n)$ [3]. We shall be interested in the identification of the set of new Voronoi points of the order k-2 diagram only, since their corresponding circumcircles enclose exactly k points. (i.e. k-3 points in the interior plus three points on the circle.)

Next we shall show that the smallest circle that encloses at least k points must be the circle that encloses exactly k points. Suppose the smallest circle enclosed $m > k$ points. There always exists a circle of smaller radius that encloses $m - 1 \geq k$ points therefore contradicting to the assumption. By a theorem of Rademacher and Toeplitz [5] the center of the smallest circle must lie either inside the triangle formed by the three determiners or on a line segment determined by two points as a diameter. Thus, for some fixed number k, the smallest bomb problem can be solved as follows.

Input: A set of n points $\{p_1,\ldots,p_n\}$ in the plane given as ordered pairs (x_i,y_i) where x_i and y_i are the x- and y-coordinates of the point p_i respectively and an integer k.

Output: The center of the smallest circle enclosing k points and its radius.

1. Construct the order-(k-2) diagram for the given set of points and obtain the set of new Voronoi points $V = \{v_1, v_2, \ldots, v_s\}$.
2. Let the radius of the smallest circle be r. Initially, $r \leftarrow \infty$.
3. For each point $v_i \in V$, $1 \leq i \leq s$ do:
4. begin If the circumcircle centered at v_i has radius $r_i < r$ then set $r \leftarrow r_i$ and center $\leftarrow v_i$.
5. If v_i lies outside the triangle formed by the three determiners of the circumcircle then do:
6. begin Find the diameter d of the set of the k points enclosed by the circumcircle.
7. If $d < 2r$ then set $r \leftarrow \frac{1}{2}d$ and update center.

end

end

Step 1 takes $O(k^2 n \log n)$ time. Step 6 takes $O(k \log k)$ time and is executed at most s times. Since s, the number of new Voronoi points in the order k-2 diagram, is upper bounded by $O(kn)$, the total running time is $O(k^2 n \log n)$. □

Thus, we have the following theorem:

Theorem 1: The smallest bomb enclosing $k \geq 3$ points of the given N points can be determined in $O(k^2 n \log n)$ time.

We remark here that for $k = 2$ the closest-point algorithm [4] can solve the problem in $O(n \log n)$ time. For all other values of k, i.e. $3 \leq k \leq n$ the problem can be solved in two steps.

1. For each k, where $3 \leq k \leq n$, construct the order-(k-2) Voronoi diagram and find the smallest circle determined by 3 points with radius r_k and center c_k.

2. For each circle determined by any two points as a diameter, count the number of points enclosed. If i points are enclosed, compare the radius r of the circle with r_i. If it is smaller than r_i, then set $r_i \leftarrow r$ and update the center c_i, otherwise repeat this step.

Step 1 takes $O(n^3 \log n)$ time. Since there are $\binom{n}{2}$ circles determined by any 2 points of the given set of n points and counting the number of points enclosed requires $O(n)$ time, step 2 takes $O(n^3)$ time. Thus $O(n^3 \log n)$ time is sufficient.

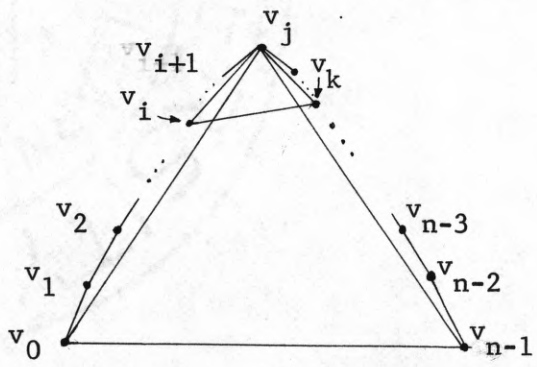
Theorem 2: The smallest bomb enclosing k points of the given n points in the plane for $2 \leq k \leq n$ can be solved in $O(n^3 \log n)$ time.

