SIAM J. COMPUT.
Vol. 10, No. 1, February 1981

0097-5397/81/1001-0006 $01.00/0

GENERALIZATION OF VORONOI DIAGRAMS IN THE PLANE*

D. T. LEE† AND R. L. DRYSDALE, III‡

Abstract. In this paper we study the Voronoi diagram for a set of N line segments and circles in the Euclidean plane. The diagram is a generalization of the Voronoi diagram for a set of points in the plane and has applications in wire layout, facility location, clustering and contouring problems. We present an $O(N(\log N)^2)$ algorithm for constructing the diagram. It is an improvement of a previous known result which takes $O(Nc^{\sqrt{\log N}})$ time. The algorithm described in this paper is also shown to be applicable under a more general metric if certain conditions are satisfied.

Key words. Voronoi diagram, computational geometry, point location, computational complexity, divide-and-conquer, analysis of algorithms

1. Introduction.

The Voronoi diagram [18] for a set of N points in the Euclidean plane has been studied by a number of people [8], [11], [12], [17], [20], [22]. Essentially, a Voronoi diagram is a partition of the plane into N polygonal regions, each of which is associated with a given point. The region associated with a point is the locus of points closer to that point than to any other given point. Shamos and Hoey [21] present a divide-and-conquer algorithm which computes the diagram in $O(N \log N)$ time and show that the all nearest-neighbor problem [10], the largest empty circle problem and other seemingly unrelated problems can all be solved very efficiently once the diagram is available. They point out that such problems arise in wire layout [1], facility location [5], cutting-stock problems and geometric optimization problems [19], clustering problems [7] and contouring problems [2].

A natural question to ask is whether the algorithm for computing the Voronoi diagram can be generalized to other figures and metrics. These generalized Voronoi diagrams would have many applications. Lee and Wong [13] study the Voronoi diagram for a set of points under the L_1- and L_∞-metrics[1] and point out that these diagrams speed up retrieval algorithms for two-dimensional storage systems. Shamos [20] poses the problem of finding minimum weight spanning trees for circles and line segments.

These spanning trees can be computed very quickly if the Voronoi diagram for the circles and line segments is known. A country's territorial waters consist of its Voronoi region intersected with the locus of all points within 200 miles of a point in the country, so Voronoi diagrams are important when computing territorial waters. Many of the applications mentioned above have analogues where the objects considered are better represented by polygons, circles, or other geometric figures than by points.

In a previous paper [4] we presented an $O(Nc^{\sqrt{\log N}})$ algorithm for constructing the Voronoi diagram for circles and line segments in the Euclidean plane. We showed that the algorithm could also be used for more general figures and metrics if certain conditions were satisfied. We now present an $O(N(\log N)^2)$ algorithm for computing the Voronoi diagram for circles or line segments in the Euclidean plane. This algorithm can be applied under a more general metric if certain conditions are met.

* Received by the editors March 2, 1978, and in final form April 18, 1980.

† Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, Illinois 60201. The research of this author was supported in part by the National Science Foundation under grant MCS76-17321 and in part by the Joint Services Electronics Program under Contract DAAB-07-72-0259.

‡ Department of Mathematics, Dartmouth College, Hanover, New Hampshire 03755. The research of this author was supported by the National Science Foundation under grant MCS77-05313.

[1] F. Hwang [9] also studies the Voronoi diagram for a set of points under the L_1-metric and independently discovers an $O(N \log N)$ time algorithm for constructing it.

2. Preliminaries. We begin by introducing some definitions and notations.

DEFINITION 1. A *closed line segment* $\overline{a, b}$ consists of two *endpoints* a and b, and a straight-line portion which is denoted by (a, b) and referred to as an *open segment*, or briefly a *segment*. Points or segments are called *elements*. The straight line containing $\overline{a, b}$ is denoted by $\overleftrightarrow{a, b}$. The same line directed from a to b is denoted by $\overrightarrow{a, b}$.

DEFINITION 2. The *projection* $p(q, \overline{a, b})$ of a point q onto a closed segment $\overline{a, b}$ is the intersection of the line $\overleftrightarrow{a, b}$ and the line perpendicular to $\overline{a, b}$ and passing through q.

DEFINITION 3. The *distance* $d(q, \overline{a, b})$ between a point q and a closed segment $\overline{a, b}$ in the Euclidean metric is defined as the distance $d(q, p(q, \overline{a, b}))$ between the point q and its projection onto $\overline{a, b}$ if $p(q, \overline{a, b})$ belongs to $\overline{a, b}$ and is MIN $(d(q, a), d(q, b))$ otherwise. In other words, $d(q, \overline{a, b}) = \text{MIN}_{u \in \overline{a,b}}\, d(q, u)$. The point of $\overline{a, b}$ which is closest to q is called the *image* $I(q, \overline{a, b})$ of q on $\overline{a, b}$.

DEFINITION 4. The *bisector* $B(e_i, e_j)$ of two elements e_i and e_j is the locus of points equidistant from e_i and e_j. The bisector $B(X, Y)$ of two sets of elements X and Y is defined to be the locus of points equidistant from X and Y, where the distance $d(q, X)$ between a point q and a set of elements X is defined to be $\text{MIN}_{e \in X}\, d(q, e)$. The bisector $B(e_i, e_j)$ is said to be *oriented* if a direction is imposed upon it so that elements e_i and e_j lie to the *left* and to the *right* of it respectively. An oriented bisector $B(X, Y)$ is defined similarly.

For example, in Fig. 1 the bisector $B(q, \overline{a, b})$ of a point q and a closed segment $\overline{a, b}$ has three components, i.e., $B(q, a)$, $B(q, b)$, and a portion of the parabola[2] whose focus and directrix are the point q and the line $\overleftrightarrow{a, b}$ respectively. Fig. 2 shows the bisector $B(\overline{a, b}, \overline{c, d})$ of two closed segments $\overline{a, b}$ and $\overline{c, d}$, where one of the components is a portion of the angular bisector of the angle formed by the lines $\overline{a, b}$ and $\overline{c, d}$. In what follows we shall also consider the bisector $B(q, (a, b))$ of a point q and a segment (a, b) to be the same as $B(q, \overline{a, b})$. Similarly, the bisector $B((a, b), (c, d))$ of two segments (a, b) and (c, d) is the same as $B(\overline{a, b}, \overline{c, d})$. We define the bisector $B(a, \overline{a, b})$ to be a line perpendicular to $\overline{a, b}$ and passing through a.

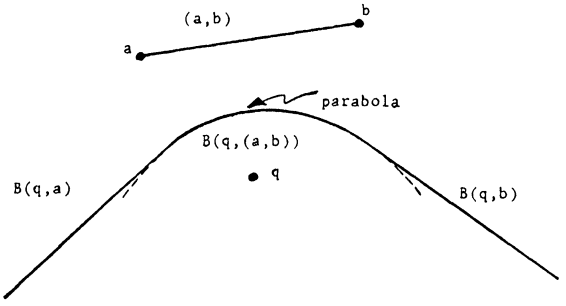

FIG. 1. *Bisector of a point and a closed segment.*

With these definitions we can define the half-plane $h(e_i, e_j)$ as the locus of points closer to element e_i than to element e_j. The *Voronoi region* $V(e_i)$ is the intersection of all the half-planes containing e_i; i.e., $V(e_i) = \cap_{j \neq i} h(e_i, e_j)$, which is the locus of points

[2] We shall use the term parabola to mean the portion of the parabola whose projection belongs to the segment.

