International Journal of Computational Geometry & Applications

THE L_∞ VORONOI DIAGRAM OF SEGMENTS AND VLSI APPLICATIONS *

EVANTHIA PAPADOPOULOU
IBM TJ Watson Research Center
P.O. Box 218, Yorktown Heights, NY 10598, USA
evanthia@watson.ibm.com

and

D. T. LEE†
Institute of Information Science, Academia Sinica
Nankang, Taipei, Taiwan
dtlee@iis.sinica.edu.tw

Received 16 May 1999
Revised 27 December 2000
Communicated by F. P. Preparata

ABSTRACT

In this paper we address the L_∞ Voronoi diagram of polygonal objects and present applications in VLSI layout and manufacturing. We show that the L_∞ Voronoi diagram of polygonal objects consists of straight line segments and thus it is much simpler to compute than its Euclidean counterpart; the *degree* of the computation is significantly lower. Moreover, it has a natural interpretation. In applications where Euclidean precision is not essential the L_∞ Voronoi diagram can provide a better alternative. Using the L_∞ Voronoi diagram of polygons we address the problem of calculating the *critical area* for shorts in a VLSI layout. The critical area computation is the main computational bottleneck in VLSI yield prediction.

Keywords: Voronoi diagram, L_∞ metric, algorithmic degree, critical area, VLSI layout

1. Introduction

The Voronoi diagram of polygonal objects has been given considerable attention because of its numerous applications in diverse areas such as biology, geography, robot motion planning, computer graphics. In this paper we address the L_∞ Voronoi diagram of line-segments and present applications in VLSI layout and

*Extended Abstract appeared in *Proc. 9th International Symposium on Algorithms and Computation*, December 1998, Taejon, Korea, *LNCS 1533*, 9-18.

†Supported in part by the National Science Foundation under the Grant CCR-9731638, and by the National Science Council under the grants NSC89-2213-E-001-012, NSC89-2219-E-001-002, and NSC89-2219-E-001-003. Also with the Department of Electrical Engineering and Computer Science, University of Illinois at Chicago, Chicago, IL 60607-7053.

manufacturing. It is well known that the ordinary Voronoi diagram of polygonal objects has linear combinatorial complexity and consists of straight-line segments and parabolic arcs. Several efficient algorithms using divide and conquer, plane sweep, or incremental construction are known for its computation (see Refs. [2,3] for a survey). However, the existence of parabolic arcs in the diagram makes it hard to compute in practice. Existing algorithms assume exact computation over the real numbers while in reality computer calculations have finite precision. To formalize the precision to which arithmetic calculations need to be executed in a robust implementation, Liotta, Preparata, and Tamassia,[15] introduced a new complexity model called the *degree* of an algorithm. Namely, an algorithm has degree d if its test computations involve the evaluation of multivariate polynomials of arithmetic degree at most d. In the construction of the Voronoi diagram of segments, using a randomized incremental approach, Burnikel showed that the well-known *incircle test* can be answered correctly with degree 40, Refs. [6,7]. However, 40 is too high for practical problems involving Voronoi diagrams that require robust and fast implementations. In a recent paper[1] Aichholzer and Aurenhammer introduced a new type of skeleton for polygonal objects in the plane called the *straight skeleton*. The straight skeleton consists of angular bisectors between edges and thus consists of straight line segments. The straight skeleton captures the *shape* of the defining elements in a natural manner. Its main advantage over the Voronoi diagram is the elimination of parabolic arcs. However, straight skeletons do not provide the proximity information that Voronoi diagrams do and thus, they do not always provide an alternative solution to Voronoi diagrams.

In applications where Euclidean accuracy is not particularly important a practical solution to the high degree problem of the Euclidean Voronoi diagram of segments may be the use of a different geometry, in particular, the L_∞ (resp. L_1) metric. Note that the Voronoi diagram of segments in arbitrary orientations in metrics derived from fixed orientations such as the L_∞ or the L_1 metric, has not been given any attention in the literature. As it is shown in this paper, the L_∞ Voronoi diagram of segments consists of straight line segments and its combinatorial complexity is similar to the Euclidean case. If the input vertices are on rational coordinates the Voronoi vertices are also rational. Furthermore, the L_∞ *in-circle* test for segments can be answered with degree 5 (see Section 4).

An intuitive way to view the L_∞ (resp. L_1) Voronoi diagram of polygonal objects is to view it as the locus of points corresponding to centers of isothetic squares[a] (resp. isothetic square diamonds[b]) that touch the boundary in at least two points. In contrast, the Euclidean Voronoi diagram can be regarded as the locus of centers of circles that touch the boundary in at least two points. It is not hard to see that the L_∞ and L_1 Voronoi diagrams of segments are equivalent under 45° rotation.[8] In the case of rectilinear segments, the L_∞ Voronoi diagram coincides with the straight-skeleton of Ref. [1]. Similarly, in the case of segments of slope ± 1 and the L_1 metric. In this paper we focus in the L_∞ case due to applications in VLSI

[a] An isothetic square is one with sides parallel to the coordinate axes.
[b] An isothetic square diamond is an isothetic square rotated by 45°.

layout and manufacturing, as described below. Note that VLSI shapes consist in the majority of axis-parallel edges but are not necessarily rectilinear. Thus, the L_∞ metric is very appropriate for proximity problems involving shapes of VLSI designs.

This paper consists of two parts. In the first part we consider the L_∞ Voronoi diagram of arbitrary segments and provide an $O(n \log n)$-time plane-sweep algorithm of degree 7 to construct it. The algorithm extends the wavefront approach of Ref. [9] for points to handle arbitrary line segments in the L_∞ metric; it is a refinement of the original plane sweep approach given by Fortune.[12] In the second part we present an important application of L_∞ Voronoi diagrams in predicting the *yield* of a VLSI chip. The *yield* of a VLSI chip is the percentage of functional chips among all chips manufactured. Predicting the yield of a chip prior to fabrication is essential in order to control the cost of manufacturing. Models for yield estimation are based on the concept of *critical area* which is a measure reflecting the sensitivity of the layout to spot defects caused by particles such as dust and other contaminants in materials and equipment. "Extra material" defects cause shorts between different conducting regions and represent the main reason for yield loss during manufacturing. For information on yield estimation and spot defects see for example Refs. [10,11,14,16,17,20,25,26,27]. In this paper we generalize the result of Ref. [19] on critical area calculation to general layouts consisting of edges in arbitrary orientations. In particular, we use the proximity information preserved in the L_∞ Voronoi diagram of arbitrary shapes on a layer of a VLSI design and show that the critical area for shorts can be computed as a function of the 2nd order L_∞ Voronoi diagram of shapes on that layer. In Ref. [19] the same result was shown for restricted *ortho-45* layouts i.e., layouts consisting of shapes with edges in four orientations: horizontal, vertical and slope ± 1.