5. CLOSEST PAIR OF VERTICES OF A CONVEX POLYGON (D. T. Lee)

The problem of finding the closest pair of n points in the plane has been solved by Shamos [6] and the running time O(n log n) is optimal. However, the lower bound cannot be applied to the problem of finding the closest pair of vertices of a given convex polygon [1]. Since an obvious lower bound is O(n), one would suspect the existence of an algorithm which is more efficient then that given in [6]. We shall show that an optimal solution to this problem indeed exists. It is conceivable that the closest pair of vertices of a convex polygon need not be adjacent to each other. This is the fact that makes the problem more difficult to solve than expected. But, on the other hand, the property of convexity of the polygon does make the lower bound O(n) achievable.

Lemma. If the diameter of the convex polygon coincides with an edge then the two closest vertices are adjacent.

Proof. Let the polygon P be denoted by a sequence of vertices $v_0, v_1, \ldots v_{n-1}$ such that $\overline{v_i v_{i+1}}$ is an edge, and $d(v_0, v_{n-1})$ is the diameter. Suppose v_i, v_k are the closest pair of vertices and are not adjacent, i.e., $k > i+1$. There exists a vertex v_j, $i < j < k$ such that $d(v_i v_k) \leq \min\big(d(v_i, v_j), d(v_j, v_k)\big)$



Thus the angle $\sphericalangle\, v_i v_j v_k$ must be less than or equal to 60°. By convexity, the vertices

$v_1, v_2, \ldots, v_{j-1}$, and $v_{j+1}\ v_{j+2}, \ldots, v_{n-2}$ must lie above the chords $v_0\ v_j$ and $v_j\ v_{n-1}$ respectively. The angle $\measuredangle\, v_0 v_j v_{n-1}$ must be less than 60^o, which contradicts the assumption that $d(v_0, v_{n-1})$ is the diameter of the polygon. □

With the above lemma, one can find the closest pair of vertices of the convex polygon in $O(n)$ time as follows:

(i) Find the diameter of the polygon. Let it be $d(v_p, v_q)$ where $p < q$. Then the diameter divides the set of vertices into two chains $C_1 = \{v_p, v_{p+1}, \ldots, v_{q-1}, v_q\}$ and $C_2 = \{v_q, \ldots, v_{n-1}, v_0, \ldots, v_p\}$.

(ii) Scan, respectively, the two chains of vertices and find the nearest pairs of vertices. Let $\delta_1 = d(v_s, v_{s+1})$, $\delta_2 = d(v_t, v_{t+1})$ be the closest pairs of vertices respectively for the two convex polygons. Let $\delta = \min(\delta_1, \delta_2)$

If the distance of the closest pair of vertices is less than δ then the two vertices v_i, v_j must be in different chains, i.e., $v_i \in C_1$, $v_j \in C_2$.

(iii) Using the method given in [6] for finding the closest pair of points, we can determine the two closest pairs of vertices in time at most $O(n)$.

Since each step ((i), (ii) or (iii)) takes $O(n)$ time, we have

Theorem: The closest pair of vertices of a convex polygon can be found in $O(n)$ time, which is optimal.

6. LOCATION OF A SET OF POINTS IN A PLANAR SUBDIVISION (F. P. Preparata)

Problem. Given a subdivision determined by a planar straight line graph G with n vertices and a set S of k target points, for each point $P_i \in S$ determine to which region of the subdivision it belongs.

The elements of S can be located in the subdivision in time O(kn) by a brute force approach, which for each point $P_i \in S$ tests its inclusion in each of the regions of the subdivision. This is accomplished by testing, for any given region R of the subdivision, on which side of each boundary edge of R any selected target point lies. Since, due to planarity, the number of edges is O(n) and there are k target points, it is immediate to conclude that the sketched algorithm runs in time O(kn).

Alternately, one may use the point location algorithm due to Lee and Preparata [7], which requires a preprocessing time O(nlogn). With this procedure each target point can be located in time $O\left((\log n)^2\right)$, thus obtaining the conclusion that the total location work for S does not exceed $O(n\log n) + O\left(k(\log n)^2\right)$. Clearly this approach is preferable to the naive one anytime $\lim_{n \to \infty} \frac{\log n}{O(k)} = 0$. However, for large k, typically when k is O(n), there is a still faster method, which is a variant of the Lee-Preparata method and which we shall now describe.