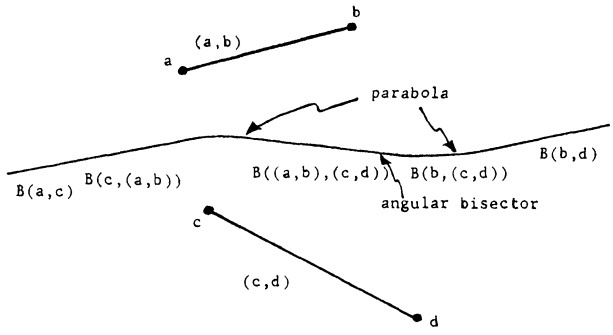

FIG. 2. *Bisector of two closed segments.*

closer to e_i than to any other element. The boundary edges of a Voronoi region are called *Voronoi edges*, and vertices of the region are called *Voronoi points*. Finally, we can define the Voronoi diagram $V(S)$ for a set S of disjoint closed segments $\{s_1, s_2, \cdots, s_n\}$ as the union of Voronoi regions $V(s_i)$; each $V(s_i)$ in turn is the union of three Voronoi regions, each associated with an element of the closed segment s_i (two endpoints and an open segment). Fig. 3 shows the Voronoi diagram for two closed segments. Since

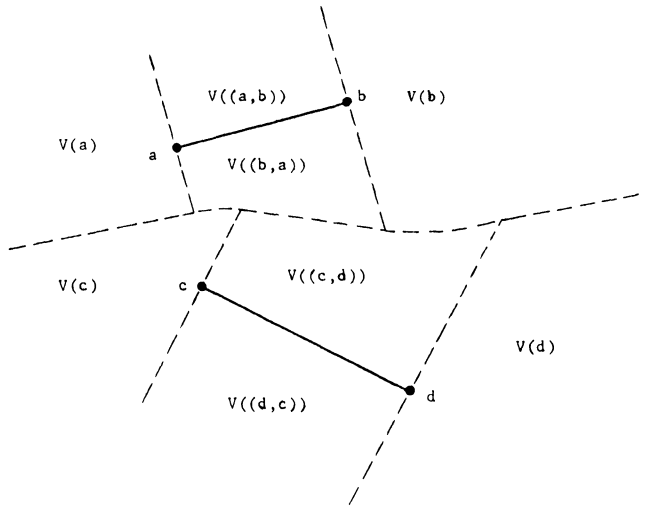

FIG. 3. *Voronoi diagram for two closed segments.*

associated with each segment there are two regions (one lying on each side of the segment), for ease of reference we shall denote the region that lies to the left of the segment (a, b) directed from a to b by $V((a, b))$ and the other by $V((b, a))$. From now on we shall consider the closed segment $\overline{a, b}$ to be composed of *four* elements, (instead of *three*), i.e., two endpoints of a and b and two directed segments (a, b) and (b, a). As a consequence, the Voronoi region $V(\overline{a, b})$ associated with the closed segment $\overline{a, b}$ is then the union of $V(a)$, $V(b)$, $V((a, b))$ and $V((b, a))$. Unless otherwise stated, in what follows we shall consider the set S to be composed of elements; i.e., the Voronoi diagram $V(S)$ is a collection of Voronoi regions $V(e_i)$ which is the locus of points closer to element e_i than to any other element.

DEFINITION 5. A polygonal region R in the plane is *generalized-star-shaped* with *nucleus* C, $C \subseteq R$, if for any point $r \in R$ there exists a point $c \in C$ such that the line segment $\overline{r, c}$ lies completely within R.

DEFINITION 6. The *dual* $D(S)$ of the Voronoi diagram $V(S)$ for a set of N closed segments is a graph with $4N$ nodes each of which corresponds to an element of S; two nodes are connected by an edge if their associated Voronoi regions share a Voronoi edge. The segment (a, b) is considered a Voronoi edge bordering the regions $V((a, b))$ and $V((b, a))$. (Fig. 4 shows the dual graph of the diagram in Fig. 3.)

Now let us state the properties of the Voronoi diagram for closed segments. Assume that the given set S of elements is $\{e_1, e_2, \cdots, e_n\}$.

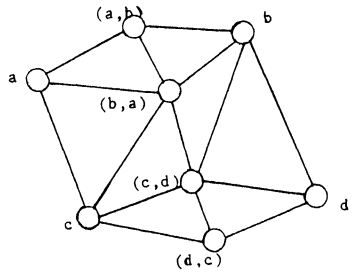

FIG. 4. *Dual graph of the diagram of Fig.* 3.

LEMMA 1. *Given a point* q, *if* $q \in V(e_i)$, *then* $\overline{q, q_i}$ *lies completely in* $V(e_i)$, *where* q_i *is the image* $I(q, e_i)$ *of* q *on* e_i. *This shows that* $V(e_i)$ *is generalized-star-shaped with nucleus* e_i.

Proof. It suffices to show that any point z on $\overline{q, q_i}$ must belong to $V(e_i)$. Suppose $z \in V(e_j)$, i.e., $d(z, e_j) < d(z, e_i)$. Let z_j be the image $I(z, e_j)$ of z on e_j. By the triangle inequality, $d(q, z_j) \leqq d(q, z) + d(z, z_j)$. Since $d(z, z_j) = d(z, e_j) < d(z, e_i)$, we have $d(q, z_j) < d(q, z) + d(z, e_i)$. Now $d(z, e_i) = d(z, q_i)$ and $d(q, z) + d(z, q_i) = d(q, q_i)$ imply that $d(q, z_j) < d(q, q_i) = d(q, e_i)$. Since $d(q, e_j) \leqq d(q, z_j)$ by Definition 3, we have $d(q, e_j) < d(q, e_i)$ which contradicts the assumption that $q \in V(e_i)$. □

LEMMA 2. *The Voronoi regions* $V(e_i)$ *and* $V(e_j)$ *share an edge if and only if there exists a point* q *such that the circle centered at* q *with radius* $d(q, e_i) = d(q, e_j)$ *does not contain any point of other elements in its interior or on its boundary.*

Proof. If $V(e_i)$ and $V(e_j)$ share an edge, then obviously the circle centered at any point q on the edge with radius $d(q, e_i) = d(q, e_j)$ will not contain any point of other elements. If the circle centered at q passes through the images $I(q, e_i)$ and $I(q, e_j)$ and does not include any point of other elements in its interior or on its boundary, then the nearest neighbor of q is either e_i or e_j; i.e., $\overline{q, I(q, e_i)}$ and $\overline{q, I(q, e_j)}$ lie entirely in $V(e_i)$ and $V(e_j)$ respectively. Therefore, $V(e_i)$ and $V(e_j)$ must share an edge. □

