Far East Journal of Mathematical Sciences (FJMS)

http://www.pphmj.com
http://dx.doi.org/10.17654/MS103101647
Volume 103, Number 10, 2018, Pages 1647-1678 **ISSN: 0972-0871**

PLANE SWEEP ALGORITHM FOR CIRCULAR ARCS

Hyeon-Suk Na

School of Computer Science and Engineering
Soongsil University
Seoul, Korea

Abstract

Given a set of n circular arcs, possibly sharing endpoints, we present a plane sweep algorithm for computing the Voronoi diagram of this set in the worst-case optimal $O(n \log n)$ time. With a simple modification, this algorithm applies to a set of points, circles, circular and line segments, possibly intersecting each other. For a set of n points, circles, circular and line segments with k intersections, the modified algorithm computes the Voronoi diagram of this set in $O((n + k)\log(n + k))$ time.

1. Introduction

Plane sweeping is one of the most simple and elegant approaches for computing Voronoi diagrams. It does not require any deletion of old part of the diagram such as in the incremental algorithms or any intricate merging of several components such as in the divide-and-conquer algorithms. As for the Voronoi diagram of circular arcs, to our best knowledge, no plane sweeping algorithm has been studied so far; two algorithms in divide-and-conquer scheme [8, 1], two randomized incremental algorithms [3, 6] and one plane

Received: January 4, 2018; Accepted: February 8, 2018
2010 Mathematics Subject Classification: 68U05.
Keywords and phrases: Voronoi diagram, plane sweep algorithm, circular arcs.

sweep algorithm for circles [7] have been introduced so far, which we summarize shortly.

Yap [8] developed a divide-and-conquer algorithm to compute the Voronoi diagram of n line segments and circular arcs, possibly sharing endpoints, in the worst-case optimal $O(n \log n)$ time. After decades, Aichholzer et al. [1, 2] presented a new divide-and-conquer approach, more practical and easier to implement, for computing the Voronoi diagram of pairwise disjoint objects. The objects must be topological disks of dimension two, one, or zero in $\mathbb{R}^2$, i.e., homeomorphic to a disk, a line segment, or a point, respectively. So, the input may include (filled) polygons, disks and open spline curves, but exclude a (unfilled) polygon or a bundle of more than two arcs sharing a point. The key idea of this algorithm is to interpret the Voronoi diagram as the medial axis of a planar region, subregions augmented along the boundary of some largest empty disks. The medial axis is computed in a divide-and-conquer scheme: splitting recursively into directly solvable base cases and merging. By empirical evidence, the authors showed that the expected time-complexity of this algorithm becomes $O(n \log n)$ for the input of few complex objects bounded by n circular arcs, but gets worse to $O(n \log^2 n)$ for the input of many simple objects such as n points.

In a different paradigm, Alt et al. [3] studied a randomized incremental algorithm that computes the Voronoi diagram of n circular arcs and spiral arcs, possibly sharing endpoints, in expected $O(n \log n)$ time. The main idea of this algorithm is to split each arc into the endpoints and an open arc and to compute three regions separately because then the region of the open arc is guaranteed to be simply-connected if it is inserted into the set after the endpoints. Based on the same idea and a topology-oriented insertion scheme, Held and Huber [6] extended their Voronoi code “VRONI” to circular arcs that runs in expected $O(n \log n)$ time.

The first plane sweep algorithm of the Voronoi diagram was introduced by Fortune [5] for a set of pairwise disjoint points and line segments.

Voronoi diagram and the beach line are maintained and updated at the so-called *site events* and *circle events*, in which any topological and combinatorial changes in these structures happen, and the algorithm runs in the worst-case optimal $O(n \log n)$ time for n points and segments. A brief sketch for the Voronoi diagram of disjoint line segments is given in [4, Chapter 7]. Jin et al. [7] presented a plane sweep algorithm that computes the Voronoi diagram of possibly intersecting n circles in the worst-case $O(k \log k)$ time, where k denotes the complexity of the output. To handle circles possibly intersecting each other, they introduced two more types of events, called *merge event*, *cross event* and the operations therein.

As noted in [1, 6, 7], none of these plane sweep algorithms are directly extendible to circular arcs. In this note, we present a plane sweep algorithm that computes the Voronoi diagram of n circular arcs, possibly sharing endpoints, in the worst-case optimal $O(n \log n)$ time. With a simple modification, our algorithm applies to a set of points, circles, circular and line segments, possibly intersecting each other. For a set of n points, circles, circular and line segments with k intersections, the modified algorithm computes the Voronoi diagram of this set in $O((n + k) \log(n + k))$ time.

2. Preliminaries

Let $\mathcal{S}$ be a set of circular arcs, possibly sharing endpoints. Since circular arcs are allowed to share endpoints, we assume that any arc is less than a semi-circle. As in [8, 3, 6], we split this set of circular arcs into three kinds of *sites*: *endpoints*, *junctions* and (*open*) *arcs*, where "endpoints" indicate the degree-one endpoints of arcs, "junctions" the points shared by two or more circular arcs, i.e., the points of degree ≥ 2, and "open arcs" the arcs left over after removing endpoints and junctions. In the sequel, we regard $\mathcal{S}$ as a set of sites, each being an endpoint, an open arc, or a junction at which two or more circular arcs meet. For an open arc site $s \in \mathcal{S}$, we call the circle passing through s the *generator circle* of s, and if s is incident to an endpoint or junction site $s' \in \mathcal{S}$, then we call it the *generator arc* of s'.

Voronoi diagram. We define the Voronoi diagram similarly to [8, 3, 6]. For two points $p, q \in \mathbb{R}^2$, let $d(p, q)$ denote the Euclidean distance between p and q. For a set P of points in $\mathbb{R}^2$, the distance $d(p, P)$ between p and P is defined as $\inf\{d(p, q) : q \in P\}$. For a site $s \in \mathcal{S}$, the *cone of influence* $\mathcal{S}\ell(s)$ is the interior of the set defined as:

- the whole plane, if s is a point,
- the cone of infinite rays emanating from the center and passing through the points of s, if s is an open circular arc.

The *Voronoi region* of a site $s \in \mathcal{S}$ is defined as

$$\mathcal{VR}(s, \mathcal{S}) := \{q \in \mathcal{C}\ell(s) : d(q, s) \le d(q, \mathcal{S})\},$$

and the *Voronoi diagram* $\mathcal{V}(\mathcal{S})$ of $\mathcal{S}$ is the union of the boundary of Voronoi region $\mathcal{VR}(s, \mathcal{S})$ of all sites $s \in \mathcal{S}$. A *Voronoi edge* is a maximal connected portion of $\mathcal{V}(\mathcal{S})$ that belongs to the boundary of exactly two Voronoi regions, and a *Voronoi vertex* is a point that belongs to the boundary of three or more Voronoi regions. For a point $p \in \mathbb{R}^2$, the *largest empty disk* $D(p, \mathcal{S})$ is defined as the closed disk centered at p with radius $d(p, \mathcal{S})$.

Figure 1 illustrates the Voronoi diagram $\mathcal{V}(\mathcal{S})$ of $\mathcal{S} = \{A, B, C, s_1, s_2\}$, where the open arc sites s_1, s_2 are tangent at the junction site B and have two endpoint sites A, C in the boundary. Note that the four sites except for B have two-dimensional Voronoi regions, but the junction site B has one-dimensional Voronoi region, the ray $\overrightarrow{c_1B}$, thus the boundary of $VR(B, \mathcal{S})$ is the point c_1 only. The rays $\overrightarrow{c_1A}$, $\overrightarrow{c_1B}$, $\overrightarrow{vC}$ are the Voronoi edges corresponding to the pairs of sites (A, s_1), (s_1, s_2), (s_2, C), respectively, and the points c_1 and v are the Voronoi vertices corresponding to the tuples of sites (A, B, s_1, s_2) and (A, C, s_2), respectively.

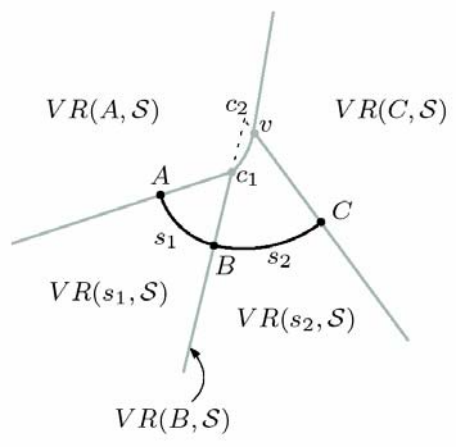


Figure 1. The Voronoi diagram of $\mathcal{S} = \{A, B, C, s_1, s_2\}$. The open arcs s_1 and s_2 are circular arcs centered at c_1 and c_2, respectively, and tangent at the junction site *B*. Voronoi edges and vertices are indicated by gray curves and circles, respectively.