Suppose we have proprocessed as in [7] the given planar straight-line graph G and obtained a complete ordered set $\mathcal{C} = (c_1, c_2, \ldots, c_m)$ of monotone chains for G. We recall that the members of $\mathcal{C}$ are also hierarchically ordered in a rooted binary tree T, which describes with its paths the sequences of chain discriminations which may occur in a point location search. For example for G as given in figure 1a, the

set $\mathcal{C}$ of chains is given in figure 1b and the tree T is illustrated in figure 1c. Notice that if we choose as the root of T the chain whose index is $2^{\lfloor \log_2 m \rfloor}$, assign all indices

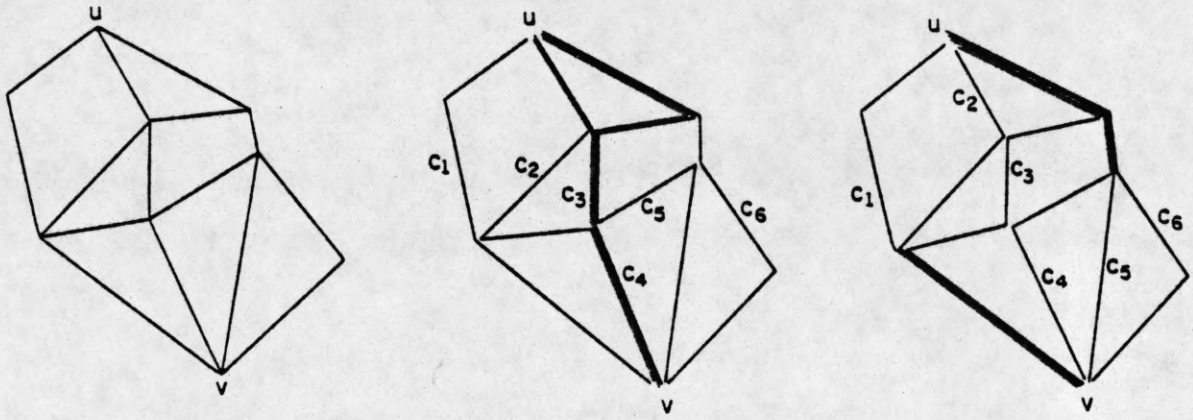


Figure 1. Examples of G, $\mathcal{C}$ and T. FP-4696

smaller than $2^{\lfloor \log_2 m \rfloor}$ to the left subtree and all the others to the right subtree, and adopt an analogous criterion for each vertex, tree T can be constructed so that the left subtree of each nonleaf vertex is a full binary tree. As was shown in [7], this organization of $\mathcal{C}$ enables us to list each edge e of G only once; specifically if e belongs to each of chains $\{c_j, c_{j+1}, \ldots, c_k\} \triangleq \mathcal{C}'$, it will be listed only in the chain $c' \in \mathcal{C}'$ which is closest to the root of T. We also assign to edge e a pair of integers $(I_{min}[e], I_{max}[e]) = (j,k)$ and the pair of names $(L[e], R[e])$ of the two regions bordering with e.

It is now rather simple to construct a recursive procedure for the location of the set S. As in the single-point algorithm described in [7], with each $P \in S$ we associate a triplet of parameters $(R(P); \ell(P), r(P))$, where R(P) is the region to which P is tentatively assigned, and $\ell(P)$ and r(P) are integers denoting that P is comprised between c_ℓ and c_r in $\mathcal{C}$. When $r(P) - \ell(P) = 1$, then $P \in R(P)$. Initially, for each $P \in S$, we set $\ell(P) = 0$ and $r(P) = m+1$. The location procedure makes use of a function, PARTITION (U,c), where U is a set of points and c is a chain in $\mathcal{C}$. This subroutine partitions U into two subsets U' and U", which respectively contain the points of U lying to the left and to the right of c. PARTITION (U,c) relates

to a merge algorithm as the "discrimination of a point against a chain", described in [7], relates to a binary search algorithm. Notationally for a point P, y(P) denotes its ordinate; for an edge e of G, y'(e) and y"(e) denote the ordinates of the upper and lower extremes of e. Practically without loss of generality, we assume that $\max_e y'(e) \geq y(P_i) \geq \min_e y''(e)$, for any $P_i \in U$.