THEOREM 1. *Let* z_1 *and* z_2 *be two points on the boundary of* $V(e_i)$ *and* z_{1i} *and* z_{2i} *the images of* z_1 *and* z_2 *on* e_i, *respectively. Either one of the two closed segments* $\overline{z_1, z_{1i}}$ *and* $\overline{z_2, z_{2i}}$ *properly contains the other or they do not intersect except possibly at endpoints.* (z_{1i} *and* z_{2i} *may coincide.*)

Proof. Suppose that neither of $\overline{z_1, z_{1i}}$ and $\overline{z_2, z_{2i}}$ properly contains the other. Assume that they intersect at a point z. Without loss of generality we may assume that $d(z, z_{1i}) \leqq d(z, z_{2i})$ (Fig. 5a). We have $d(z_2, z_{2i}) = d(z_2, z) + d(z, z_{2i}) \geqq d(z_2, z) + d(z, z_{1i}) > d(z_2, z_{1i})$. Since z_{2i} is the image of z_2 on e_i, we have a contradiction. □

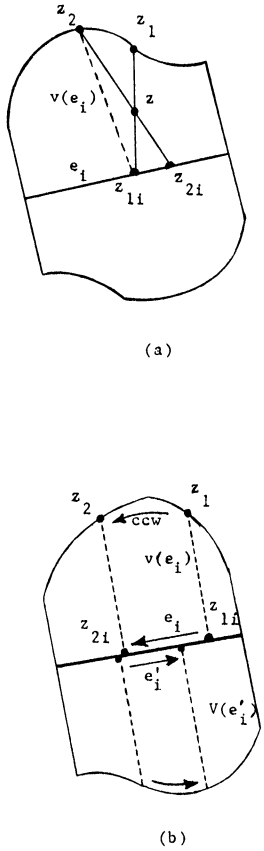

FIG. 5. *Illustration of the proof of Theorem* 1.

What this theorem implies is that if we move a point z along the boundary of $V(e_i)$ in a counterclockwise or a clockwise direction, then the image $I(z, e_i)$ also moves in the same direction along e_i. If the element e_i is a point, e_i itself is the image $I(z, e_i)$ for all $z \in V(e_i)$. Fig. 5b illustrates this phenomenon.

THEOREM 2. *The element e_i is on the convex hull* CH (S) *of S if and only if its associated Voronoi region is unbounded.*

Proof. We first prove that if an element e_i is on the convex hull, its associated Voronoi region is unbounded. Take a point z of e_i which is on the convex hull CH (S). z being on CH (S), there exists a supporting line $l(z)$ of CH (S) which passes through z with the property that all the elements including e_i lie on one side of the line. Consider the closed segment $\overline{z, t}$ perpendicular to $l(z)$ and lying outside of CH (S). We have $d(t, z) = \text{MIN}_{a \in e_i} d(t, a)$. Since the circle centered at t with radius $d(z, t)$ does not contain any other point of S in its interior, we have $d(t, z) = \text{MIN}_{a \in S} d(t, a)$, i.e., t must belong to the Voronoi region $V(e_i)$. Moreover, the endpoint t of the closed segment $\overline{z, t}$ can be extended arbitrarily, so the Voronoi region $V(e_i)$ is unbounded.

To prove the converse we shall show that for any element that is not on CH (S), its associated Voronoi region must be bounded. Suppose e_i is not on CH (S). Any line intersecting e_i will divide the set of elements into two nonempty subsets. Let l be a line intersecting e_i at a point z_i of e_i. Consider the line $\overline{z_i, z}$ perpendicular to the line l. The circle centered at z with radius $d(z, z_i)$ becomes the line l if we let $d(z, z_i)$

approach infinity. Therefore, when $d(z, z_i)$ is sufficiently large the circle will contain a point of some element e_j in its interior. That is, the nearest neighbor of z is not e_i. This shows that there are no half-lines contained in $V(e_i)$. But it is easy to show from Lemma 1 that an unbounded Voronoi region must contain a half-line. Therefore, $V(e_i)$ is bounded. □

THEOREM 3. *The dual graph $D(S)$ of the Voronoi diagram $V(S)$ for a set of $N \geqq 3$ elements is planar and has at most $3N-5$ edges and $2N-3$ faces.*

Proof. We first show that $D(S)$ is planar by exhibiting a planar embedding for it, and then obtain the bounds on the number of edges and faces by using Euler's Polyhedra Formula [6]. Let $D_i(S)$ be the set[3] of elements $\{e_{i1}, e_{i2}, \cdots, e_{im}\}$ whose associated regions and $V(e_i)$ share an edge. For any edge of $V(e_i)$ we can find a path connecting the two elements whose regions share the edge as follows. Let z_j be a point on the edge $\bar{B}(e_i, e_{ij})$, $j = 1, 2, \cdots, m$. By Lemma 1, the closed segments $\overline{z_j, I(z_j, e_i)}$ and $\overline{z_j, I(z_j, e_{ij})}$ lie entirely in $V(e_i)$ and $V(e_{ij})$, respectively. The path connecting e_i and e_{ij} consists of $\overline{I(z_j, e_i)}$, z_j and $\overline{z_j, I(z_j, e_{ij})}$. Note that if e_{ij} is an endpoint of the closed segment for e_i, the two closed segments $\overline{z_j, I(z_j, e_i)}$ and $\overline{z_j, I(z_j, e_{ij})}$ coincide. By Theorem 1, all the closed segment $\overline{z_j, I(z_j, e_i)}$, for $j = 1, 2, \cdots, m$, do not intersect except possibly at endpoints (on e_i). We conclude that the dual graph is planar.

Now, for the bounds on the number of edges and faces observe that in the dual graph each interior face is bounded by at least three edges and the exterior face is bounded by at least two edges when $N \geqq 2$. This is because two nodes n_i and n_j of $D(S)$ which correspond to elements e_i and e_j, respectively, have multiple edges only if the bisector $B(e_i, e_j)$ is broken into two or more pieces. The node n_k whose associated element e_k caused $B(e_i, e_j)$ to break into two pieces will appear on any interior face containing both n_i and n_j. Therefore our interior face must have more than two edges bounding it. If we sum over the number of edges bounding each face we will get at least $3*(f-1)+2$ edges, where f is the number of faces of $D(S)$. Since each edge appears on exactly two faces, we have $2e \geqq 3f-1$. Solving for e and plugging into Euler's Polyhedra Formula, $N-e+f=2$, we have $N-f/2 \geqq 3/2$. Therefore, $f \leqq 2N-3$ and $e \leqq 3N-5$. □

COROLLARY 1. *Given a set S of N elements, the number of Voronoi edges and the number of Voronoi points are both $O(N)$.*