The following topological and combinatorial properties of $\mathcal{V}(\mathcal{S})$ were proven in [8, 3]:

Property 1. Let $\mathcal{S}$ be a set of *n* sites *s*. Then every Voronoi region $\mathcal{VR}(s, \mathcal{S})$ is simply-connected, and the Voronoi diagram $\mathcal{V}(\mathcal{S})$ is a planar connected graph with at most $3n - 6$ edges.

Bisectors. Given two sites $s_1, s_2 \in \mathcal{S}$, denote by $B(s_1, s_2)$ the bisector of s_1 and s_2, the set of points equidistant from both sites. $B(s_1, s_2)$ is a line, an ellipse, or a hyperbola. It is a line if s_1 is a point site and s_2 is either a point site or the generator arc of s_1. If s_1 and s_2 are open arcs, let their generator circles g_1 and g_2 be centered at c_1 and c_2 with radii r_1 and r_2, respectively, and $r_1 \geq r_2$. Then any point $p \in B(s_1, s_2)$ satisfies one of the following equations:

$$d(p, c_1) - d(p, c_2) = r_1 - r_2, \tag{1}$$

$$d(p, c_1) + d(p, c_2) = r_1 - r_2. \tag{2}$$

If the point p lies both inside or both outside of g_1 and g_2, then p lies on a hyperbola satisfying (1). Otherwise, i.e., the point p lies inside of one generator circle and outside of the other, then p lies on an ellipse satisfying (2). The case where s_2 is a point site and s_1 is an open arc, not being the generator arc of s_2, can be regarded as being $r_2 = 0$ in the above equations.

Plane sweeping. The strategy of a plane sweep algorithm for Voronoi diagram $\mathcal{V}(\mathcal{S})$ is to sweep the plane from top to bottom with a horizontal line, called the *sweep line* ℓ. Denote by H^+ the closed half-plane above ℓ and by $\mathcal{S}^+$ the set of points lying in $\mathcal{S} \cap H^+$. The locus of points closer to any point of $p \in \mathcal{S}^+$ than to ℓ is bounded by a chain of parabolic arcs, called the *beach line*. The *breakpoints* between two adjacent *beach line edges* trace out Voronoi edges. While the sweep line moves down, combinatorial information of the Voronoi diagram and the sequence of breakpoints along the beach line are maintained. The moments when these combinatorial changes arise are called *events*.

For a set of points, two types of events are introduced: *site event* when the sweep line ℓ meets a new site and *circle event* when a beach line edge shrinks to a point and disappears [5]. But for a set of circles possibly intersecting or crossing each other [7], two more types of events are necessary: *merge event* when the sweep line ℓ leaves a circle and *cross event* when the sweep line ℓ passes through an intersecting point of circles.

In this paper, to deal with circular arcs, we use the circle, merge and cross events in the same sense, but we subdivide the site event into top, start and end events. Clearly, the operations to be done in each event are different from those of previous works [5, 7].

Beach line edges. Given a sweep line ℓ and an arc site $s \in \mathcal{S}$ with generator circle centered at c with radius r, the beach line edge corresponding to s is defined as the set of points $p \in H^+$ satisfying that

$$d(p, \mathcal{S}^+) = d(p, s \cap H^+) \quad \text{and} \quad |\, d(p, c) - r \,| = d(p, \ell). \tag{3}$$

So, if p is outside of the generator circle, then it lies on the parabola opening upward whose focus is c and whose directrix is the translation of ℓ by r downward. Otherwise, i.e., p is inside of the generator circle, then it lies on the parabola opening downward whose focus is c and whose directrix is the translation of ℓ by r upward. The case where s is a point site can be regarded as being $r = 0$. Thus, the beach line of $\mathcal{S}$ is a x-monotone sequence of parabolic segments whose breakpoints trace out either an arc site or a Voronoi edge being a line segment or an elliptic or hyperbolic segment. When a beach line edge shrinks to a point, the Voronoi edges traced out by the endpoints of this edge meet at a point, a Voronoi vertex. In the next section, we will prove Theorem 1, showing that this happens only at cross, merge and circle events.

3. Events for the Voronoi Diagram of Circular Arcs

During the plane sweep, the algorithm needs to maintain and update the combinatorial information of the Voronoi diagram and the sequence of breakpoints along the beach line. So, the most important element of the algorithm is to list up all events, i.e., the locations (or the y-coordinates) of the sweep line ℓ in which any combinatorial information changes in these structures and to process all updates correctly. In Subsection 3.1, we briefly sketch when the events occur, and in the following subsections, we describe all details of events and necessary updates in the data structures.

In the sequel, we assume that no arc site is cut from its generator circle at the topmost or bottommost point. This implies that no endpoint or junction site is the topmost or bottommost point of its generator arc site and that no bisector between an endpoint or junction site and its generator arc site is vertical. We say that a disk D *touches* a site $s \in \mathcal{S}$ if the interior of D contains no points of s and the boundary of D shares a point with the closure of s. Thus, a disk D might touch an endpoint site s and its generator arc site s' at the same time, and D and s' do not necessarily have the same tangent direction. We denote by ℓ^u and ℓ^b a line infinitesimally above and below the sweep line ℓ, respectively.

3.1. Types of events

We classify all events into the following six types of events:

- **Top event:** This event occurs when the sweep line ℓ meets the interior point of $s \in \mathcal{S}$ being the topmost of the generator circle of s. For instance, in Figure 2, the topmost point of arc site s_2 calls a top event.

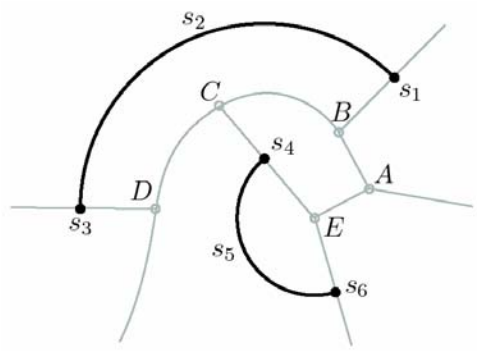


Figure 2. Illustration of five types of events: The topmost point of s_2 calls a top event and the bottommost point of s_5 calls a *merge event*. The endpoint sites s_4, s_6 call *start events*, and s_1, s_3 call *end events*. During the plane sweep, four circle events have taken place and have generated Voronoi vertices from A to D. The Voronoi vertex E is generated at the merge event. Voronoi edges and vertices are indicated by gray curves and circles, respectively.

- **Start event:** This event occurs when the sweep line ℓ meets an endpoint site $s \in \mathcal{S}$ whose generator arc $s' \in \mathcal{S}$ lies *locally below* the sweep line ℓ. In Figure 2, the endpoint sites s_4 and s_6 call start events.
- **End event:** This event occurs when the sweep line ℓ meets an endpoint site $s \in \mathcal{S}$ whose generator arc $s' \in \mathcal{S}$ lies *locally above* the sweep line ℓ. In Figure 2, the endpoint sites s_1 and s_3 call end events.

- **Merge event:** This event occurs when the sweep line ℓ meets the interior point of $s \in \mathcal{S}$ being the bottommost of the generator circle of *s*. In Figure 2, the bottommost point of arc site s_5 calls a merge event and the Voronoi vertex *E* is generated at this event.
- **Circle event:** This event occurs when a beach line edge disappears from the beach line. As a result, the Voronoi edges traced out by the endpoints (i.e., breakpoints) of this edge meet at a Voronoi vertex, and the sweep line ℓ passes through the bottommost point of the largest empty disk that touches the three sites corresponding to this edge and its neighbors. In the example of Figure 2, four circle events have taken place and have generated Voronoi vertices from *A* to *D*. Theorem 1 will clarify the characteristics of this event more precisely.
- **Cross event:** This event occurs when the sweep line ℓ meets a junction site $s \in \mathcal{S}$ having two or more generator arcs incident to *s*. We treat this event by dividing the configuration of the vicinity of *s* into two cases: when the union of cone of clearance of generator arcs of *s* covers an open neighborhood of *s* and when it does not cover any open neighborhood of *s*.

3.2. Initialization and data structures

We adopt the usual data structures and techniques of plane sweep algorithms [4]. We employ the balanced binary search tree $\mathcal{T}$ for the beach line, the priority queue $\mathcal{Q}$ for events, and the doubly connected edge list (DCEL) for the Voronoi diagram. Each leaf of $\mathcal{T}$ represents a beach line edge and stores information of its corresponding site and a pointer to the circle event in $\mathcal{Q}$ in which this beach line edge disappears. Each internal node of $\mathcal{T}$ represents a breakpoint as an ordered tuple of sites and stores a pointer to a half-edge in the doubly connected edge list of the Voronoi diagram. Each node of the priority queue *Q* stores information of the event it represents and for a circle event additionally a pointer to the leaf in $\mathcal{T}$ that disappears in this event.