Applications of Voronoi diagrams in extracting from the physical description of a design the equivalent resistance are discussed in Refs. [18,23]. The L_∞ version would be as good in this case[22] but much simpler to obtain. Due to the simplicity of L_∞ Voronoi diagrams we expect them to also find applications in other areas. For example, in Ref. [24] an automatic generation for finite element meshes for multiply connected planar domains with polygonal boundaries is described. The Euclidean Voronoi diagram of the domain was used as a starting point and could be substituted by the L_∞ version[22] which is easier to compute.

This paper is organized as follows. In Section 2 we investigate L_∞ Voronoi diagrams of polygonal objects. In Section 3, we provide a simple $O(n \log n)$ plane sweep algorithm and in Section 4, we show that the *degree* of this algorithm is 7. In Section 5, we give an important application in VLSI yield prediction for computing the *critical area* of a VLSI layout.

2. L_∞ Voronoi Diagrams

The L_∞ distance between two points $p = (x_p, y_p)$ and $q = (x_q, y_q)$ is the maximum between the horizontal and the vertical distance between p and q i.e., $d(p,q) = \max\{d_x(p,q), d_y(p,q)\}$ where $d_x(p,q) = |x_p - x_q|$ and $d_y(p,q) = |y_p - y_q|$. The L_∞ distance between a point p and a line l is $d(p,l) = \min\{d(p,q), \forall q \in l\}$.

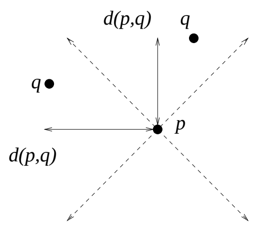

Fig. 1. The $45°$ rays emanating from p partition the plane into four quadrants.

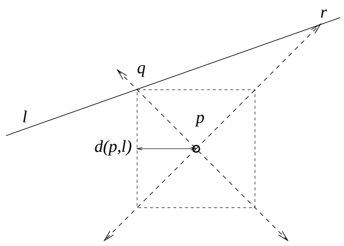

Fig. 2. The L_∞ distance form p to l is $d_x(p,q)$.

Intuitively, the L_∞ distance between two elements (points or lines) can be defined in terms of the smallest isothetic square touching the two elements. We shall refer to a line of slope (± 1) simply as a *45-degree* line, denoted $45°$ line.

Consider a point p and the four $45°$ rays emanating away from p (see Fig. 1). They partition the plane into four quadrants. In each quadrant the L_∞ distance between p and any point q simplifies to either the vertical (q in upper and lower quadrant) or the horizontal (q in right and left quadrant) distance between p and q. The L_∞ distance between point q and a horizontal or vertical line is the distance between q and its orthogonal projection to the line. An arbitrary line l that does not contain p intersects two of the $45°$ rays emanating from p at points q and r respectively. (If l is $45°$ the second intersection is at infinity). Assuming that $d_x(p,q) < d_x(p,r)$, the L_∞ distance between p and l is $d(p,l) = d_x(p,q)$ (see Fig. 2).

Lemma 1 *If the L_∞ distance from a point p to a line l of slope $\pm s, s \geq 0$ is r, the Euclidean distance from p to l is $S_l r$, where $S_l = \frac{s+1}{\sqrt{s^2+1}}$ for $s \neq 0, \infty$ and $S_l = 1$ for $s = 0, \infty$.*

Proof. Clearly $S_l = 1$ for $s = 0, \infty$. Consider a line l of non-negative slope $s \neq \infty$ as shown in Fig. 2. Considering q, the intersection of l with the (-1)-slope ray emanating from p, as the origin of the coordinate system, l is given by $y = sx$ and $p = (r, -r)$. Then the Euclidean distance from p to l is $d_e(p,l) = (sr+r)/(\sqrt{s^2+1})$. Similarly for a line of negative slope. □

Consider an arbitrary line segment s; it consists of three *elements*: two endpoints s_1, s_2, where s_1 is assumed to be at the top, and an open line-segment l. Consider the two parallel $45°$ lines passing through s_1 and s_2 whose slope is of opposite sign than the slope of s (see Fig. 3). They partition the plane into three regions where the L_∞ distance between a point q and segment s, $d(q,s)$, simplifies to $d(q,s_1)$,

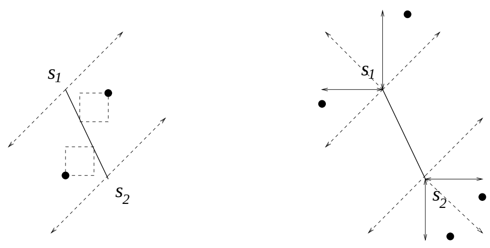

Fig. 3. Six 45° induced by a line segment.

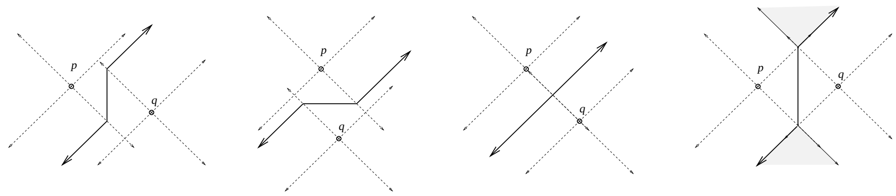

Fig. 4. The L_∞ bisector of two points.

$d(q, l)$, or $d(q, s_2)$ depending on whether q lies in the upper, middle, or bottom region respectively. The top and bottom regions are further subdivided into two quadrants by the 45° rays emanating from s_1 and s_2 having slope of the same sign as s. For a point q in the north or south quadrant, $d(q, s) = d_y(q, s_1)$ and $d(q, s) = d_y(q, s_2)$ respectively, and for q in the east or west quadrant $d(q, s) = d_x(q, s_1)$ and $d(q, s) = d_x(q, s_2)$ respectively.

The *bisector* of two elements (points or lines) is the locus of points equidistant from the two elements. In L_∞, the bisector can be regarded as the locus of centers of squares touching the two elements. Figs. 4 and 5 illustrate the L_∞ bisector of two points and two lines respectively. In case of two points along the same horizontal or vertical line the bisector consists of a line segment and two unbounded regions (shaded regions in Fig. 4). Without creating any significant difference we will assign one region to each point and consider only the outermost boundary of the bisecting region as the bisector (thick lines in Fig. 4). Alternatively, both regions could be assigned to the *dominating* point according to the lexicographic order as proposed in Ref. [2]. The L_∞ bisector of two lines consists of two lines with slopes given by the following lemma.