LEMMA 3. *Given a set $S = \{s_1, s_2, \cdots, s_n\}$ of closed segments, let $D_i(S)$ denote the subset of closed segments of S whose Voronoi regions are adjacent to $V(s_i)$. Then there exists a closed segment $s_j \in D_i(S)$ such that $d(s_i, s_j) = \text{MIN}_{a \in S-\{s_i\}} d(s_i, a)$, i.e., $D_i(S)$ contains a nearest neighbor of s_i.*

Proof. Immediate from Lemma 2. □

3. Construction of the Voronoi diagram for a set of line segments.

3.1. Introduction. Now let us turn to the problem of computing the Voronoi diagram for a set of line segments. Our algorithm is a generalization of the divide-and-conquer scheme first presented by Shamos and Hoey [21] for a set $S = \{p_1, p_2, \cdots, p_N\}$ of N points. We therefore start by describing a version of this algorithm which includes a modification made by Lee [12]. Sort the N points lexicographically on their (x, y) coordinates. Let $L = \{p_1, p_2, \cdots, p_{\lfloor N/2 \rfloor}\}$ and let $R = \{p_{\lfloor N/2 \rfloor + 1}, \cdots, p_N\}$, where $\lfloor x \rfloor$ is the greatest integer in x. Then compute the Voronoi diagrams $V(L)$ and $V(R)$ recursively. If we can merge $V(L)$ and $V(R)$ in $O(N)$ time, we can compute $V(S)$ in $O(N \log N)$ time.

[3] The set in general is a multiset; i.e., the elements are not necessarily distinct.

The key to the merge step is the merge curve $B(L, R)$, which is the oriented bisector of the two sets L and R of points. (Fig. 6.) Every point to the left of $B(L, R)$ is closer to some point in L than to any point in R. Similarly, every point to the right of $B(L, R)$ is closer to some point in R than to any point in L. Therefore if we chop off the part of any region in $V(L)$ extending to the right of $B(L, R)$ and the part of any region in $V(R)$ extending to the left of $B(L, R)$, we end up with the diagram $V(S)$. We shall construct $B(L, R)$ in linear time as follows. Theorem 2 in the previous section suggests that we compute the convex hull CH (S). The convex hulls CH (L) and CH (R) are disjoint and can be constructed recursively when constructing $V(L)$ and $V(R)$. CH (S) can be computed by merging CH (L) and CH (R) in linear time using an algorithm invented by Preparata and Hong [15]. In merging CH (L) and CH (R) we create two new hull edges, each joining a point l in L and a point r in R. By Lemma 2 we know that $B(l, r)$ is a component bisector, called a *starting bisector.* We can imagine forming $B(L, R)$ by moving a point from infinity inward along the bisector $B(l, r)$. For example, in Fig. 6 we will move the imaginary point along $B(p_4, p_{10})$. The imaginary point follows $B(p_4, p_{10})$ until it meets an edge at $V(p_4)$. At this point it is closer to p_8 than to p_4, so it leaves $V(p_4)$ and enters the region $V(p_8)$ by following $B(p_8, p_{10})$ until it leaves $V(p_{10})$. In this way it traces out $B(L, R)$, always following the bisector $B(p_u, p_v)$ of a point p_u in L and a point p_v in R. When the imaginary point leaves $V(p_u)$, p_u is changed to the point associated with the new region entered. p_v is updated similarly.

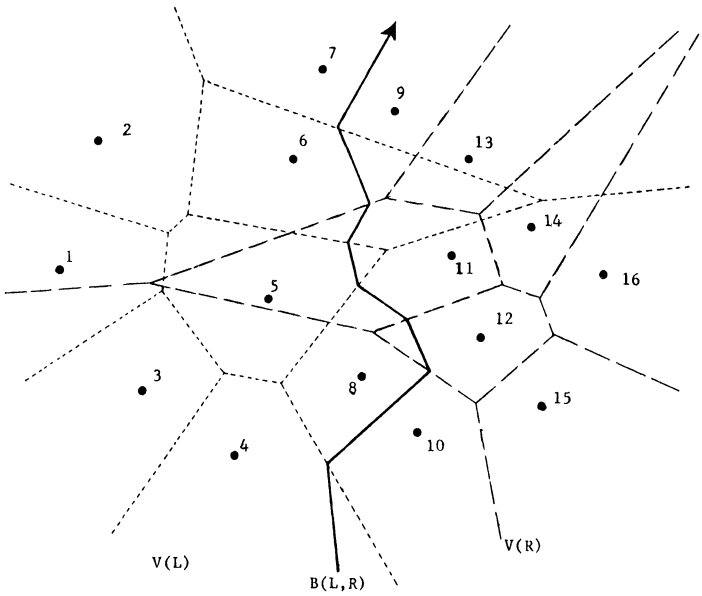

FIG. 6. $V(L)$, $V(R)$ *and* $B(L, R)$.

Because each $V(p_i)$ can have $O(N)$ edges on its boundary and we might examine the edges of a single region $O(N)$ times to see where a bisector first intersects the boundary of that region, the naive approach of checking each edge of $V(p_u)$ and $V(p_v)$ to find the first intersection with $B(p_u, p_v)$ could result in $O(N^2)$ work. We can avoid this by the following *scanning scheme* for edges: beginning with the point where $B(p_u, p_v)$ enters a region, scan the edges of a region in $V(L)$ in a counterclockwise (CCW) direction and the edges of a region in $V(R)$ in a clockwise (CW) direction. If an edge

does not intersect $B(p_u, p_v)$, then discard it. The following theorem shows that no backtracking is needed when using this scanning scheme.

THEOREM 4. *Let p_u be in L and p_v in R and let $B(p_u, p_v)$ be constructed during the merge process. No point of $V(p_u)$ in $V(L)$ which lies to the right of the oriented bisector $B(p_u, p_v)$ will be included in the region $V(P_u)$ in the final diagram. Similarly, no point of $V(P_v)$ in $V(R)$ which lies to the left of the oriented bisector $B(p_u, p_v)$ will be included in the region $V(p_v)$. Furthermore, scanning $V(p_u)$ in a* CCW *direction and $V(p_v)$ in a* CW *direction will find the first intersection between $B(p_u, p_v)$ and either $V(p_u)$ or $V(p_v)$.*

Proof. The fact that no point of $V(p_u)$ in $V(L)$ lying to the right of $B(p_u, p_v)$ can be included in the region $V(p_u)$ in the final diagram follows from the definition. To show that the CW-CCW scan indeed finds the first intersection point consider Fig. 7. Suppose t is the intersection point where $B(p_u, p_v)$ meets some edge of $V(p_v)$ before it meets any edge of $V(p_u)$. If t were not the first intersection, then moving a point z along the boundary of $V(p_v)$ in a CW direction we will find at least two intersection points with $B(p_u, p_v)$. Since $V(p_v)$ is generalized-star-shaped with nucleus p_v, by Theorem 1, this is impossible. Similar arguments hold if $B(p_u, p_v)$ meets an edge of $V(p_u)$ before any edge of $V(p_v)$. □

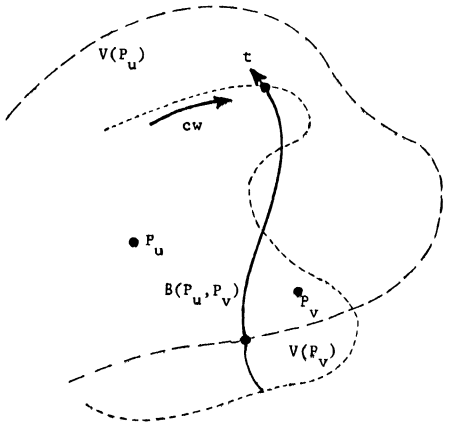

FIG. 7. *Illustration of the proof of Theorem* 4.