As an initialization process, we first insert all start/end and cross events into $\mathcal{Q}$, corresponding to the endpoint sites and junction sites of $\mathcal{S}$, respectively. We then insert all top or merge events by checking for each arc site s if the topmost or bottommost point of its generator circle is contained in the interior of s. Circle events are enqueued during the run of the algorithm, and some circle events are removed before their turns as soon as they become infeasible. Deletion of these false-alarms can be done using the pointers from the leaves of $\mathcal{T}$ to the corresponding circle events in $\mathcal{Q}$.

3.3. Circle event

This event indicates the moment that a beach line edge disappears from the beach line. Then the two old Voronoi edges traced out by the endpoints (i.e., breakpoints) of this edge meet at a Voronoi vertex, and the sweep line ℓ passes through the bottommost point of the largest empty disk that touches the three sites corresponding to this edge and its neighbors. The following theorem shows that every Voronoi vertex v is either itself a junction site (calling a *cross event*), or is generated either at a merge event or at a circle event defined by a triple of consecutive beach line edges, with the middle one shrinking. Cross and merge events are computed and enqueued into $\mathcal{Q}$ in the beginning of the algorithm, so only circle events defined by a triple of consecutive beach line edges, with the middle one shrinking, need to be computed during the algorithm.

Theorem 1. *Let v be a Voronoi vertex, not being a junction site, defined by three sites $s_1, s_2, s_3 \in \mathcal{S}$. Denote simply by D the largest empty disk $D(v, \mathcal{S})$ with positive radius touching s_1, s_2, s_3, and let the sweep line ℓ pass through the bottommost point of D. Then one of the following is true*:

(i) *One of s_1, s_2, s_3 is the generator arc of the other two and the arc site contains the bottommost point of their generator circle D; thus ℓ corresponds to the merge event called by this arc site.*

(ii) *In the beach line of ℓ^u, a line infinitesimally above ℓ and below the sites s_1, s_2, s_3, there exist three consecutive beach line edges $\alpha_1, \alpha_2, \alpha_3$ corresponding to s_1, s_2, s_3, respectively, with α_2 shrinking.*

Proof. Consider first the case where one of s_1, s_2, s_3 is the generator arc of the other two. Without loss of generality, we may assume that s_2 is the arc site having s_1 and s_3 as the endpoints. If s_2 contains the bottommost point of D (in the interior), then the largest empty disk D must be the generating circle of s_2 passing through all three sites and ℓ corresponds to the merge event called by s_2. Refer to the Voronoi vertex E defined by the sites s_4, s_5, s_6 in Figure 2.

If s_2 does not contain the bottommost point of D, then consider a line ℓ^u infinitesimally above ℓ and below all the sites s_1, s_2, s_3. We can obtain a pencil of disks D_2 whose bottommost points lie on ℓ^u, touching s_2 at every interior point of s_2 and containing no points of $\mathcal{S}$ in the interior. Then the set of centers of such D_2s is the beach line edge defined by ℓ^u and s_2, denoted by α_2. Starting from the ends of the pencil of D_2s, we can get two pencils of disks D_1 and D_3 whose bottommost points lie on ℓ^u, touching s_1 and s_3, respectively, and containing no points of $\mathcal{S}$ in the interior. The centers of such D_1s and D_3s are the beach line edges α_1 and α_3 defined by ℓ^u and s_1 and by ℓ^u and s_3, respectively. Refer to the Voronoi vertex B with the beach line edges α_1, α_2, α_3 in $\mathcal{T}_1$ and the Voronoi vertex D with the beach line edges α_4, α_5, α_9 in $\mathcal{T}_3$ in Figure 3.

Consider next the case where one of s_1, s_2, s_3 is a point site and one of the other two is the generator arc of the point site; Refer to the Voronoi vertices B, C, D in Figure 2 or A in Figure 3. Like the Voronoi vertex A of Figure 3, we may assume that s_3 is the generator arc of endpoint site s_2. Then the Voronoi vertex v lies on the line normal to s_3 at s_2 and D touches s_2 and s_3 at s_2. By shrinking D while touching s_3 at s_2, we can obtain a disk having the bottommost point on ℓ^u and containing no points from $\mathcal{S}$ in the interior. Starting from this disk, we can get two pencils of empty disks

D_2 and D_3 whose bottommost points lie on ℓ^u, touching s_2 and s_3 at every interior points, respectively. The centers of such D_2s and D_3s are the beach line edges α_2 and α_3 defined by ℓ^u and s_2 and by ℓ^u and s_3, respectively. Among the empty disks D_2, there must be some disk touching both s_1 and s_2. Starting from this disk we can get a pencil of disks D_1 whose bottommost points lie on ℓ^u, touching s_1 and containing no points from $\mathcal{S}$ in the interior, and the set of centers of such D_1s is the beach line edge α_1 defined by ℓ^u and s_1. Refer to the Voronoi vertex A with the beach line edges α_6, α_7, α_8 in $\mathcal{T}_1$ in Figure 3.

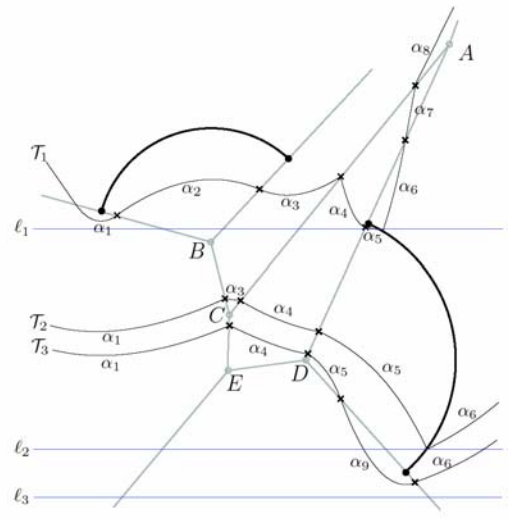


Figure 3. Illustration of circle events described in Theorem 1(ii). Voronoi edges and vertices are indicated by gray curves and circles, respectively, and beach line edges and breakpoints are by black solid curves and crosses.

The last remaining case is where none of s_1, s_2, s_3 are the generator arcs of another. Refer to the Voronoi vertex A in Figure 2 or C, E in Figure 3. Then the points in which D touches s_1, s_2, s_3, denoted by x_1, x_2, x_3 are all distinct. Let $i = 1, 2, 3$ be fixed. By shrinking D while touching s_i in x_i,

we can obtain the disk D_i that touches s_i in x_i and has the bottommost point on ℓ^u. The disk D_i contains no points from $\mathcal{S}$ in the interior, so its center lies on the beach line edge defined by ℓ^u and s_i, namely α_i. Refer to the Voronoi vertex *C* with the beach line edges α_1, α_3, α_4 in $\mathcal{T}_2$.

In all cases above, as the sweep line moves downward from ℓ^u to ℓ, the centers of empty disks D_1 and D_3 become closer approaching the center *v* of *D*, and the beach line edge in the middle, α_2, shrinks to the Voronoi vertex *v*. □

Now we verify how to detect all circle events described in Theorem 1(ii) and what to do in a circle event for the data structures. The strategy to find all circle events is the same as all previous works: whenever a new triple of consecutive edges appears in the beach line, to check if there exists a disk touching the corresponding sites with the bottommost point lying below ℓ, and if feasible, to push this circle event into $\mathcal{Q}$ with adding pointers between the event and the leaf of $\mathcal{T}$ representing the middle edge to be disappeared. To avoid false-alarms, when the triple becomes non-consecutive in the beach line, we remove the event from $\mathcal{Q}$ using the pointer from the leaf of $\mathcal{T}$.

Necessary updates of $\mathcal{T}$ and $\mathcal{Q}$ are the following: assuming that β_L and β_R denote the left and the right neighbors of α_1 and α_3, respectively, we first remove the edge α_2 from $\mathcal{T}$ and the circle events involving α_2 from $\mathcal{Q}$ (namely, the events defined by triples $(\beta_L, \alpha_1, \alpha_2)$ and $(\alpha_2, \alpha_3, \beta_R)$ which can be found using the pointers from α_1 and α_3) and then insert two events into $\mathcal{Q}$ defined by triples $(\beta_L, \alpha_1, \alpha_3)$ and $(\alpha_1, \alpha_3, \beta_R)$, if feasible. For the Voronoi diagram, we add a new Voronoi vertex *v* as the endpoints of two old Voronoi edges traced by the breakpoints $\alpha_1 \cap \alpha_2$ and $\alpha_2 \cap \alpha_3$, and then add a new Voronoi edge stemming from *v*, defined by the sites corresponding to α_1 and α_3 and traced out by the breakpoint $\alpha_1 \cap \alpha_3$.