Lemma 2 *The L_∞ bisector of two non-parallel lines l_1 and l_2 of slopes b_1 and b_2 ($b_1, b_2 \neq \infty$), respectively, consists of two lines as follows:*

- *If $0 \leq b_1 < b_2$ the bisector consists of two lines of slopes -1 and $\frac{(b_1+b_2+2b_1b_2)}{(b_1+b_2+2)}$.*
- *If $b_2 < b_1 \leq 0$ the slopes of the bisector are $+1$ and $\frac{(-b_1-b_2+2b_1b_2)}{(b_1+b_2-2)}$.*
- *If $b_1 > 0$ and $b_2 < 0$ the bisector has slopes $\frac{b_2-b_1+2b_1b_2}{b_1+b_2}$ and $\frac{b_1+b_2}{b_1-b_2+2}$.*

If l_2 is vertical the L_∞ bisector consists of a 45° line and a line of slope $(1+2b_1)$ for $b_1 > 0$ (resp. $(-1+2b_1)$ for $b_1 < 0$). The L_∞ bisector of two parallel lines is a single parallel line as in the Euclidean metric.

Proof. Let's consider the point of intersection of lines l_1 and l_2 as the origin of

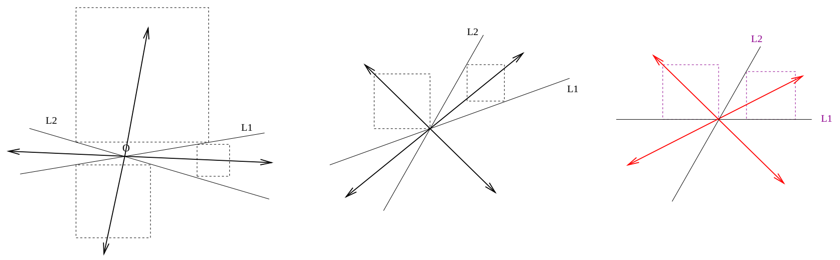

Fig. 5. The L_∞ bisector of two lines.

the coordinate system (see Fig. 5). Then the equations of l_1 and l_2 are $y = b_1 x$ and $y = b_2 x$ respectively. Let (x_0, y_0) be any point along the bisector. (x_0, y_0) must be the center of a square $\mathcal{D}$ touching l_1 and l_2 at two points. Let x_1, x_2, y_1, y_2, where $x_1 < x_2, y_1 < y_2$, denote the x and y coordinates of $\mathcal{D}$. Clearly, $x_0 = x_1 + r/2$ and $y_0 = y_1 + r/2$ where $r = x_2 - x_1 = y_2 - y_1$. For simplicity we assume that $y_2 > 0$.

Let's first assume that $b_1 > 0, b_2 < 0$. Then $\mathcal{D}$ must touch l_1 and l_2 with two adjacent corners . If $\mathcal{D}$ touches l_1, l_2 with its lower horizontal side then $y_1 = b_1 x_2$, $y_1 = b_2 x_1$, and $y_1 = c, c \geq 0$. Thus, $x_1 = c/b_2$, $x_2 = c/b_1$, and $r = c(b_2 - b_1)/(b_1 b_2)$. Then $x_0 = c(b_1 + b_2)/(2b_1 b_2)$ and $y_0 = c(2b_1 b_2 + b_2 - b_1)/(2b_1 b_2)$. Thus, $y_0 = Bx_0$ where $B = \frac{b_2 - b_1 + 2b_1 b_2}{b_2 + b_1}$. Similarly for $\mathcal{D}$ touching l_1, l_2 with its upper horizontal side. If $\mathcal{D}$ touches l_1, l_2 with its left vertical side then $y_1 = b_2 x_1$, $y_2 = b_1 x_1$, and $x_1 = c, c \geq 0$. Then $r = c(b_1 - b_2), x_0 = c(b_1 - b_2 + 2)/2$, and $y_0 = c(b_1 + b_2)/2$. Thus, $y_0 = Bx_0$ with $B = \frac{b_1 + b_2}{b_1 - b_2 + 2}$.

If $0 \leq b_1 < b_2$, $\mathcal{D}$ must touch l_1, l_2 with two non-adjacent corners or $\mathcal{D}$ touches the intersection point of l_1, l_2 with exactly one corner. In the latter case the bisector is clearly a (-1)-slope line. If $\mathcal{D}$ touches l_1, l_2 with its lower-right and upper-left corner respectively, then $y_1 = b_1 x_2$, $y_2 = b_2 x_1$, $y_1 + x_2 = c$, and $x_1 + y_2 = c, c \geq 0$. Thus, $x_2 = c/(b_1 + 1), x_1 = c/(b_2 + 1), y_2 = cb_2/(b_2 + 1)$ and $y_1 = cb_1/(b_1 + 1)$. Then $x_0 = c(b_1 + b_2 + 2)/(2(b_1 + 1)(b_2 + 1))$ and $y_0 = c(b_1 + b_2 + 2b_1 b_2)/(2(b_1 + 1)(b_2 + 1))$. Thus, $y_0 = Bx_0$, $B = \frac{b_1 + b_2 + 2b_1 b_2}{b_1 + b_2 + 2}$.

Similarly for $b_2 < b_1 \leq 0$ and for a vertical l_2. The case of parallel lines is easy to see. $\square$

The L_∞ bisector of a point p and a line l ($p \notin l$, see Fig. 6) consists of at most four parts, one for each quadrant of p. Each part corresponds to the portion of the bisector between l and a vertical or horizontal line through p depending on the quadrant of p. Thus, each part is a line segment or ray whose slopes can be derived by Lemma 2. The unbounded parts of the bisector are always 45° rays (see Fig. 6). For a point p along a non-axis-parallel line l the bisector is a 45° line through p and for a point p along an axis-parallel line l the bisector is the area enclosed by the 45° lines through p. For simplicity we only consider the boundary of this region as the bisector between p and l; the interior of this region is considered to be closer to p.

Let G be a *planar straight line graph* on n points in the plane as defined in Refs. [1,3] i.e., a set of non-crossing line segments spanned by these points. The

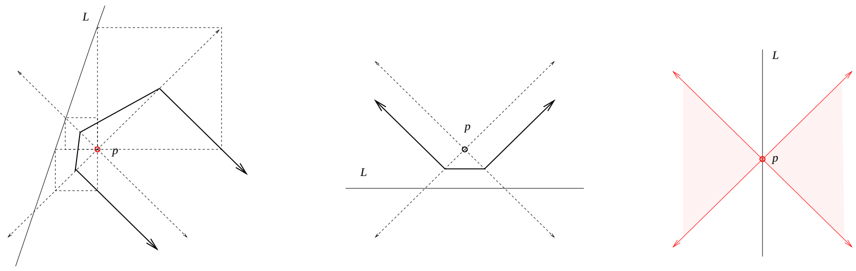

Fig. 6. The L_∞ bisector between a point and a line.

elements of G, $\{e_1, e_2, \ldots, e_n\}$, consist of the points and the open portions of the segments of G. The L_∞ Voronoi diagram of G, denoted $\mathcal{V}(G)$, is a partitioning of the plane into regions, called *Voronoi cells*, each of which is associated with an element of G, called the *owner*, of the cell. The boundary that borders two Voronoi cells is called a *Voronoi edge*, and adjacent Voronoi edges of each Voronoi cell are common to a *Voronoi vertex*. The collection of Voronoi edges and vertices is also called the Voronoi diagram. By the above discussion it is clear that the L_∞ Voronoi diagram of a planar straight-line graph G consists of straight line edges. Moreover, if the vertices of G are on rational coordinates the L_∞ Voronoi vertices are also rational. Fig. 7 illustrates the L_∞ Voronoi diagram in the interior of a simple polygon (the *medial axis*) and Fig. 8 illustrates the L_∞ Voronoi diagram of polygons. Most of the existing algorithms to compute the Euclidean Voronoi diagram of a planar straight line graph can be easily modified to compute the L_∞ Voronoi diagram in $O(n \log n)$ time. The well known *incircle test* to determine whether an element is in or out of the circle defined by three other elements (lines or points) now simplifies to a test involving the square defined by the three elements[c]. The details of the L_∞ incircle test and the precision required are given in Section 4. The following lemma is easy to see.