We now analyze the time taken for the merge process. After each test either an edge of $V(L)$ or an edge of $V(R)$ is discarded or a new Voronoi point is created in $V(S)$. Therefore, the time needed is proportional to the number of edges in $V(L)$ and $V(R)$ plus the number of Voronoi points on $B(L, R)$. By Corollary 1 this is $O(N)$. Therefore, the overall time needed to compute $V(S)$ is $O(N \log N)$.

What happens when we apply this algorithm to a set of closed segments? The first difficulty arises when ordering the closed segments. Lexicographical order is not defined for segments. Therefore, we arbitrarily choose the ordering of closed segments according to their left endpoints. Once we have chosen the ordering we can divide the given set S into two subsets L and R such that L and R contain the leftmost $\lfloor N/2 \rfloor$ and rightmost $\lceil N/2 \rceil$ closed segments respectively. Now we want to construct $V(L)$ and $V(R)$ recursively and merge them to form $V(S)$ as we did before. The first step in the merge process is to find the union of the convex hulls CH (L) and CH (R). But, because of the ordering that we have chosen, these two convex hulls are not necessarily disjoint (CH (L) may contain CH (R) in its interior). Therefore, using the idea of convex hull to

find the starting bisector may not help. Besides, the merge curve is not necessarily composed of a single piece. It may be broken into several pieces as shown in Fig. 8. That is, we have to determine the starting bisectors for all the pieces and then use the scanning scheme to construct all pieces of merge curves. The starting bisectors for the two unbounded pieces in Fig. 8 can be found by forming the union of CH (L) and

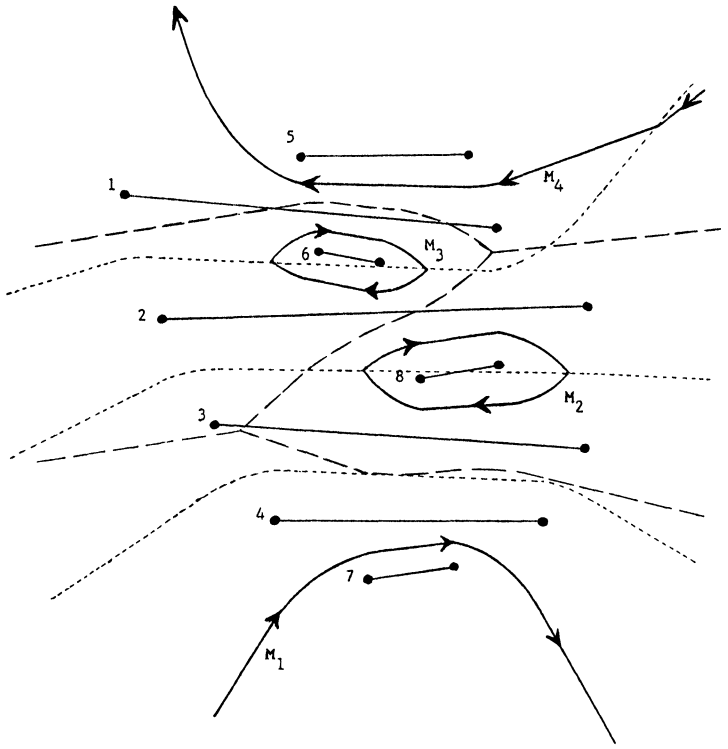

FIG. 8. *Merge curve is broken into four pieces.*

CH (R) just as we did before. However, the convex hull gives no information about the starting bisectors for the other two pieces of $B(L, R)$ that are closed curves, called *islands*. Islands result from the closed segments in R getting "trapped" in CH (L) and are a consequence of the initial ordering. The authors tried to find an ordering for the closed segments which would prevent these islands from forming. But finding this ordering seems to be as difficult as finding the Voronoi diagram itself. They also tried a number of approaches, but found no method for finding all islands in $O(N)$ time. However, the following method has been devised to determine a set of starting bisectors in $O(N \log N)$ time.

3.2. Determination of a set of starting bisectors. Suppose that we have obtained two Voronoi diagrams $V(L)$ and $V(R)$. To merge them we shall find a set, called *starter set* of L and R (denoted by ST (L, R)), of points (along with the information from which the corresponding starting bisectors can be derived) such that it is guaranteed that every piece of the merge curve passes through at least one of the points in ST (L, R). In other words, the intersection of the set of points on the merge curve and ST (L, R) is nonempty. The idea is based on the following observations. First each merge curve piece M_i encloses some subset R_i of closed segments of R. Either the subset R_i will be

completely surrounded by M_i (an island), or M_i and an edge "at infinity" will enclose R_i. A line from some segment in R_i to a segment not in R_i will cross M_i before it crosses any other piece of the merge curve. Second, M_i is the bisector $B(L_i, R_i)$ of R_i and some subset L_i of closed segments of L. It contains a bisector $B(e_j, e_k)$ for some elements $e_j \in L_i$ and $e_k \in R_i$ such that e_k is an endpoint of some closed segment of R_i. The first observation is due to the initial ordering of the set of closed segments. The second observation can be established as follows. Consider the convex hull CH (R_i) of the set R_i of closed segments enclosed by M_i. One of the hull vertices, say q_k, must be an endpoint of some closed segment l_k. The half-line emanating from the endpoint q_k and perpendicular to l_k must intersect M_i at some point which is equidistant from q_k and some element e_j of L_i. Therefore, the intersection of $B(e_j, q_k)$ and M_i is nonempty and the observation follows. Based on these two observations we can determine the starter set ST (L, R) as follows.