HandleCircleEvent(α)

1. Search the predecessor and the successor of α, denoted by α_1 and α_3, and delete the leaf in $\mathcal{T}$ corresponding to α.
2. Delete two circle events involving α from $\mathcal{Q}$, by following the pointers from α_1 and α_3. Check if two new triples of consecutive beach line edges with α_1 and α_3 in the middle, respectively, generate feasible circle events. If so, insert them into $\mathcal{Q}$.
3. Generate a new Voronoi edge defined by the sites corresponding to α_1 and α_3, and add a new Voronoi vertex as the endpoints of three Voronoi edges: two old edges traced by breakpoints $\alpha_1 \cap \alpha$ and $\alpha \cap \alpha_3$ and the new edge just created.

3.4. Top event

This event occurs when the sweep line ℓ meets an interior point of $s \in \mathcal{S}$ being the topmost point T of the generator circle g of s. Figure 4 illustrates the Voronoi diagram and the beach line before and after a top event.

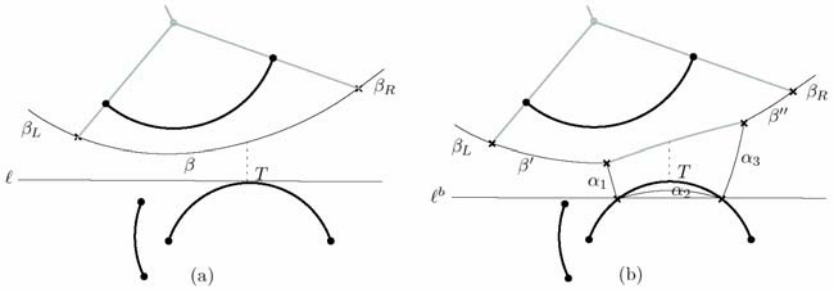


Figure 4. Voronoi diagram and the beach line (a) before and (b) after a top event.

In the beach line $\mathcal{T}$, we first locate the beach line edge β vertically above T and its left and right neighbors β_L and β_R. We then split β into β', β'' and insert three new edges α_1, α_2, α_3 between β' and β'', where α_1

and α_3 are the left and the right segments of points lying *outside* of *g* and equidistant from ℓ^b and *s*, and α_2 is the set of points lying *inside* of *g* and equidistant from ℓ^b and *s*. As seen in Section 2, both α_1 and α_3 lie on a parabola opening upward whose focus is the center of *g* and whose directrix is a translation of ℓ^b downward, and α_2 lies on a parabola opening downward with the same focus and with directrix being a translation of ℓ^b upward.

For the event queue $\mathcal{Q}$, we remove the circle event $(\beta_L, \beta, \beta_R)$ and insert two circle events defined by the triples $(\beta_L, \beta', \alpha_1)$ and $(\alpha_3, \beta'', \beta_R)$, if feasible. In $\mathcal{V}(\mathcal{S})$, we add a new Voronoi edge defined by *s* and the site corresponding to β, stemming from the point of β vertically above *T* and traced out leftward and rightward by the breakpoints $\beta' \cap \alpha_1$ and $\alpha_3 \cap \beta''$.

HandleTopEvent(Topmost point *T* of arc site *s*)

1. In $\mathcal{T}$, locate the beach line edge β vertically above *T*. Split β into β', β'' and insert new edges α_1, α_2, α_3 between β' and β'', where α_1 and α_3 are the left and the right segments of points lying *outside* of *g* and equidistant from ℓ^b and *s*, and α_2 is the set of points *lying* inside of *g* and equidistant from ℓ^b and *s*.
2. Delete the circle event corresponding to β from $\mathcal{Q}$, and if new triples of consecutive beach line edges with β' and β'' in the middle, respectively, generate feasible circle events, insert them into $\mathcal{Q}$.
3. Generate a new Voronoi edge defined by *s* and the site corresponding to β, stemming from the point of β vertically above *T* and traced out leftward and rightward by the breakpoints $\beta' \cap \alpha_1$ and $\alpha_3 \cap \beta''$.

3.5. Start event

This event occurs when the sweep line ℓ meets an endpoint site $s \in \mathcal{S}$ whose generator arc $s' \in \mathcal{S}$ lies locally below ℓ. Denote by *g* the generating

circle of s'. Figure 5 illustrates the Voronoi diagram and the beach line before and after a start event.

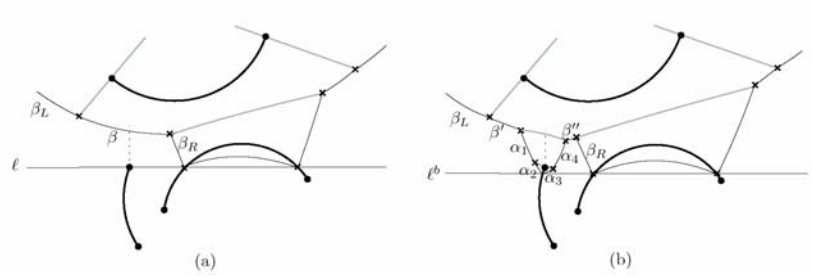


Figure 5. Voronoi diagram and the beach line, (a) before and (b) after a start event.

The operations on the beach line are similar to what we did for a top event. We locate the beach line edge β vertically above *s* having the left and right neighbors β_L and β_R. We then split β into β′, β″ and insert four new edges α_1, ..., α_4 between β′ and β″, where α_1 and α_4 are the portions of parabola defined by ℓ^b and *s*, and α_2 and α_3 are the portions of parabola defined by ℓ^b and s', one lying outside of *g* and the other lying inside of *g*.

For $\mathcal{Q}$, we remove the circle event corresponding to β and add any feasible circle events, defined by triples of consecutive beach line edges with any of β′, α_1, α_4, β″ in the middle. For $\mathcal{V}(\mathcal{S})$, we add a new Voronoi edge defined by *s* and the site corresponding to β, stemming from the point of β vertically above *s* and traced out leftward and rightward by the breakpoints $\beta' \cap \alpha_1$ and $\alpha_4 \cap \beta''$. A new Voronoi edge defined by *s* and s', i.e., the line normal to s' at *s*, stems from *s* leftward and rightward by the breakpoints $\alpha_1 \cap \alpha_2$ and $\alpha_3 \cap \alpha_4$.

HandleStartEvent(Endpoint site *s* of a generator arc s')

1. In $\mathcal{T}$, locate the beach line edge β vertically above *s*. Split β into β′, β″ and insert new edges α_1, ..., α_4 between β′ and β″, where α_1 and α_4 are the portions of parabola defined by ℓ^b and *s*, and α_2 and

α_3 are the portions of parabola defined by ℓ^b and s', one lying outside of g and the other lying inside of g.

2. Delete the circle event corresponding to β from $\mathcal{Q}$, and add into $\mathcal{Q}$ any feasible circle events defined by triples of consecutive beach line edges with one of β', α_1, α_4 and β'' in the middle.

3. Generate two new Voronoi edges: one defined by s and the site of β, stemming from the point of β vertically above s and traced out leftward and rightward by the breakpoints $\beta' \bigcap \alpha_1$ and $\alpha_4 \bigcap \beta''$, and the other defined by s and s', i.e., the line normal to s' at s, stemming from s leftward and rightward by the breakpoints $\alpha_1 \bigcap \alpha_2$ and $\alpha_3 \bigcap \alpha_4$.

3.6. End event

This event occurs when the sweep line ℓ meets an endpoint site $s \in \mathcal{S}$ whose generator arc $s' \in \mathcal{S}$ lies locally above the sweep line ℓ. Figure 6 illustrates the Voronoi diagram and the beach line before and after an end event.

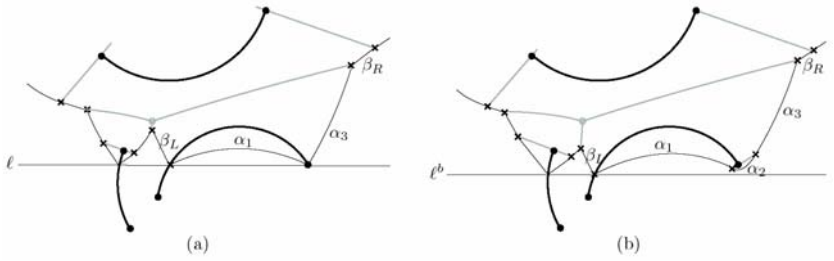


Figure 6. Voronoi diagram and the beach line, (a) before and (b) after an end event.