Lemma 3 *Three lines define a square if and only if two of the lines have non-negative (resp. non-positive) slope and one has non-positive (resp. non-negative) slope.*

The combinatorial complexity of the L_∞ Voronoi diagram is similar to the Euclidean one. However, there is some difference in the number of Voronoi vertices produced by reflex angles. The number of 45° rays emanating from a concave vertex v depends on the slopes of the incident edges and can be either one, two, or three. In particular, if the incident edges belong in the same, consecutive, or non-consecutive quadrant of v, there are respectively three, two, or one 45° rays emanating from v. In the Euclidean case the *normals* bisecting a vertex from the incident edges are always two and in the *straight-skeleton*[1] the corresponding angular bisector is always one. Thus, the combinatorial complexity of the L_∞ diagram can be lower or higher than the Euclidean one depending on the input. An exact bound that

[c]Three elements are said to define a square if there exists a square touching the three elements. Three elements need not always define a square.

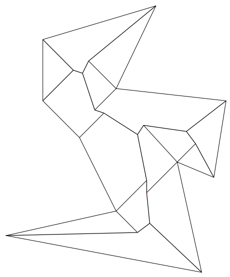

Fig. 7. The L_∞ medial axis of a simple polygon.

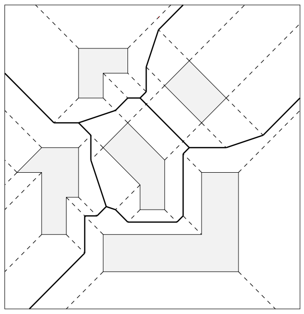

Fig. 8. The L_∞ Voronoi diagram of polygons.

also counts the "infinite" vertices on unbounded edges and 45° rays, can be derived following the proof of Ref. [3] for the Euclidean case.

Lemma 4 *Let G be a planar straight line graph on n points in the plane such that G has t terminals (vertices of degree 1), t_n of which are incident to non-orthogonal edges, r_2 reflex angles inducing two* 45° *rays, and r_3 reflex angles inducing three* 45° *rays. The number of (finite and infinite) vertices of the L_∞ Voronoi diagram of G is exactly $2n + t + t_n + r_2 + 2r_3 - 2$.*

Proof. Assuming G in general position, all Voronoi vertices are induced by a triplet of elements. Some Voronoi vertices are induced by a triplet including a segment and its own endpoint, referred to as *type-2* vertices, and the rest are induced by a triplet of distinct elements, referred to as *type-3* vertices (terminology follows Ref. [3]). A *Type-2* vertex is always incident to a 45° ray emanating from a point of G. In the diagram obtained by removing all 45° rays incident to the points of G, *type-2* vertices have degree 2 and *type-3* vertices have degree three. Type-3 vertices are induced in exactly the same way as in the Euclidean case, thus their number remains the same. The difference with the Euclidean case are vertices of type-2.

Let's first count type-2 vertices. Type-2 vertices are induced by the 45° rays emanating from terminals and reflex angles. Let r denote the total number of reflex angles of G and let r_1 denote those inducing only one 45° ray. Non-orthogonal terminals induce three 45° rays i.e., $3t_n$ 45° rays in total. Orthogonal terminals induce two 45° rays i.e., $2(t - t_n)$. Similarly, reflex angles induce $(r_1 + 2r_2 + 3r_3)$

$45°$ rays. i.e, $r + r_2 + 2r_3$. Each $45°$ ray has exactly one type-2 vertex as endpoint (finite or infinite). Thus the total number of type-2 vertices is $(2t+t_n+r+r_2+2r_3)$.

To derive the number of type-3 vertices we restrict the proof of Ref. [3] to count only type-3 vertices. This number is the same in both the Euclidean and the L_∞ case. If G consists of e disjoint line segments then the number of type-3 vertices (including those at infinity) is $2e-2$ (see Ref. [3] page 44). Now let's transform G to a set of disjoint segments by shortening each segment slightly such that the segment endpoints are in general position.[3] An endpoint p of degree $d \geq 2$ in G gives rise to d copies in the transformation. The Voronoi diagram of these copies has $d-2$ finite *type-3* vertices due to these copies. Since the sum of degrees $d \geq 2$ in G is $2e-t$, we get $(2e-t)-2(n-t)$ new *type-3* vertices because of the copies. A reflex angle at p gives rise to one new *type-3* vertex (finite or infinite) on the bisector of the copies of p. A convex angle at p does not produce new type-3 vertices Thus, the number of new type-3 vertices due to the transformation is $(2e-2n+t+r)$. Hence, the total number of *type-3* vertices is $(2e-2)-(2e-2n+t+r) = 2n-r-t-2$. The lemma follows by summing up type-2 and type-3 vertices. □

Note that in the Euclidean case this bound is $2n+t+r-2$, where r is the total number of reflex vertices[3] (i.e., $r = r_1 + r_2 + r_3$, where r_1 is the number of reflex vertices inducing a single $45°$ ray). In the rectilinear case the bound becomes $2n+t-2$.

3. A Sweep-Line Algorithm for the L_∞ Voronoi Diagram

In Ref. [9], Dehne and Klein presented a plane sweep paradigm for the Voronoi diagram of points in "nice metrics". In the Euclidean case it is a refinement of the original plane sweep approach given by Fortune.[12] In this section we modify the algorithm of Ref. [9] to accommodate segments in the L_∞ metric.

We will compute the L_∞ Voronoi diagram of a planar straight-line graph G, $\mathcal{V}(G)$. Consider a vertical sweep-line L sweeping across the entire plane from left to right. At any instant t of the sweeping process the sweep-line partitions the set of segments (edges) of G into three subsets S_l, S_m and S_r, corresponding to those that lie totally to the left of L, intersect L, and lie totally to the right of L, respectively. The segments in S_m cut L into $|S_m|+1$ sweepline segments, denoted as L_t, two of which are unbounded. Let S_t denote the portions of segments in S_m to the left of L. In other words, $S_t = \{s_t \mid s \in S_m\}$ where s_t denotes the portion of s to the left of L. At every instant t of the sweeping process we compute the Voronoi diagram of $G_t = S_l \cup S_t \cup L_t$. Note that at any instant t the sweep-line segments are treated as being part of G. (In case G is a collection of simple polygons and we only wish to compute the Voronoi diagram in the exterior of the polygons, we exclude from L_t the segments in the interior of polygons, Fig. 9).