procedure PARTITION (U,c)

Input: a list U: $(P_1,P_2,\ldots,P_t,P_{t+1})$ where $i < j \Rightarrow y(P_i) \geq y(P_j)$

a list c = $(e_1,e_2,\ldots,e_s,e_{s+1})$ where $h < \ell \Rightarrow y''(e_h) \geq y'(e_\ell)$

P_{t+1} and e_{s+1} are dummy sentinels, with $y(P_{t+1}) = y'(e_{s+1}) = y''(e_{s+1}) = -\infty$.

```
 1. k ← i ← j ← 1, U' ← U" ← ∅.
 2. While k ≤ t+s do
 3.   begin If   y(P_i) > y'(e_j) then
 4.              If ℓ(P_i) ≥ index(c) then U" ← U" ∪{P_i}, else U' ← U' ∪{P_i}
 5.              i ← i+1
 6.        else If   y(P_i) < y"(e_j) then j ← j+1
 7.              else If P_i lies to the right of e_j then U" ← U" ∪{P_i},
                      ℓ(P_i) ← I_max[e_j], R(P_i) ← R[e_j]
 8.                   else U' ← U' ∪{P_i}, r(P_i) ← I_min[e_j], R(P_i) ← L[e_j]
 9.                   i ← i+1
10.        k ← k+1
      end
11. return {U',U"}
```

It is easily verified that PARTITION (U,c) runs in time proportional to t+s = |U|+|c|.

We can now describe the location procedure, where T(c) denotes the subtree T whose root is c ∈ T.

LOCATE (S,T)

Input: S,T
Output: a set $K = \{(P,R(P)) \mid P \in S, R(P) = $ a region of the subdivision containing P$\}$

```
1. begin K ← ∅
2.       If S = ∅ then return K
3.       else
4.           begin c ← ROOT(T)
5.                 (S',S") ← PARTITION (S,c)
```

```
6.                  If RIGHTSON (c) ≠ Λ then K' ← LOCATE (S',T(RIGHTSON(c)))
7.                     else K' ← {(P,R(P))|P ∈ S'}
8.                  If LEFTSON (c) ≠ Λ then K" ← LOCATE (S",T(LEFTSON(c)))
9.                     else K" ← {(P,R(P))|P ∈ S"}
10.                 K ← K ∪ K' ∪ K"
11.                 return K
12.           end
13. end
```

We now evaluate the performance of the described algorithm. The bulk of the computational work is performed in Step 5, and we have already noted that PARTITION (S,c) runs in time $O(|S|+|c|)$. Since the algorithm entails a visit of each vertex of T, we may view the total work as the sum of the works performed at the vertices of T. Specifically let $S(c) \subseteq S$ be the set of points to be discriminated against c. Thus the total computational effort is

$$O(\sum_{c\in T}|S(c)|) + O(\sum_{c\in T}|c|) \quad ;$$

but, by the construction of the data structure T (see [7]), $\sum_{c\in T}|c|$ equals the number of edges of G, i.e., it is O(n) due to the planarity of G. Moreover, since obviously $|S(\text{LEFTSON }(c))| + |S(\text{RIGHTSON }(c))| = |S(c)|$, at any given depth in T the sum of $|S(c)|$ is a constant and is equal to $|S| = k$. Since T has at most $\lceil \log_2 m \rceil$ levels, and m is at most O(n), we conclude that $\sum_{c\in T}|S(c)| = O(k\log n)$. It follows that the total location work, including preprocessing, is $O((n+k)\log n)$, whereas work $O((n+k\log n)\log n)$ would have been required by the original algorithm described in [7].