Just before this event, there must be two consecutive edges α_1 and α_3 in the beach line, defined by ℓ^u and s'. The operation for $\mathcal{T}$ is to insert a new edge α_2 between α_1 and α_3 defined by ℓ^b and s, and for the Voronoi diagram to add a new Voronoi edge defined by s and s', stemming from s

leftward and rightward by $\alpha_1 \cap \alpha_2$ and $\alpha_2 \cap \alpha_3$. For the event queue $\mathcal{Q}$, we add two circle events, if feasible, defined by the triples $(\beta_L, \alpha_1, \alpha_2)$ and $(\alpha_2, \alpha_3, \beta_R)$, where β_L, β_R denote the left and the right neighbors of α_1 and α_3, respectively. Note that as seen from $(\alpha_4, \alpha_5, \alpha_9)$ in Figure 3, if either β_L or β_R is the other endpoint of s', then the corresponding circle event is the bottommost point of the generator circle of s'.

HandleEndEvent(Endpoint site s of a generator arc s')

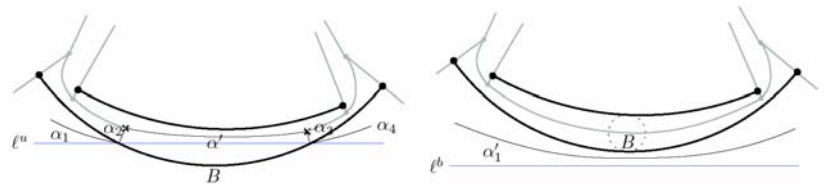
1. In $\mathcal{T}$, find two consecutive edges α_1, α_3 defined by ℓ^u and s', and insert a new edge α_2 defined by ℓ^b and s between them.
2. Insert into $\mathcal{Q}$ any feasible circle events defined by triples of consecutive beach line edges with α_1 or α_3 in the middle.
3. Generate a new Voronoi edge defined by s and s', stemming from s leftward and rightward by $\alpha_1 \cap \alpha_2$ and $\alpha_2 \cap \alpha_3$.

3.7. Merge event

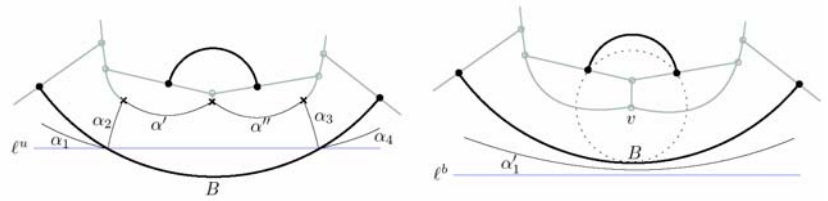
This event occurs when the sweep line ℓ meets an interior point of $s \in \mathcal{S}$ being the bottommost point B of the generator circle g of s. Just before this event, in the beach line, there must be four edges α_1, α_2, α_3, α_4 defined by ℓ^u and s, where α_1 and α_4 are the left and the right segments lying *outside* of g, and α_2 and α_3 are the left and the right segments lying *inside* of g. Also, between α_2 and α_3, there exists a sequence of consecutive edges defined by ℓ^u and other sites. Figure 7 illustrates three different configurations before and after a merge event.

The operation for $\mathcal{T}$ is to remove all edges from α_2 to α_3 and to merge α_1 and α_4 into one edge α_1' defined by ℓ^u and s. For $\mathcal{Q}$, remove all circle events defined by the edges from α_1 to α_4, and add a new circle event if feasible defined by a triple $(\beta_L, \alpha_1', \beta_R)$, where β_L and β_R are the left and

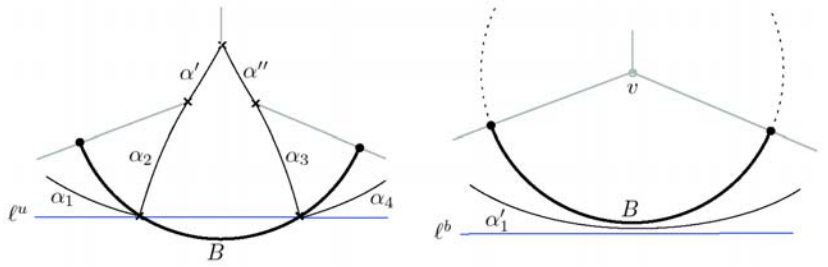
the right neighbors of α_1 and α_4, respectively. For the Voronoi diagram, in such cases as Figure 7(a), we just merge two old Voronoi edges traced by $\alpha_2 \cap \alpha'$ and $\alpha' \cap \alpha_3$ into one edge. In such cases as Figure 7(b) and Figure 7(c), we add a new Voronoi vertex *v* as the endpoints of all old Voronoi edges traced out by breakpoints between the edges from α_2 to α_3; these cases can be regarded as a coincidence of multiple circle events and a merge event.



(a) When there exists only one beach line edge α' between α_2 and α_3



(b) When there exist at least two beach line edges between α_2 and α_3



(c) When the endpoint sites of *s* are the sites corresponding to the edges between α_2 and α_3

Figure 7. Three different configurations of a merge event.

HandleMergeEvent(Bottommost point B of arc site s)

1. In $\mathcal{T}$, find four edges $\alpha_1 \cdots \alpha_4$ defined by ℓ^u and s. Delete all edges from α_1 to α_4 and instead add a new edge α_1' defined by ℓ^b and s.
2. Delete all circle events involving the deleted beach line edges from $\mathcal{Q}$, and if the new triple of consecutive beach line edges with α_1' in the middle generates a feasible circle event, insert it into $\mathcal{Q}$.
3. In case where only one edge α' between α_2 and α_3 existed, merge two old Voronoi edges traced by $\alpha_2 \cap \alpha'$ and $\alpha' \cap \alpha_3$ into one edge. In other cases, add the center of s as a new Voronoi vertex, and set it to be the endpoints of all old Voronoi edges traced out by the breakpoints between edges from α_2 to α_3.

3.8. Cross event

This event occurs when the sweep line ℓ meets a junction site $s \in \mathcal{S}$ having two or more generator arcs incident to s. Assume that m generator arcs are incident to s $(m \geq 2)$ and that within a sufficiently small neighborhood D of s, there exists no point of the other sites of $\mathcal{S}$ except s and its generator arcs. Let k and $m - k$ be the numbers of the generator arcs lying in the upper and the lower half-plane locally within D, respectively, where k is any number between 0 and m.

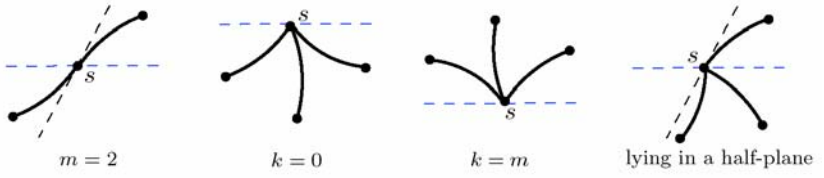


Figure 8. Some configurations of Case (ii) of junction site s.

We separate two cases: Case (i) when the union of cone of clearance of m generator arcs covers D, and Case (ii) when it does not cover D. Note that the case where $m = 2$ is classified into Case (ii) since the cone of clearance of an open arc less than a semi-circle is a two-dimensional open set less than

a half-plane and thus the union of cone of clearance of two generator arcs cannot cover D. In the same reason, Case (ii) includes the cases where $k = 0$, $k = m$, or all generator arcs lie in a half-plane locally within D. Refer to Figure 8. In the following theorem, we show that the junction site s becomes a Voronoi vertex at which m Voronoi edges meet in Case (i) and $(m + 1)$ Voronoi edges meet in Case (ii).

Theorem 2. *Let $s \in \mathcal{S}$ be a junction site described above and m, k be the numbers defined above. When the sweep line ℓ passes through s, the changes in the beach line and in the Voronoi diagram are as follows*:

- $1 \le k \le m$: $2(k-1)$ *old edges disappear from the beach line, and* $2(m-k)$ *new edges in Case* (i) *and* $2(m-k)+1$ *new edges in Case* (ii) *appear in the beach line. In* $\mathcal{V}(\mathcal{S})$, *s becomes a new Voronoi vertex which* $(k-1)$ *old Voronoi edges end in and* $(m-k+1)$ *new Voronoi edges in Case* (i) *and* $(m-k+2)$ *new Voronoi edges in Case* (ii) *start from.*
- $k = 0$: $(2m+4)$ *new edges appear in the beach line, and in* $\mathcal{V}(\mathcal{S})$ $(m+2)$ *new Voronoi edges are generated and among them* $(m+1)$ *edges start from s.*

Proof. Refer to Figure 9 for illustration of Case (i) and Figure 10 for Case (ii). In Case (i), since the union of cone of clearance of generator arcs covers D, $\mathcal{VR}(s, \mathcal{S})$ is just a single point s. Each generator arc has a two-dimensional Voronoi region with non-empty interior, so the neighborhood D of s is divided into m Voronoi regions, each having non-empty interior and sharing s in the boundary. On the other hand, in Case (ii), $\mathcal{VR}(s, \mathcal{S})$ is a two-dimensional region with non-empty interior, bounded by two rays emanating from s and normal at s to a pair of generator arcs. In fact, $\mathcal{VR}(s, \mathcal{S})$ may be a line segment or ray in case where $m = 2$ and the two arcs have the same tangent direction at s, as in Figure 1, but this case can be regarded as a degenerate situation that the two rays bounding the region $\mathcal{VR}(s, \mathcal{S})$ become collinear. So, in Case (ii), the neighborhood D of s is

divided into $(m+1)$ Voronoi regions, one for each of m arcs and one for the junction site s. This proves that s is a Voronoi vertex at which m Voronoi edges in Case (i) and $(m+1)$ Voronoi edges in Case (ii) meet.