The boundary of the Voronoi cell of each segment in L_t is referred to as a *wavefront component* and the collection of all wavefront components for all segments in L_t is called the *wavefront*. In Fig. 9, the wavefront is shown in dashed lines. Note that in the case of points the whole wavefront consists of a single component. Clearly, in the L_∞ metric the wavefront is y-monotone. The elements of G

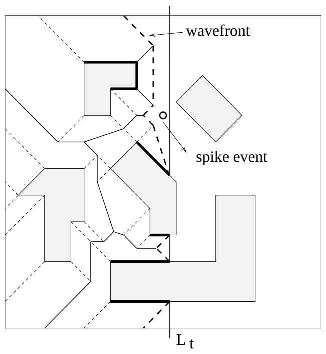

Fig. 9. The L_∞ Voronoi diagram of G_t.

that contribute segments (bisectors) to the wavefront are referred to as *wavefront elements*. In Fig. 9, the wavefront-elements are shown in thick lines. The Voronoi edges (bisectors) that have an endpoint common with the wavefront are called *spike bisectors*. A segment of the wavefront is called a *wave* and has as owner a wavefront element.

As the plane sweep proceeds, the wavefront, and therefore the endpoints of spike bisectors, as well as the endpoints of segments in S_t move continuously to the right until an *event* takes place which causes the wavefront to change. Following Ref. [9], we have two kinds of events: 1) events corresponding to the occurrence of a point or a vertical edge in G, referred to as *site events* and 2) events corresponding to the intersection point of two neighboring spike bisectors, referred to as *spike events*. A site event induced by a point p takes place at $t = x_p$, where x_p is the abscissa of point p. A spike event C takes place at $t = x_c + w$ where x_c is the abscissa of the intersection point and w is the distance of the intersection from the inducing element i.e., as soon as the intersection is reached by the wavefront. Throughout the plane sweep we implicitly maintain the wavefront in a *height-balanced* binary tree, $\mathcal{T}$, referred to as the *sweep-line status*. In particular, the sweep-line status contains the waves ordered according to their order in the wavefront. Each wave corresponds to a wavefront element. Note that a wavefront element may introduce more than one wave. Each wave is *sliding* along two *tracks* which are either spike bisectors or segments in S_m. A segment in S_m is the owner of the incident waves. At any instant t of the sweeping process, the endpoint of a spike bisector corresponds to the center of the square defined by the owners of the bisector and sweep-line L_t. The endpoint of a segment in S_m corresponds to the intersection of the segment with the sweep-line L_t. At any instant t, the endpoints of a wave are given by the endpoints of the incident spike bisector(s) or segment in S_m. The wavefront is the polygonal line obtained by connecting the endpoints of consecutive waves in the order they appear in $\mathcal{T}$. Since the wavefront is y-monotone it coincides with the polygonal line obtained by connecting the endpoints of spike bisectors and of segments in S_m in increasing y-coordinate value. The events are organized in a priority queue referred to as the *event list* in increasing priority value.

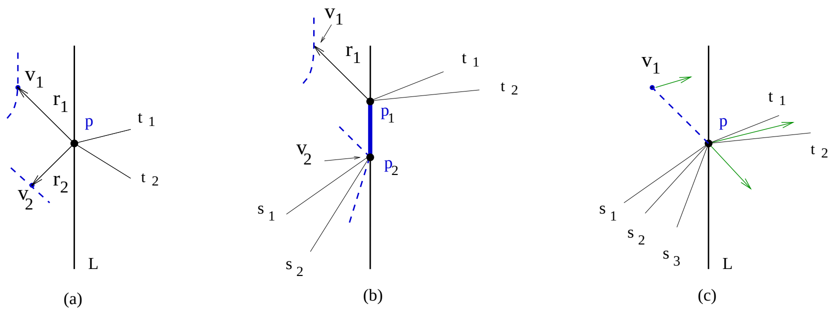

Fig. 10. Handling site events.

Now let us discuss how to handle the event points. Let P be a site event corresponding to a point p or to a vertical segment $\overline{p_1p_2}$ in G (p_1 is assumed to be above p_2). For simplicity of presentation, a point p is treated as a vertical segment $\overline{p_1p_2}$ of zero length i.e., $P = p = p_1 = p_2$. For a point p_i, let $S(p_i) = \{s_1(p_i), \ldots, s_k(p_i)\}$ and $T(p_i) = \{t_1(p_i), \ldots, t_m(p_i)\}$ denote the segments of G incident to point p_i to the left and to the right of L_t respectively. $S(p_i)$ and $T(p_i)$ are ordered from top to bottom (see Fig. 10) and are referred to as the set of *incoming* and *outgoing* segments of p_i respectively. The set of incoming and outgoing segments of P are $S(P) = S(p_1) \cup S(p_2)$ and $T(P) = T(p_1) \cup \overline{p_1p_2} \cup T(p_2)$ respectively. (If P is a point, $T(P) = T(p_1) = T(p_2)$). The vertical and the horizontal axes through p_i partition the plane into four quadrants. Recall that if the segments incident to p_i form a reflex angle, there must be $45°$ ray(s) emanating from p_i depending on which quadrant of p_i the incident segments lie. We do the following:

(i) Determine the portion of the wavefront that becomes part of the final Voronoi diagram. In other words, determine points v_1 and v_2 on the wavefront such that the part of the wavefront between v_1 and v_2 is finalized. (v_1 is assumed to be *above* v_2, i.e., v_2 follows v_1 as we walk along the wavefront from top to bottom).

  (a) Determine v_1 as follows:
  If the upper-left quadrant of p_1 has no incoming segments (i.e., $S(p_1) = \emptyset$ or $s_1(p_1)$ has positive slope), consider a ray r_1 of slope -1 emanating from p_1 extending to the left of L_t (see Fig. 10). If $S(p_1) \neq \emptyset$, ray r_1 is part of the wavefront itself, thus let v_1 be the other endpoint of r_1 (Fig. 10c). If $S(p_1) = \emptyset$, let v_1 be the intersection point of r_1 with the wavefront (Fig. 10a,b). To compute vertex v_1 in the latter case we perform binary search in $\mathcal{T}$ to identify the wave intersected by r_1.
  The binary search proceeds as follows: Let e_i and e_{i+1} denote two consecutive wavefront elements in $\mathcal{T}$ and let $b(e_i, e_{i+1})$ be their spike bisector (the wave of e_i is assumed below the wave of e_{i+1}). Consider the square $\mathcal{D}_i$ defined by e_i, e_{i+1} and the sweepline $\mathcal{L}_t$. Let Y_i denote the lower ordinate of $\mathcal{D}_i$. If p_1 lies below Y_i, we can exclude from the search all

wavefront elements above and including e_{i+1}. (Note that r_1 must intersect the wavefront below the endpoint of $b(e_i, e_{i+1})$.) If p_1 lies above Y_i then we can exclude all wavefront elements below e_i (not including e_i). If the ordinate of p_1 is identical to Y_i then v_1 is the endpoint of $b(e_i, e_{i+1})$. In the latter case we have four "co-circular" elements (in the L_∞ sense). For a wavefront element e_i corresponding to a segment in S_m the decision is easier: If p_1 lies below (resp. above) e_i we can exclude all wavefront elements above (resp. below) e_i.
If the upper-left quadrant of p_1 contains incoming segments (i.e., $s_1(p_1)$ has non-positive slope) let $v_1 = p_1$.