We construct the Voronoi diagram $V(Q)$ for the set Q of endpoints of R and then merge $V(Q)$ and $V(L)$ to form $V(L \cup Q)$. To merge $V(Q)$ and $V(L)$ we are confronted with the same problems as we were in merging $V(R)$ and $V(L)$. That is, the merge curve $M' = B(L, Q)$ is not necessarily composed of a single piece. We shall locate the set Q of points in the Voronoi regions of $V(L)$ to find for each point $q \in Q$ its nearest neighbor in L. If q is located in the region $V(e_l)$ then the nearest neighbor of q is e_l. Suppose $q_k \in V(e_j)$. Let q'_k be the image $I(q_k, e_j)$ of $q_k \in Q$ on some element $e_j \in L$. By Lemma 1 $\overline{q_k, q'_k}$ lies entirely in $V(e_j)$. We next find the midpoint m_k of $\overline{q_k, q'_k}$. It is obvious that the circle K centered at m_k with radius $d(m_k, q'_k)$ will not contain any other point of L in its interior. If m_k is found to be in the Voronoi region $V(q_k)$ of $V(Q)$, then the circle K will not contain any other point of Q in its interior either. Thus, m_k will be in the starter set ST (L, Q) for the merge curve M' and $B(e_j, q_k)$ is a starting bisector for a piece of M'. If m_k does not lie in $V(q_k)$, we discard it. Now we have to show that indeed each piece of M' passes through at least one point of ST (L, Q). To see this consider a piece of merge curve $M'_i = B(L_i, Q_i)$, where $L_i \subseteq L$ and $Q_i \subseteq Q$. There exists a point $q_j \in Q$ and a point t of some element $e_i \in L_i$ such that $d(t, q_j) = d(L_i, Q_i)$. As a result, the point $q_j \in Q_i$ must

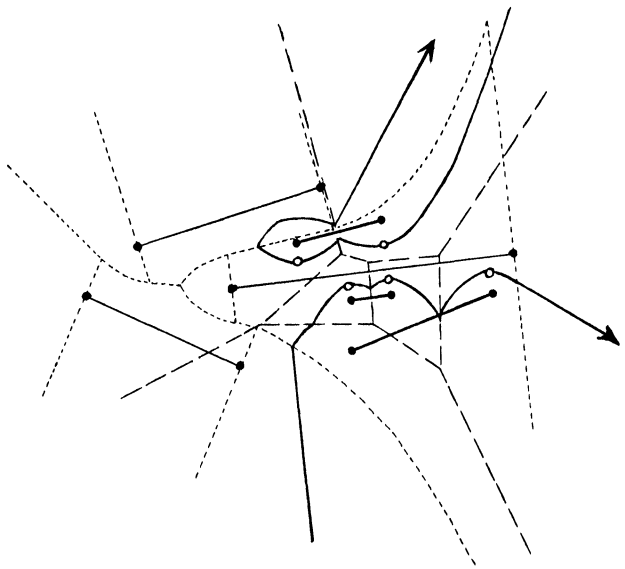

FIG. 9. *Merge of two Voronoi diagrams $V(L)$ and $V(Q)$.*

lie in $V(e_i)$ of $V(L)$ such that t is the image $I(q_j, e_i)$. Furthermore, t must lie in $V(q_j)$ of $V(Q)$. Consequently, the midpoint m of $\overline{t, q_j}$ must belong to both $V(e_i)$ and $V(q_j)$. Therefore, m is in ST (L, Q). This proves that every piece of M' indeed passes through at least a point of ST (L, Q). In other words, we shall locate the set Q of points in $V(L)$ and, after obtaining the set C of midpoints of $\overline{q_k, q'_k}$, where $q_k \in Q$ and q'_k is the image $I(q_k, e_j)$ of q_k on $e_j \in L$, we locate the set C of midpoints in $V(Q)$. If the midpoint m_k of $\overline{q_k, q'_k}$ is found in $V(q_k)$, then we store it in ST (L, Q). Otherwise, we ignore it. Fig. 9 shows the merge curve for the two Voronoi diagrams $V(L)$ and $V(Q)$, and "$\circ$" denotes the point in ST (L, Q). The time needed for the determination of ST (L, Q), is $O(N \log N)$. This result is due to Preparata [16]. In [16] Preparata has shown that a set of N points can be located in $O(N \log N)$ time in a planar subdivision with N vertices. After we have obtained ST (L, Q), we can use the procedure to be described in the next section to construct M' in $O(N \log N)$ time.

Suppose we have constructed the Voronoi diagram $V(L \cup Q)$. The Voronoi region $V'(q_k)$ associated with $q_k \in Q$ is the intersection of the half-planes $h(q_k, e_l)$, for all $e_l \in L \cup Q - \{q_k\}$. Since the Voronoi region $V(q_k)$ associated with $q_k \in Q$ in the Voronoi diagram $V(R)$ is the intersection of the half-planes $h(q_k, e_r)$, for all $e_r \in R - \{q_k\}$, the intersection of $V'(q_k)$ and $V(q_k)$ gives rise to the Voronoi region $V''(q_k)$ in the final diagram $V(S)$. Fig. 10 shows the Voronoi regions $V''(q)$, the intersection of $V(q)$ and

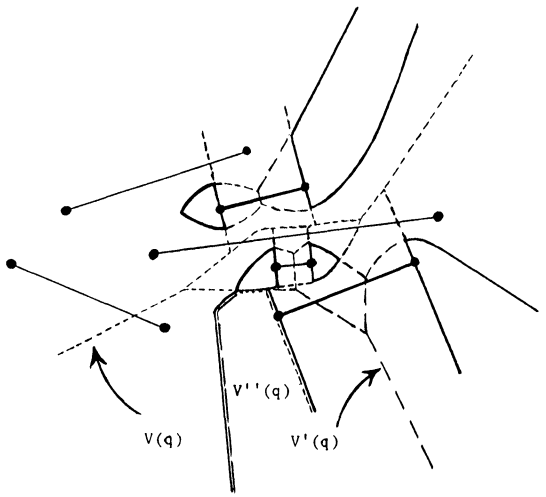

FIG. 10. *Intersection of $V(q)$ and $V'(q)$.*

$V'(q)$. Any point on the Voronoi edge $\bar{B}(q_k, e_l)$ of $V''(q_k)$, where $e_l \in L$, can be a point in the starter set ST (L, R) and $B(q_k, e_l)$ is a possible starting bisector for a piece of the merge curve $M = B(L, R)$. The intersection of $V(q_k)$ and $V'(q_k)$, both of which are generalized-star-shaped with nucleus being a point can be found in time $O(s + t)$, where s and t are the numbers of edges of $V(q_k)$ and $V'(q_k)$, respectively, by a technique similar to one given in [20] in finding the intersection of two convex polygons. Therefore, the starter set ST (L, R) can be obtained in $O(N \log N)$ time and will include at most one point for each q_k. Now we shall show that the merge curve $M = B(L, R)$ can be constructed in $O(N \log N)$ time.

3.3. Merging two Voronoi diagrams. After we have determined the starter set, we shall construct the merge curve M piece by piece. We shall take a point in ST (L, R) and use the corresponding bisector as our starting bisector to construct a piece of merge curve in two passes[4] [13]. If the piece of merge curve passes through a point of ST (L, R), that point will *not* be used again. Because each point is associated with a particular q_k, it is easy to check in constant time if the current bisector piece is passing through a starter point. The merge process works in a way very similar to that given before except that we make the following modifications.