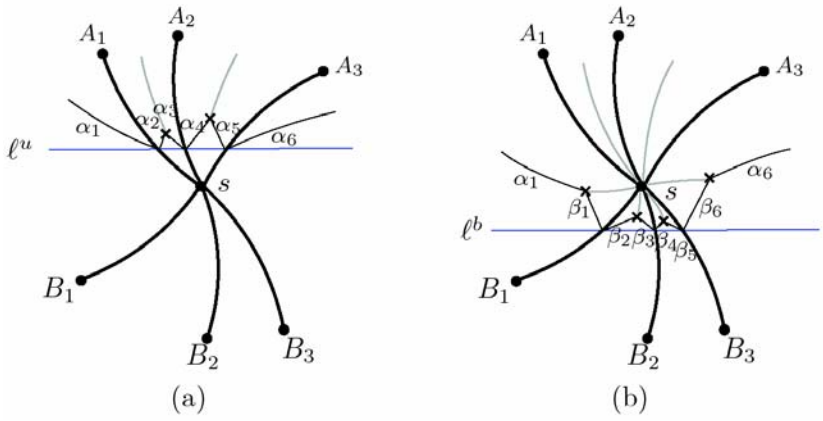


Figure 9. In Case (i), Voronoi diagram and the beach line, (a) before and (b) after a cross event.

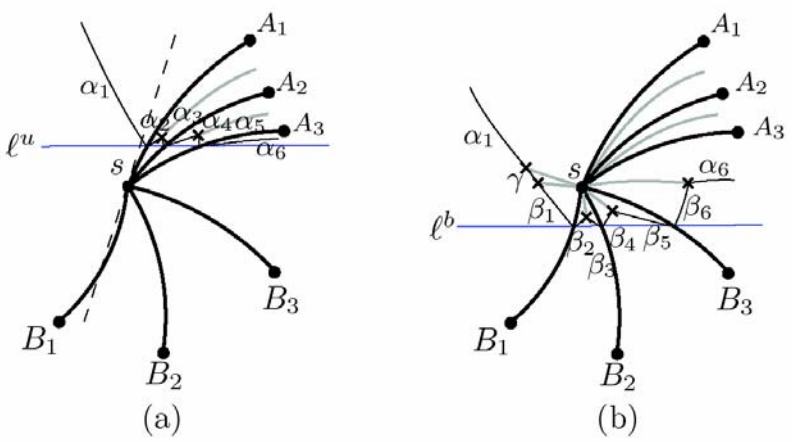


Figure 10. In Case (ii), Voronoi diagram and the beach line, (a) before and (b) after a cross event.

In order to examine further, we denote by A_1, ..., A_k and B_1, ..., B_{m-k}, the arcs lying above and below ℓ locally within D, respectively, in increasing x-coordinate order. Assume first that $1 \le k \le m$. Just before this

event, in the beach line of ℓ^u, there must be $2k$ edges $\alpha_1, ..., \alpha_{2k}$ defined by ℓ^u and k arcs such that: for each i from 1 to k, the edges α_{2i-1} and α_{2i} are the sets of points equidistant from ℓ^u and A_i, one lying inside and the other lying outside of the generator circle of A_i, and for each i from 1 to $k-1$, the breakpoint $\alpha_{2i} \cap \alpha_{2i+1}$ traces out an old Voronoi edge defined by A_i and A_{i+1}. While the sweep line moves downward from ℓ^u to ℓ, all beach line edges except α_1 and α_{2k} shrink to s, thus the $(k-1)$ old Voronoi edges end in the new Voronoi vertex s.

To see the situation when the sweep line moves downward from ℓ to ℓ^b, we first consider Case (i). In the beach line of ℓ^b, $2(m-k)$ new edges $\beta_1, ..., \beta_{2(m-k)}$ appear between α_1 and α_{2k}, where for each i from 1 to $m-k$, the edges β_{2i-1} and β_{2i} are the sets of points equidistant from ℓ^b and B_i, one lying inside and the other lying outside of the generator circle of B_i, and for each i from 1 to $m-k-1$, the breakpoint $\beta_{2i} \cap \beta_{2i+1}$ traces out a new Voronoi edge defined by B_i and B_{i+1}. Moreover, two breakpoints $\alpha_1 \cap \beta_1$ and $\beta_{2(m-k)} \cap \alpha_{2k}$ trace out two new Voronoi edges defined by A_1 and B_1 and by A_k and B_{m-k}. Thus, in Case (i), $2(m-k)$ new edges appear in the beach line and $(m-k-1)+2 = m-k+1$ new Voronoi edges stem from the Voronoi vertex s in $\mathcal{V}(\mathcal{S})$.

Now consider Case (ii) where $1 \le k \le m-1$. The situation of new beach line edges and new Voronoi edges defined by B_is is same as in Case (i); $2(m-k)$ new edges $\beta_1, ..., \beta_{2(m-k)}$ appear between α_1 and α_{2k} in the beach line of ℓ^b and the $(m-k-1)$ breakpoints between them trace out $(m-k-1)$ new Voronoi edges defined among B_is. The difference is that a new edge γ defined by ℓ^b and s appears either between α_1 and β_1 or between $\beta_{2(m-k)}$ and α_{2k}, not both. Thus, in the former case, three breakpoints $\alpha_1 \cap \gamma$, $\gamma \cap \beta_1$ and $\beta_{2(m-k)} \cap \alpha_{2k}$ trace out three new Voronoi

edges defined by A_1 and s, by s and B_1 and by B_{m-k} and A_k, respectively, and in the latter case, the breakpoints $\alpha_1 \cap \beta_1$, $\beta_{2(m-k)} \cap \gamma$ and $\gamma \cap \alpha_{2k}$ trace out new Voronoi edges defined by A_1 and B_1, by B_{m-k} and s and by s and A_k, respectively. Therefore, in total, $2(m-k)+1$ new edges appear in the beach line and $(m-k-1)+3 = m-k+2$ new Voronoi edges stem from s in $\mathcal{V}(\mathcal{S})$.

In Case (ii) where $k = m$, no generator arc of s lies below ℓ locally in D. So, while the sweep line moves downward from ℓ to ℓ^b, only one new beach line edge γ defined by ℓ^b and s appears between α_1 and α_{2m}, and two breakpoints $\alpha_1 \cap \gamma$ and $\gamma \cap \alpha_{2m}$ trace out two new Voronoi edges, stemming from s and defined by A_1 and s and by s and A_m, respectively.

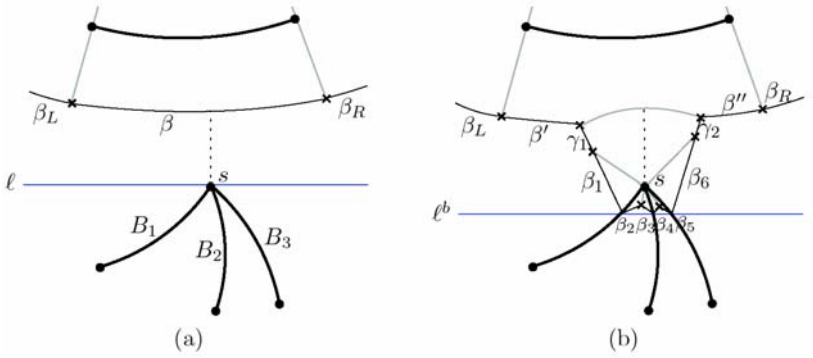


Figure 11. Case (ii) where $k = 0$.