(b) Determine v_2 similarly:
If the lower-left quadrant of p_2 has no incoming segments (i.e., $S(p_2) = \emptyset$ or $s_1(p_2)$ has negative slope), consider a ray r_2 of slope $+1$ emanating from p_2 extending to the left of L_t. If $S(p_2) \neq \emptyset$, r_2 is part of the wavefront; let v_2 be the other endpoint of r_2. If $S(p_2) = \emptyset$, let v_2 be the point of intersection of r_2 with the wavefront; v_2 can be identified by binary search similarly to step (a). If the lower-left quadrant of p_2 contains incoming segments (i.e., $s_k(p_2)$ has non-negative slope) let $v_2 = p_2$.

(ii) Update the Voronoi diagram so far by inserting the portion of the wavefront identified in step (i).

(iii) Delete from the sweep line status $\mathcal{T}$ all waves between v_1 and v_2 (except those containing v_1 and v_2).

(iv) Generate new spike bisectors: (In Fig. 10c, new spike bisectors are illustrated in arrows):

(a) For every pair of consecutive segments in $T(p_1)$ and $T(p_2)$ (if any) generate a new spike bisector.

(b) If the upper-right quadrant of p_1 (resp. lower-right quadrant of p_2) contains no outgoing segments i.e., $T(p_1) = \emptyset$ or $t_1(p_1)$ has negative slope (resp. $T(p_2) = \emptyset$ or $t_m(p_2)$ has positive slope), generate a new spike bisector as a ray of slope $+1$ (resp. -1) emanating from p_1 (resp. p_2) extending to the right of L_t.

(c) Generate a new spike bisector for every Voronoi vertex $v_i, i = 1, 2$, produced in step (i) (if any). Let $e_i, i = 1, 2$ denote the element of G inducing the wave incident to v_i. Let $b_i, i = 1, 2$ denote the new spike bisector with startpoint at v_i. b_i is defined by e_i and an outgoing segment t_i as follows: If the upper-right quadrant of p_1 (resp. lower-right quadrant of p_2) contains outgoing segments, let $t_1 = t_1(p_1)$ (resp. $t_2 = t_m(p_2)$). (t_1 has positive slope and t_2 has negative slope in this case). Otherwise, let t_i denote the horizontal line through p_i.

(d) If both upper (resp. lower) quadrants of p_1 (resp. p_2) contain segments generate a new spike bisector b_1 (resp. b_2) with startpoint at p_1 (resp. p_2), extending above p_1 (resp. below p_2), defined by the lines through $s_1(p_1)$ and $t_1(p_1)$ (resp. $s_k(p_2)$ and $t_m(p_2)$).

Note that b_i may extend to the right or to the left of L_t and that it may be a 45° ray.

At any instant t of the sweeping process the endpoint of a spike bisector corresponds to the center of a square defined by L_t and the two wavefront elements inducing the bisector.

(v) Update the sweep-line status $\mathcal{T}$ by inserting waves corresponding to new wavefront elements sliding along the new spike bisectors generated in step (iv) and the outgoing segments in $T(P)$. Each wave corresponds to a wavefront element. Clearly, all elements in $T(P)$ define new wavefront elements. If both upper (resp. lower) quadrants of p_1 (resp. p_2) contain no segments, then p_1 (resp. p_2) forms a new wavefront element equivalent to a horizontal line through p_1 (resp. p_2). Note that in this case, a spike bisector corresponding to a 45° ray emanating from p_i must have been generated in step (iv). If $T(P) = \emptyset$ then P itself induces a new wave equivalent to the one of a vertical line. In this case two 45° rays emanating from p_i must have been generated in step (iv).

The insertion order of the new wavefront elements in $\mathcal{T}$ is the natural order of the wavefront. Note that an element may appear more than once in $\mathcal{T}$ since it may define multiple waves. All new waves are inserted between the waves of the elements inducing v_1 and v_2.

(vi) For every new spike bisector b generated in step (iv) create at most one new spike event corresponding to the first intersection of b (if any) with the neighboring spike bisectors from above and below. (The order is induced by the wavefront). If b is induced by two consecutive segments in $T(p_i)$, there are no neighboring spike bisectors and thus no spike event gets generated. A spike event induced by the intersection of two neighboring spike bisectors corresponds to the center of the square defined by the three wavefront elements inducing the spike bisectors. The priority value of the spike event is given by the rightmost abscissa of this square. Insert those spike events in the event-list.

Spike events are handled similarly. Let q be a spike event induced by the intersection of two spike bisectors i.e., q is induced by three elements e_1, e_2, e_3. If q is *valid*, the wavefront elements must be neighboring in $\mathcal{T}$; otherwise q is *invalid*. Invalid spike events are simply discarded. A valid spike event q is a Voronoi vertex of the final diagram and the incident spike bisectors are Voronoi edges. The wave of the middle wavefront element vanishes at q and gets deleted from $\mathcal{T}$. A new spike bisector b induced by e_1 and e_3, emanating from q, is generated. At most two new spike events are generated due the intersections of b with its upper and lower neighbors. These spike events correspond to the center of the squares defined by e_1, e_3 and the wavefront element preceding e_1 and following e_3 respectively. The priority value of the spike events is given by the rightmost abscissa of the squares. The spike event of minimum priority value is inserted in the event-list; the second spike event must be invalid. Note that in case of more than three "co-circular" elements the new spike event may coincide with q.

The correctness of the algorithm follows using the same arguments as in Ref. [9]. By the same observations the number of site and spike events is $O(n)$. Thus, the time complexity of the algorithm is $O(n \log n)$.