In order to start with a bisector $B(u, v)$, where $u \in L$ and $v \in R$, we need to scan the edges of $V(u)$ and $V(v)$ respectively to find the first intersection points of $B(u, v)$ and an edge of $V(u)$ and of $B(u, v)$ and an edge of $V(v)$; a simple-minded approach would result in $O(N^2)$ time to construct $M = B(L, R)$ since (i) the number of edges of $V(u)$ may be $O(N)$, (ii) there would be $O(N)$ pieces of M starting with bisectors of the form $B(u, v)$, $v \in R$, and (iii) for each piece it takes $O(N)$ time to find the first intersection point. However, since the points of ST (L, R), whose corresponding bisectors involve an element $u \in L$, are on the boundary of $V(u)$ in the final Voronoi diagram, they can be "ordered" in the sense of Theorem 1. That is, if u is an endpoint, they can be ordered by polar angle with point u as the origin. If u is a segment, they can be ordered by their images on the segment. To avoid repeated examinations of the edges of $V(u)$ we shall keep track of the ordering of the points in ST (L, R). For example, suppose u is a segment (a, b) and $V((a, b))$ is considered. Let m_i and m_j be two midpoints in ST (L, R) such that the corresponding bisectors are $B((a, b), e_i)$ and $B((a, b), e_j)$, respectively, and m_j appears "after" m_i in the CCW direction along the boundary of $V((a, b))$. Let M_i and M_j denote the two pieces of the merge curve M using m_i and m_j as starting points, respectively. Assume that M_i has been constructed. When constructing M_j starting with $B((a, b), e_j)$ we shall scan the edges of $V((a, b))$ in the CCW direction (stipulated by the scanning scheme) starting with the edge where the piece of merge curve M_i *exits* from $V((a, b))$. In this manner we can eliminate backtracking. A similar situation holds if u is an endpoint.

After we have determined the starting edge to be scanned in constructing each piece of merge curve M_i, the CW-CCW scan procedure can be applied. However, instead of discarding edges when no intersection is found we replace them with a single "dummy" edge. The merge process is more complex, but not conceptually difficult. For details see [3], which includes a full description and an implementation of this merge procedure. The following theorem ensures that the modified CW-CCW scan works correctly and that no backtracking is necessary.

THEOREM 4′. *Let e_i and e_j be elements in L and R, respectively, and let $B(e_i, e_j)$ be constructed during the merge process. No point of $V(e_i)$ in $V(L)$ which lies to the right of the oriented bisector $B(e_i, e_j)$ will be included in the region $V(e_i)$ in the final diagram. Similarly, no point of $V(e_j)$ in $V(R)$ which lies to the left of the oriented bisector $B(e_i, e_j)$ will be included in the region $V(e_j)$. Furthermore, the scanning of $V(e_i)$ in a* CCW *direction and $V(e_j)$ in a* CW *direction will find the first intersection between $B(e_i, e_j)$ and either $V(e_i)$ or $V(e_j)$.*

Proof. Similar to the proof of Theorem 4. □

The time for constructing all pieces of merge curves is $O(N \log N)$ which is due to the ordering on the points in ST (L, R) to determine the starting edge for the initial scan for each piece of merge curve. Fig. 11 shows the merge curve M of the two Voronoi

[4] If the piece of merge curve is an island, the procedure provided in [13] will bring us back to the starting point. One pass is thus sufficient.

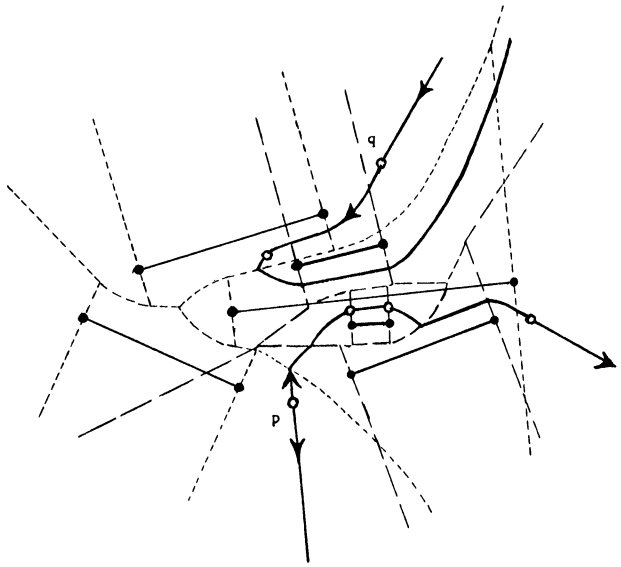

FIG. 11. *Merge of two Voronoi diagrams.*

diagrams $V(L)$ and $V(R)$ shown in short- and long-dashed lines respectively, starting with the points p and q; "$\circ$" denotes the points in the starter set ST (L, R).

Let us summarize the entire procedure for constructing the Voronoi diagram for a set S of N closed segments as follows.

Step 1. Order the set of N closed segments according to their left endpoints.

Step 2. Recursively construct the Voronoi diagrams $V(L)$ and $V(R)$, where L and R are the sets containing the left $\lfloor N/2 \rfloor$ and right $\lceil N/2 \rceil$ closed segments respectively.

Step 3. Construct the Voronoi diagram $V(Q)$ for the set Q of endpoints of R.

Step 4. Merge $V(Q)$ and $V(L)$ to form $V(L \cup Q)$.

Step 5. For each point $q \in Q$, find the intersection of $V(q)$ in the diagram $V(L \cup Q)$ and $V'(q)$ in the diagram $V(R)$. Let $V''(q)$ be the intersection of the two Voronoi polygons $V(q)$ and $V'(q)$.

Step 6. If the boundary of $V''(q)$ has an edge $B(q, e)$, where e is an element of L, then take any point on $B(q, e)$ and enter it into ST (L, R) along with the information that $B(q, e)$ is the corresponding bisector.

Step 7. Order these points in ST (L, R) in the sense of Theorem 1.

Step 8. Merge $V(L)$ and $V(R)$ by taking points of ST (L, R) in a specific order and applying the CW-CCW scanning scheme. Mark those points in the starter set which are passed through by the current piece of merge curve. Repeat this step until all the points in ST (L, R) are marked.

The time required for the initial sorting (Step 1) of endpoints is $O(N \log N)$. Constructing $V(Q)$ (Step 3) and $V(L \cup Q)$ (Step 4) takes $O(N \log N)$ time. Steps 6 and 8 take $O(N)$ time. Step 7 takes $O(N \log N)$ time. Since Steps 3 through 8 are executed $O(\log N)$ times due to recursion, the entire procedure takes $O(N(\log N)^2)$ time. Thus, we have the following results.

THEOREM 5. *Given a set S of N disjoint closed segments in the Euclidean plane, the Voronoi diagram $V(S)$ can be computed in $O(N(\log N)^2)$ time.*

COROLLARY 2. *Given a set of N disjoint closed segments in the Euclidean plane, the minimum weight spanning tree can be found in $O(N(\log N)^2)$ time.*

COROLLARY 3. *Given a set of N closed segments in the Euclidean plane, the all-nearest-neighbor problem can be solved in* $O(N(\log N)^2)$ *time.*

3.4. Extension to other figures and metrics. The technique used in constructing the Voronoi diagrams for a set of line segments in the Euclidean plane can be easily extended to the case when the given set of "objects" is a set of disjoint polygons or disjoint circles.