Finally, consider Case (ii) where $k = 0$. Refer to Figure 11. In this case, no generator arc of s lies above ℓ locally in D, so the situation is similar as in a start event. In the beach line of ℓ^b, an old edge β vertically above s is split into two new edges β', β'', and between them, two new edges γ_1 and γ_2 defined by ℓ^b and s appear. The breakpoints $\beta' \cap \gamma_1$ and $\gamma_2 \cap \beta''$ trace out leftward and rightward a new Voronoi edge, defined by s and the site corresponding to β and stemming from the point of β vertically above s. Moreover, $2m$ new edges $\beta_1, \ldots, \beta_{2m}$ defined by ℓ^b and B_is appear

between γ_1 and γ_2, where the breakpoints $\gamma_1 \cap \beta_1$ and $\beta_{2m} \cap \gamma_2$ trace out two new Voronoi edges defined by s and B_1 and by B_m and s, respectively, and for each i from 1 to $m-1$, the breakpoint $\beta_{2i} \cap \beta_{2i+1}$ traces out a new Voronoi edge defined by B_i and B_{i+1}. Therefore, in total, $(4+2m)$ new edges appear in the beach line, $(m+2)$ new Voronoi edges are generated in $\mathcal{V}(\mathcal{S})$, among which $(m+1)$ edges start from s. □

Due to the theorem, what remains is to verify operations for the event queue $\mathcal{Q}$. Assume first that $1 \le k \le m-1$. In the beach line of ℓ^u, we denote by α_1 and α_{2k} the leftmost and the rightmost edges among the $2k$ edges defined by ℓ^u and the k arcs (lying above ℓ), and by β_L and β_R the left and the right neighbors of α_1 and α_{2k}, respectively. Similarly, in the beach line of ℓ^b, denote by β_1 and $\beta_{2(m-k)}$ the leftmost and the rightmost edges among the $2(m-k)$ new edges defined by ℓ^b and the $(m-k)$ arcs (lying below ℓ).

In Cases (i) and (ii), we first remove from $\mathcal{Q}$ all circle events corresponding to the $2k$ old beach line edges, and push into $\mathcal{Q}$ all feasible circle events defined by triples of consecutive edges with any of $2(m-k)$ new edges in the middle. We then check and push more circle events: those defined by $(\beta_L, \alpha_1, \beta_1)$ and $(\beta_{2(m-k)}, \alpha_{2k}, \beta_R)$ in Case (i), those defined by $(\beta_L, \alpha_1, \gamma)$, $(\alpha_1, \gamma, \beta_1)$, $(\beta_{2(m-k)}, \alpha_{2k}, \beta_R)$ if in Case (ii), a new edge γ defined by ℓ^b and s appears between α_1 and β_1, and those defined by $(\beta_L, \alpha_1, \beta_1)$, $(\beta_{2(m-k)}, \gamma, \alpha_{2k})$, $(\gamma, \alpha_{2k}, \beta_R)$ if in Case (ii), the edge γ appears between $\beta_{2(m-k)}$ and α_{2k}.

Assume now that $k = 0$. In the beach line of ℓ^u, denote by β the edge vertically above s, and by β_L, β_R the left and the right neighbors of β. In the beach line of ℓ^b, denote by β', β'' the split edges of β, by γ_1, γ_2 the new

edges defined by ℓ^b and s, and by β_1, β_{2m} the leftmost and the rightmost edges among $2m$ new edges defined by ℓ^b and the m arcs (lying below ℓ). We first remove from $\mathcal{Q}$ the circle event defined by $(\beta_L, \beta, \beta_R)$, and then push into $\mathcal{Q}$ all circle events defined by triples from $2m$ new edges β_is and those defined by $(\beta_L, \beta', \gamma_1)$, $(\beta', \gamma_1, \beta_1)$, $(\beta_{2m}, \gamma_2, \beta'')$ and $(\gamma_2, \beta'', \beta_R)$.

HandleCrossEvent(Junction site s with $m \geq 2$ generator arcs)

1. Determine if the union of cone of clearance of m generator arcs covers a sufficiently small neighborhood D of s that contains no part of the other sites of $\mathcal{S}$ except s and its generator arcs. If yes, set a boolean variable f_1 to be 'TRUE', and otherwise 'FALSE'. If all generator arcs lie above locally within D, then go to Step 5.
2. Let $A_1, ..., A_k$ and $B_1, ..., B_{m-k}$ be the generator arcs lying above and below ℓ locally within D, respectively, in increasing x-coordinate order.
   - Find $2k$ consecutive edges $\alpha_1 \cdots \alpha_{2k}$ in $\mathcal{T}$ such that α_{2i-1} and α_{2i} are defined by ℓ^u and A_i and for each i from 1 to $k-1$, the breakpoint $\alpha_{2i} \cap \alpha_{2i+1}$ traces out Voronoi edge defined by A_i and A_{i+1}.
   - Delete all edges except α_1 and α_{2k} from $\mathcal{T}$, and delete all circle events corresponding to $\alpha_1 \cdots \alpha_{2k}$ from $\mathcal{Q}$. In $\mathcal{V}(\mathcal{S})$, add s as a new Voronoi vertex terminating the $(k-1)$ Voronoi edges.
   - If $k = m$, then do the following and exit: Insert into $\mathcal{T}$ a new edge γ defined by ℓ^b and s between α_1 and α_{2m}, and add into $\mathcal{V}(\mathcal{S})$ two new Voronoi edges stemming from s, defined by A_1 and s and by s and A_m, and traced out by breakpoints $\alpha_1 \cap \gamma$ and $\gamma \cap \alpha_{2m}$, respectively. Insert into

$\mathcal{Q}$ any feasible circle events defined by triples of consecutive edges with any of α_1, γ, α_{2k} in the middle.

3. Insert into $\mathcal{T}$ between α_1 and α_{2k} $2(m-k)$ edges $\beta_1, ..., \beta_{2(m-k)}$ defined by ℓ^u and B_is. If f_1 = 'FALSE', then insert one more edge γ defined by ℓ^b and s, either between α_1 and β_1 (setting f_2 = 'TRUE') or between $\beta_{2(m-k)}$ and α_{2k} (setting f_2 = 'FALSE').
4. For $\mathcal{V}(\mathcal{S})$, generate $(m-k-1)$ new Voronoi edges stemming from s, defined by B_i and B_{i+1} and traced by the breakpoints $\beta_{2i} \cap \beta_{2i+1}$.
   - If f_1 = 'TRUE', then add two more Voronoi edges stemming from s, defined by A_1 and B_1 and by A_k and B_{m-k} and traced out by the breakpoints $\alpha_1 \cap \beta_1$ and $\beta_{2(m-k)} \cap \alpha_{2k}$, respectively. Insert into $\mathcal{Q}$ any feasible circle events defined by triples of consecutive edges with any of α_1, β_1, ..., $\beta_{2(m-k)}$, α_{2k} in the middle.
   - If f_1 = 'FALSE' and f_2 = 'TRUE', then add three more edges stemming from s, defined by A_1 and s, by s and B_1, by B_{m-k} and A_k, and traced out by breakpoints $\alpha_1 \cap \gamma$, $\gamma \cap \beta_1$, $\beta_{2(m-k)} \cap \alpha_{2k}$, respectively. Insert into $\mathcal{Q}$ feasible circle events defined by triples of consecutive edges with any of α_1, γ, β_1, ..., $\beta_{2(m-k)}$, α_{2k} in the middle.
   - If f_1 = 'FALSE' and f_2 = 'FALSE', then add three more edges stemming from s, defined by A_1 and B_1, by B_{m-k} and s, by s and A_k, and traced out by breakpoints $\alpha_1 \cap \beta_1$, $\beta_{2(m-k)} \cap \gamma$, $\gamma \cap \alpha_{2k}$, respectively. Insert into $\mathcal{Q}$ any feasible circle events defined by triples of consecutive edges with any of α_1, β_1, ..., $\beta_{2(m-k)}$, γ, α_{2k} in the middle.

5. Locate the beach line edge β vertically above s in $\mathcal{T}$. Split β into β', β'', and insert between β' and β'' new edges γ_1, γ_2 defined by s and the site of β, and between γ_1 and γ_2 $2m$ edges $\beta_1, ..., \beta_{2m}$ defined by ℓ^b and B_is. For $\mathcal{V}(\mathcal{S})$, generate a new Voronoi edge, defined by s and the site corresponding to β and traced out leftward and rightward by the breakpoints $\beta' \cap \gamma_1$ and $\gamma_2 \cap \beta''$. Add $(m+1)$ more edges stemming from s: two defined by s and B_1 and by B_m and s and traced out by breakpoints $\gamma_1 \cap \beta_1$ and $\beta_{2m} \cap \gamma_2$, respectively, and $(m-1)$ defined by B_i and B_{i+1} and traced by $\beta_{2i} \cap \beta_{2i+1}$ for each i from 1 to $m-1$. Delete all circle events corresponding to β from $\mathcal{Q}$ and add feasible circle events defined by new triples.