4. The degree of the plane sweep algorithm

Let's first review some definitions and notation from Refs. [15,5]. An algorithm has *degree d* if its test computations (*predicates*) involve the evaluation of multivariate polynomials of arithmetic degree at most d Ref. [15]. The arithmetic degree of a polynomial is the maximum arithmetic degree of its monomials, and the arithmetic degree of a monomial is the sum of the arithmetic degrees of its *variables*. An input variable is considered to have degree 1; the arithmetic degree of an arbitrary variable is the degree of the polynomial that computes it. An *elementary predicate* is the sign of a homogeneous multivariate polynomial of input variables and more generally a *predicate* is a boolean function of elementary predicates.[5] The degree of an elementary predicate is the maximum degree of its irreducible (over the rationals) factors whose sign is not constant. Clearly the degree of a predicate is the maximum degree of its elementary predicates and the degree of an algorithm is the maximum degree of its predicates. For more information on the degree of an algorithm and its implication on the speed and robustness of the algorithm see Ref. [5]. All input data are considered to be b-bit integers.

Following the notation of Ref. [15], an unspecified multivariate polynomial of degree s over input variables is denoted by α^s. A specific term ρ, which is known to be a polynomial of degree s over input variables, can be written as α^s using the *genericization* operation ($\rho \to \alpha^s$) Ref. [15]. To manipulate unspecified polynomials the following rules were given in Ref. [15]: $\alpha^s\alpha^r \to \alpha^{s+r}$ (product rule), $\alpha^s + \alpha^s \to \alpha^s$ (sum rule), $-\alpha^s \to \alpha^s$ (sign rule). In the following we will show that the degree of the plane sweep algorithm in Section 3 is seven.

Let's consider a spike event C induced by three wavefront elements e_i, e_j, e_k appearing on the wavefront from bottom to top. C corresponds to the intersection point of $b(e_i, e_j)$ and $b(e_j, e_k)$ which is the center of a square $\mathcal{D}(e_i, e_j, e_k)$ (for brevity $\mathcal{D}$) touching e_i, e_j and e_k. The priority of C is the abscissae of the rightmost side of $\mathcal{D}$. Without loss of generality we assume that e_i, e_j, e_k correspond to three lines, denoted by l_i, l_j, l_k, respectively. Note that in L_∞, any point can be regarded as a vertical or horizontal line depending on which quadrant of the point is relevant. Let's also assume that the slope of l_i is non-negative; the case where l_i has non-positive slope is equivalent. By Lemma 3 one of the slopes must be of opposite sign than the others. Thus, assuming that the slope of l_i is non-negative, the slopes of l_j, l_k must be non-positive and non-negative respectively.

Let $X_1(e_i, e_j, e_k), X_2(e_i, e_j, e_k), Y_1(e_i, e_j, e_k)$ and $Y_2(e_i, e_j, e_k)$ (for brevity X_1, X_2, Y_1, Y_2) denote the x and y coordinates of $\mathcal{D}(e_i, e_j, e_k)$ respectively, where $X_1 < X_2$ and $Y_1 < Y_2$ (see Fig. 11). Under our assumptions, points (X_1, Y_2), (X_1, Y_1), and (X_2, Y_1) must belong in l_k, l_j and l_i respectively. Let $a_m x + b_m y + c_m = 0$ be the line equation of line $l_m, m = i, j, k$. To determine $\mathcal{D}$ we need to solve the following linear system of four equations:

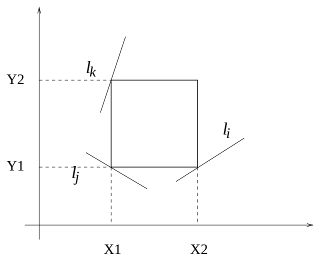

Fig. 11. The square defined by three lines

$$A\mathbf{z} = \mathbf{c}$$

$$A = \begin{pmatrix} 0 & a_i & b_i & 0 \\ a_j & 0 & b_j & 0 \\ a_k & 0 & 0 & b_k \\ -1 & 1 & 1 & -1 \end{pmatrix}, \ \mathbf{z}^T = (X_1, X_2, Y_1, Y_2), \ \mathbf{c}^T = (-c_i, -c_j, -c_k, 0)$$

Thus, we obtain the following formulas for the coordinates of $\mathcal{D}$.

$$\begin{aligned} X_1 &= \frac{N_1}{D}, \ N_1 = -b_j b_k c_i - a_i b_k c_j + b_i b_k c_j + a_i b_j c_k && (1) \\ D &= -a_i a_k b_j + a_i a_j b_k - a_j b_i b_k + a_i b_j b_k \\ X_2 &= \frac{N_2}{D}, \ N_2 = a_k b_j c_i - a_j b_k c_i - b_j b_k c_i - a_k b_i c_j + b_i b_k c_j + a_j b_i c_k && (2) \\ Y_1 &= \frac{M_1}{D}, \ M_1 = a_j b_k c_i + a_i a_k c_j - a_i b_k c_j - a_i a_j c_k && (3) \\ Y_2 &= \frac{M_2}{D}, \ M_2 = a_k b_j c_i + a_i a_k c_j - a_k b_i c_j - a_i a_j c_k + a_j b_i c_k - a_i b_j c_k && (4) \end{aligned}$$

The priority value of spike event C is X_2 and the Voronoi vertex induced by C is the center of $\mathcal{D}$, point $(X_1 + r, Y_1 + r), r = (X_2 - X_1)/2$.

To implement the plane sweep algorithm robustly, it is essential to correctly compute at any instant the event of lowest priority value. For this purpose we need to compare the priority values of spike and site events accurately. Furthermore, we need the ability to correctly perform the binary search described at step (i) of the algorithm. These comparisons correspond to correctly answering the predicates given below. These are the only predicates that are used in the algorithm. We use the following notation: For two points p, q, $p <_x q$ means that the x-coordinate of p is less than the x-coordinate of q. Similarly for $<_y$. Given a point p and a line l, $p <_y l$ means that point p lies below line l, i.e., $p <_y l_y$, where l_y is the intersection of l and the vertical line that passes through p.

Predicate 1: $p <_y l$

Predicate 2: $p <_y Y_1(L_t, e_j, e_k)$ or $p <_y Y_2(L_t, e_j, e_k)$

Predicate 3: $X_2(e_i, e_j, e_k) <_x p$: Compare the priority value of a spike and a site event.

Predicate 4: $X_2(e_1, e_2, e_3) <_x X_2(e_4, e_5, e_6)$: Compare the priority values of two arbitrary spike events.

Here p denotes a point site of the straight-line graph G. A line l: $ax+by+c=0$, is defined by two points $p=(x_p, y_p)$ and $q=(x_q, y_q)$ where $a=y_p-y_q$, $b=x_q-x_p$, and $c=-y_p x_q + x_p y_q$. Using the genericization operation of Ref. [15] $a, b \to \alpha$ and $c \to \alpha^2$.

Lemma 5 *The predicates of the plane sweep algorithm can be answered with the following degrees: Predicate 1: degree* 2*; Predicate 2: degree* 3*; Predicate 3: degree* 4*; Predicate 4: degree* 7*.*

Proof. The degree of Predicate 1 is 2 as shown in Ref. [5].