The procedure for constructing the Voronoi diagram for a set of line segments applies for the case of polygons, since the set of polygons can be treated as a set of nondisjoint line segments. However, for circles, the situation is different, since there is no "endpoint" that we can locate. But the following scheme works just as well. The idea is based on the fact that each piece of merge curve contains a point which is of the shortest distance from the corresponding sets of objects. The distance between two circles in the Euclidean metric is equal to the difference of the distance between the centers and the sum of their radii. Therefore as usual, we divide the set of circles into two subsets L and R of circles according to their leftmost points and recursively construct the Voronoi diagrams $V(L)$ and $V(R)$, respectively, for the two subsets L and R of circles. To merge them we need to determine the starting bisectors first. We shall locate the centers of the circles of the set R in the Voronoi diagram $V(L)$, and consequently find a set of images on the boundaries of the circles of L which are the nearest neighbors of the centers of the circles of R. Then we do another pass of point-location to locate the set of images in the Voronoi diagram $V(R)$. If the nearest neighbor of the image of c on some circle o_j in L, where c is a center of some circle o_k in R, is o_k, then $B(o_j, o_k)$ is a starting bisector. Let the image $I(c, o_j)$ of c on circle o_j in L be denoted by u, and the intersection of the straight line determined by u and c and the circle o_k in R be v. The midpoint of the line segment $\overline{u, v}$ is a point to be included in the starter set. The fact that the bisector $B(o_j, o_k)$ is a Voronoi edge can be shown in exactly the manner described earlier. Therefore, the Voronoi diagram for a set of N disjoint circles in the Euclidean plane can be found in $O(N(\log N)^2)$ time. We have the following results.

THEOREM 6. *Given a set of N disjoint objects (circles or polygons) in the Euclidean plane, the Voronoi diagram for the set of objects can be found in* $O(N(\log N)^2)$ *time.*

COROLLARY 4. *The minimum spanning tree for a set of N disjoint polygons or a set of disjoint circles can be constructed in* $O(N(\log N)^2)$ *time.*

The above results were stated in terms of the Euclidean metric. However, they can be proved using only the following assumptions about the metric in the plane.

(1) (Strict triangle inequality.) For all points x, y, and z, $d(x, y)+d(y, z)=d(x, z)$ if and only if x, y, and z are collinear in the standard Euclidean geometry in the plane.

(2) (Sides of triangles opposite nonacute angles are longest.) For any three points x, y, and z, if angle (x, y, z) has measure at least $\pi/2$ in the standard Euclidean geometry, then $d(x, z)>d(x, y)$ and $d(x, z)>d(y, z)$.

The L_p-metric, $1<p<\infty$, satisfies these properties. But L_1- and L_∞-metrics do not. Finding Voronoi diagrams for line segments or other objects in these metrics is still an open problem.

4. Conclusion. We presented an $O(N(\log N)^2)$ algorithm for computing the Voronoi diagram for a set of N circles or line segments in the plane under a general metric which satisfies (i) strict triangle inequality and (ii) the property that sides of triangles opposite nonacute angles are longest. The best lower bound for this problem is $\Omega(N \log N)$, so there is still room for improvement in either the algorithm or the bound.

5. Acknowledgment. Most of this work was done when Lee was with University of Illinois at Urbana-Champaign, supported in part by the National Science Foundation under grant MCS76-17321 and in part by the Joint Services Electronics Program under Contract DAAB-07-72-0259; and when Drysdale was at Stanford University, supported by the National Science Foundation under grant MCS77-05313.

REFERENCES

[1] M. A. BREUER, ed., *Design Automation of Digital Systems*, vol. 1, Prentice-Hall, Englewood Cliffs NJ, 1974.

[2] J. C. DAVIS AND M. J. MCCULLAGH, eds., *Display and Analysis of Spatial Data*, John Wiley, New York, 1975.

[3] R. L. DRYSDALE, III, *Generalized Voronoi diagrams and geometric searching*, Ph.D. Thesis, Department of Computer Science, Tech. Rep. STAN-CS-79-705, Stanford University, Stanford CA, 1979.

[4] R. L. DRYSDALE, III AND D. T. LEE, *Generalized Voronoi diagram in the plane*, Proceedings of the 16th Annual Allerton Conference on Communications, Control and Computing, Oct. 1978, pp. 833–842.

[5] R. L. FRANCIS AND J. A. WHITE, *Facility Layout and Location*, Prentice-Hall, Englewood Cliffs NJ, 1974.

[6] F. HARARY, *Graph Theory*, Addison-Wesley, Reading MA, 1972.

[7] J. A. HARTIGAN, *Clustering Algorithms*, Wiley Series in Probability and Statistics, John Wiley, New York, 1975.

[8] R. E. HORTON, *Rational study of rainfall data makes possible better estimates of water yield* $\cdots$, Engineering News-Record (1917), pp. 211–213.

[9] F. K. HWANG, *An $O(n \log n)$ algorithm for rectilinear minimal spanning trees*, J. Assoc. Comput. Mach., 26 (1979), pp. 177–182.

[10] D. E. KNUTH, *The Art of Computer Programming*, vol. 3, Addison-Wesley, Reading MA, 1973.

[11] R. J. KOPEC, *An alternative method for the construction of Thiessen polygons*, Professional Geographer, 15 (1963), pp. 24–26.

[12] D. T. LEE, *On finding k-nearest neighbors in the plane*, M.S. Thesis, Coordinated Science Lab., Rep. R-728, University of Illinois, Urbana IL, 1976.

[13] D. T. LEE AND C. K. WONG, *Voronoi diagrams in $L_1(L_\infty)$ metrics with 2-dimensional storage applications*, this Journal, 9 (1980), pp. 200–211.

[14] D. T. LEE, *Proximity and reachability in the plane*, Ph.D. Thesis, Department of Computer Science, Coordinated Sci. Lab. Rep. R-831, University of Illinois, Urbana IL, 1978.

[15] F. P. PREPARATA AND S. J. HONG, *Convex hulls of finite sets of points in two and three dimensions*, Comm. ACM, 20 (1977), pp. 87–93.

[16] F. P. PREPARATA, *A new approach to planar point location*, submitted for publication.

[17] D. RHYNSBURGER, *Analytic delineation of Thiessen polygons*, Geographic Analysis, 5 (1973), pp. 133–144.

[18] C. A. ROGER, *Packing and Covering*, Cambridge University Press, Cambridge, 1964.

[19] T. L. SAATY, *Optimization Problems in Integers and Related Extremal Problems*, McGraw-Hill, New York, 1970.

[20] M. I. SHAMOS, *Computational Geometry*, Department of Computer Science, Yale University, 1977, to be published by Springer-Verlag.

[21] M. I. SHAMOS AND D. HOEY, *Closest-point problems*, Proc. 16th IEEE Symposium on Foundations of Computer Science, Oct. 1975, pp. 151–162.

[22] A. H. THIESSEN, *Precipitation averages for large areas*, Monthly Weather Review, 39 (1911), pp. 1082–1084.