4. The Algorithm and its Analysis

We now present the overall structure of the plane sweep algorithm and analyze its worst-case time complexity.

Plane Sweep Algorithm $(\mathcal{S})$

Input: A set of n circular arcs, possibly sharing endpoints.

Output: Voronoi diagram $\mathcal{V}(\mathcal{S})$

1. Initialize the data structures $\mathcal{T}$, $\mathcal{Q}$ and $\mathcal{V}(\mathcal{S})$.
2. Push into $\mathcal{Q}$ all start/end and cross events corresponding to the endpoints and junction points of $\mathcal{S}$, respectively, and then all top or merge events by checking for each arc s if the topmost or bottommost point of its generator circle is contained in the interior of s.
3. while ($\mathcal{Q}$ is not empty), do:

(a) Pop the event with largest y-coordinate from $\mathcal{Q}$.

(b) Depending on the type of the event, execute one of the following:

i. HandleTopEvent(Topmost point T of arc site s)

ii. HandleStartEvent(Endpoint site s of a generator arc s')

iii. HandleEndEvent(Endpoint site s of a generator arc s')

iv. HandleCircleEvent(α)

v. HandleMergeEvent(Bottommost point B of arc site s)

vi. HandleCrossEvent(Junction site s with $m \geq 2$ generator arcs)

4. Compute a bounding circle that contains all Voronoi vertices in the interior, and connect the remaining half-infinite Voronoi edges to the circle. Update the information of endpoints and cells of them appropriately.

Initialization and postprocessing. In Step 2, as an initialization process, we push all events except circle events into $\mathcal{Q}$. The input $\mathcal{S}$ consists of n circular arcs possibly sharing endpoints, each being less than a semi-circle, so the total number of such events is at most $3n$ since each circular arc can have at most two endpoints or junctions and at most one topmost or bottommost point of its generator circle. Checking of the topmost or bottommost point of the generator circle is contained in the interior of an arc costs $O(1)$ time, and pushing an event into the priority queue $\mathcal{Q}$ costs $O(\log|\mathcal{Q}|)$. Thus, the time-complexity of Step 2 is $O(n \log n)$.

Since the number of Voronoi edges in $\mathcal{V}(\mathcal{S})$ is at most $3 \cdot 3n$ by Property 1, in Step 4, computing a bounding circle containing all Voronoi vertices and updating the information of unbounded Voronoi regions and the half-infinite Voronoi edges between them require at most $O(n)$ time.

Event processing. Now we analyze the time required for handling all events in Step 3. For every event except merge or cross events, we show that its subroutine performs $O(1)$ number of search, insertion and deletion in $\mathcal{T}$, $O(1)$ number of insertion and deletion of circle events in $\mathcal{Q}$, and $O(1)$

number of addition or update in $\mathcal{V}(\mathcal{S})$. The number of these events is less than 3n, so the total number of operations required in $\mathcal{T}$, $\mathcal{Q}$ and $\mathcal{V}(\mathcal{S})$ to process all these events is $O(n)$. For a merge event, it can be shown that the number of operations performed in the structures is linear in the complexity of Voronoi region of the arc site causing this event, so the total number of operations to process all merge events is $O(n)$. For a cross event, the number of operations required is shown to be linear in the number of generator arcs at the junction site causing that event, implying that the total number of operations to process all cross events is $O(n)$. Combining these results leads to the conclusion that the space-complexity of $\mathcal{T}$, $\mathcal{Q}$, $\mathcal{V}(\mathcal{S})$ is $O(n)$, thus each operation in $\mathcal{T}$ and $\mathcal{Q}$ takes $O(\log n)$ time and therefore it takes $O(n \log n)$ time to process all events in Step 3.

i. HandleTopEvent(Topmost point T of arc site s): In this event, one searching, one deletion and five insertions are performed in $\mathcal{T}$, one deletion and two insertions of circle events are used in $\mathcal{Q}$, and one Voronoi edge is generated in $\mathcal{V}(\mathcal{S})$.

ii. HandleStartEvent(Endpoint site s of a generator arc s') : In this event, one searching, one deletion and six insertions are performed in $\mathcal{T}$, one deletion and four insertions of circle events are used in $\mathcal{Q}$, and two Voronoi edges are generated in $\mathcal{V}(\mathcal{S})$.

iii. HandleEndEvent(Endpoint site s of a generator arc s') : In this event, two searches and one insertion are performed in $\mathcal{T}$, two insertions of circle events are used in $\mathcal{Q}$, and one Voronoi edge is generated in $\mathcal{V}(\mathcal{S})$.

iv. HandleCircleEvent(α) : In this event, two searching of predecessor and successor of α and one deletion are required in $\mathcal{T}$, two deletions and two insertions of circle events are performed in $\mathcal{Q}$, and generating one vertex and one edge with updating the endpoints of three edges are necessary in $\mathcal{V}(\mathcal{S})$.

v. HandleMergeEvent(Bottommost point B of arc site s): As seen in Figure 7, in this event, the number of operations required in $\mathcal{T}$, $\mathcal{Q}$ and $\mathcal{V}(\mathcal{S})$ is linear in the number of the other sites touching the largest empty disk (indicated by dotted lines in the figure) whose bottommost point coincides with B. Denote such number by m. Then this subroutine performs $(m+4)$ searches, $(m+4)$ deletions and one insertion in $\mathcal{T}$, $(m+4)$ deletions and one insertion in $\mathcal{Q}$, and generating at most one vertex with updating the endpoints of $(m+1)$ old edges in $\mathcal{V}(\mathcal{S})$. Thus, the number of operations performed in $\mathcal{T}$, $\mathcal{Q}$ and $\mathcal{V}(\mathcal{S})$ is $O(m)$, which is linear in the complexity of Voronoi region of s. Summing the complexity of Voronoi region over all arc sites is $O(n)$.

vi. HandleCrossEvent(Junction site s with $m \geq 2$ generator arcs): In this event, as shown in Theorem 2, $O(m)$ operations are required in both $\mathcal{T}$ and $\mathcal{Q}$, and at most $(m+1)$ new edges and one vertex(s) are generated in $\mathcal{V}(\mathcal{S})$. So, the total number of operations to process all cross events is $O(n)$.

5. Conclusions

In this note, we have presented a plane sweep algorithm that computes the Voronoi diagram of n circular arcs, possibly sharing endpoints, in $O(n \log n)$ time. Our algorithm can be applied to a set of points, circles, circular and line segments, possibly intersecting each other, with a simple modification; Compute all intersection points, and split the objects into junction points, endpoints, open arcs and circles. Circles are divided into three equal-length sub-arcs, e.g., by cutting at the points $\left(r\cos\frac{2i\pi}{3}, r\sin\frac{2i\pi}{3}\right)$ for $i = 0, 1, 2,$ and open arcs longer than semi-circles are divided into two equal-length sub-arcs. If the original set consists of n

objects with k intersections, then the modified algorithm can compute its Voronoi diagram in $O((n+k)\log(n+k))$ time.

Computing Voronoi diagram or medial axis of circular arcs has many applications [1]. Modeling the boundary of objects with circular splines makes the input size significantly smaller, and piecewise circular representation of the environments in robot path planning or shape offset computation makes the problem much simpler and easier to handle.

References

[1] O. Aichholzer, W. Aigner, F. Aurenhammer, T. Hackl, B. Jüttler, E. Pilgerstorfer and M. Rabl, Divide-and-conquer for voronoi diagrams revisited. Comput. Geom. 43 (2010), 688-699.

[2] O. Aichholzer, W. Aigner, F. Aurenhammer, T. Hackl, B. Jüttler and M. Rabl, Medial axis computation for planar free-form shapes, Computer-Aided Design 41 (2009), 339-349.

[3] H. Alt, O. Cheong and A. Vigneron, The Voronoi diagram of curved objects, Discrete Comput. Geom. 34 (2005), 439-453.

[4] M. de Berg, O. Cheong, M. van Kreveld and M. Overmars, Computational Geometry: Algorithms and Applications, Springer-Verlag, 3rd ed., 2008.

[5] S. J. Fortune, A sweepline algorithm for Voronoi diagrams, Algorithmica 2 (1987), 153-174.

[6] M. Held and S. Huber, Topology-oriented incremental computation of Voronoi diagrams of circular arcs and straight-line segments, Computer-Aided Design 41 (2009), 327-338.

[7] L. Jin, D. Kim, L. Mu, D.-S. Kim and S.-M. Hu, A sweepline algorithm for Euclidean Voronoi diagram of circles, Computer-Aided Design 38 (2006), 260-272.

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

Hyeon-Suk Na: hsnaa@ssu.ac.kr