7. A NEAREST-POINT VORONOI POLYHEDRON FOR n POINTS MAY HAVE ($O(n^2)$) VERTICES (F. P. Preparata)

It is well-known that the nearest-point Voronoi diagram for n points in the plane has O(n) vertices and can be constructed with no more than O(nlogn) operations. It has also been conjectured that the nearest-point Voronoi partition of the three-dimensional space (for short, the Voronoi polyhedron) for n given points may be constructible with the same order of effort. We now disprove this conjecture by showing that the Voronoi polyhedron on n points may have as many as $O(n^2)$ vertices, whence $O(n^2)$ is also a trivial lower-bound to the construction time of such polyhedron.

Consider the following set of n points in 3-space, where n is chosen to be of the form n = 4s:

1) 2s of these points are the vertices of a regular polygon in the plane (x,y), and are conveniently given in polar coordinate as $(r, j\frac{\pi}{s})$, for some r and $j = 0,\ldots,2s-1$;

2) of the remaining 2s points, s lie on the positive part and s lie on the negative part of the z-axis. From the symmetry induced by the polygon, the portions of the Voronoi polyhedron contained in any of two cylindrical sectors $(j\pi/s,(j+1)\pi/s)$, for $j = 0,\ldots,2s-1$, are isomorphic; thus it suffices to consider any of these sectors, say, $(0,\pi/s)$. In the latter, the Voronoi vertices are contained in the plane passing through the axis z and having azimuth $\pi/2s$. The corresponding diagram is shown in figure 1, and it clearly contains O(s) Voronoi vertices. Since there are 2s such sectors, we conclude that the Voronoi polyhedron for the current example contains $O(s^2) = O(n^2)$ vertices. Obviously the numbers of faces and of edges are also $O(n^2)$.

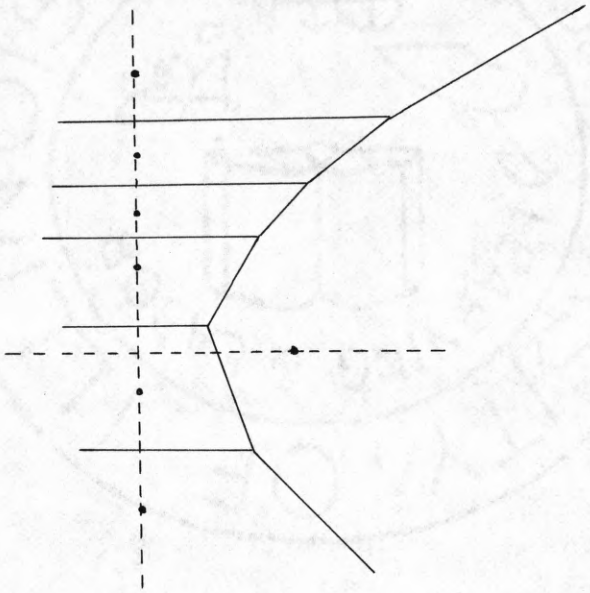


Figure 1. Voronoi vertices in the vertical plane of azimuth $\pi/2s$.

References

[1] M. I. Shamos, "Problems in Computational Geometry," Department of Computer Science, Yale University, New Haven, Conn., May 1975 (to be published by Springer Verlag).

[2] D. P. Dobkin, R. J. Lipton, and S. P. Reiss, "Excursions into Geometry," Rep. 71, Department of Computer Science, Yale University, New Haven, Conn., Aug. 1976.

[3] D. T. Lee, "On Finding k Nearest Neighbors in the Plane," Master Thesis, Coordinated Science Laboratory Report R-728, University of Illinois, Urbana, Illinois, May 1976.

[4] M. I. Shamos and D. Hoey, "Closest-Point Problems," Proc. 16th Annual IEEE Symposium on Foundations of Computer Science, 1975, pp. 151-162.

[5] H. Rademacher and O. Toeplitz, The Enjoyment of Mathematics, Princeton University Press, 1966, pp. 103-110.

[6] J. L. Bentley and M. I. Shamos, "Divide-and-Conquer in Multidimensional Space," Proceedings Eighth ACM Symposium on Theory of Computing, May 1976, pp. 220-230.

[7] D. T. Lee and F. P. Preparata, "Location of a Point in a Planar Subdivision and its Applications," Proceedings Eighth ACM Symposium on Theory of Computing, May 1976, pp. 231-235.