As mentioned above, we assume that the slope of e_i is non-negative and thus the slopes of e_j and e_k are non-positive and non-negative respectively. For Predicate 2, we use the equation of the sweep-line L_t (i.e., $a_i=1, b_i=0, c_i=-t$) in equation (3). Then $Y_1(L_t, e_j, e_k) = M_1(L_t, e_j, e_k)/D(L_t, e_j, e_k)$, where $M_1(L_t, e_j, e_k) = -a_j b_k t + a_k c_j - b_k c_j - a_j c_k \to \alpha^3$ and $D(L_t, e_j, e_k) = -a_k b_j + a_j b_k + b_j b_k \to \alpha^2$. Thus Predicate 2, $P_2 = y_p * D(L_t, e_j, e_k) - M_1(L_t, e_j, e_k) < 0$, and $P_2 \to \alpha^3$. Similarly for $p <_y Y_2(L_t, e_j, e_k)$.

Using Eq.(4), Predicate 3 becomes $P_3 = N_2(e_i, e_j, e_k) - D(e_i, e_j, e_k) * x_p < 0$, where $N_2(e_i, e_j, e_k) \to \alpha^4$ and $D(e_i, e_j, e_k) \to \alpha^3$. Thus, $P_3 \to \alpha^4 - \alpha^3\alpha \to \alpha^4$.

The most expensive predicate in terms of degree is Predicate 4: Given two arbitrary spike events determine which one has lowest priority value. Using Eq.(2) this predicate can be written as $X_2(e_1, e_2, e_3) < X_2(e_4, e_5, e_6) \Rightarrow N_2(e_1, e_2, e_3)D(e_4, e_5, e_6) - N_2(e_4, e_5, e_6)D(e_1, e_2, e_3) < 0$. $N_2(e_i, e_j, e_k) \to \alpha^4$, $D(e_i, e_j, e_k) \to \alpha^3, i = 1,4, j=2,5, k=3,6$. Thus, $P_4 = N_2(e_1, e_2, e_3)D(e_4, e_5, e_6) - N_2(e_4, e_5, e_6)D(e_1, e_2, e_3) \to \alpha^7$. □

Theorem 1 *Given a planar straight line graph G on n points in the plane there exists a sweep-line algorithm of degree 7 to compute the L_∞ Voronoi diagram of G in $O(n \log n)$ time.*

Proof. The $O(n \log n)$-time complexity of the algorithm was established in Section 3. By Lemma 5 the degree of the algorithm is at most 7. To show that it is exactly 7 we need to show that the polynomial of Predicate 4 is irreducible over the rationals. This is shown in Lemma A.2 in the appendix. □

In the case of segments in fixed orientations where the slopes of segments can be assumed to be small constants the degree of the algorithm is only 1. This situation is very common in VLSI layouts and thus this algorithm can be directly implemented in this case with no robustness problems.

The L_∞ Voronoi diagram of segments could also be computed in an incremental fashion where the most expensive predicate to be answered correctly is the L_∞ *in-circle* test. An algorithm based on the in-circle test (e.g., similar to those presented in Ref. [21,6] for the Euclidean case) would have lower degree than the plane sweep since the L_∞ in-circle test for line segments can be answered with degree 5 (see Lemma 6). In this paper we chose the plane sweep approach because it is simple and very appropriate for our application presented in the next section. In this application there is no need to maintain the whole Voronoi diagram throughout the computation; it is enough to only maintain Voronoi cells incident to the wavefront. Since the amount of data in modern VLSI designs is extremely large, given in a compact hierarchical manner, the plane sweep approach results in much lower memory requirement. Moreover, as mentioned above, typical VLSI designs contain shapes in a small number of constant orientations (often ortho-45) in which case the plane sweep is robust.

Lemma 6 *The L_∞* in-circle *test for line segments can be answered with degree 5.*

Proof. Let $l_i, i = 1,2,3$ be three lines defining a square $\mathcal{D}$. (Note that three lines may define more than one square but the test is performed for the one corresponding to a given Voronoi vertex). Let s denote the query segment and let's assume that s is not orthogonal (otherwise the test is easier).

We first perform the in-circle test for the endpoints of S; if an endpoint lies within $\mathcal{D}$ then clearly s intersects the interior of $\mathcal{D}$. Otherwise, let l denote the line through s. The test is the following: If the slope of l is negative (resp. positive) check whether the corners (X_1, Y_1) and (X_2, Y_2) (resp. (X_1, Y_2) and (X_2, Y_1)) lie on opposite sides of l. Thus, we need to answer the following predicate where l is $ax+by+c=0$, $a, b \rightarrow \alpha$, and $c \rightarrow \alpha^2$: $P_1 = aX_1+bY_1+c < 0 \wedge P_2 = aX_2+bY_2+c > 0$ (if $-a/b < 0$), or $P_1 = aX_1 + bY_2 + c < 0 \wedge P_2 = aX_2 + bY_1 + c > 0$ (if $-a/b > 0$). This is equivalent to $P_1 = aN_1 + bM_1 + cD < 0 \wedge P_2 = aN_2 + bM_2 + cD > 0$ or $P_1 = aN_1 + bM_2 + cD < 0 \wedge P_2 = aN_2 + bM_1 + cD > 0$. Thus, $P_1 \rightarrow \alpha^5, P_2 \rightarrow \alpha^5$. □

In the Euclidean metric, the in-circle test for points and segments can be answered with degree 4 and 40 respectively.[6] However, the plane sweep approach for the Euclidean Voronoi diagram of points and segments must have higher degree. The main reason is the need to accurately compare the priority value of two arbitrary spike events i.e., the predicate equivalent to our Predicate 4. As the following lemma shows, in the case of points in the Euclidean metric this predicate (denoted as P_4') can be answered with degree 20. Note that the same predicate in the L_∞ metric has degree 1. For segments predicate P_4' must have higher degree.

Lemma 7 *Predicate P_4' for points in the Euclidean metric can be answered with degree 20.*

Proof. The equation of the circle through three non-collinear points $(x_i, y_i), i = 1,2,3$, is given by the following determinant.

$$\begin{vmatrix} x^2+y^2 & x & y & 1 \\ x_1^2+y_1^2 & x_1 & y_1 & 1 \\ x_2^2+y_2^2 & x_2 & y_2 & 1 \\ x_3^2+y_3^2 & x_3 & y_3 & 1 \end{vmatrix} = 0$$

Equivalently, the circle equation is $(x^2+y^2)D - Ax + By - C = 0$, where A, B, C, D are subdeterminants. Note that $A, B \rightarrow \alpha^3$, $C \rightarrow \alpha^4$, and $D \rightarrow \alpha^2$. The center of the circle is (x_0, y_0) where $x_0 = \frac{A}{2D}$ and $y_0 = \frac{-B}{2D}$. The radius of the circle is $R = \frac{1}{2D}(\sqrt{\rho}), \rho = A^2 + B^2 + 4CD$.

Given two distinct triplets of non-collinear points, let $(x_0, y_0), R$, and $(x'_0, y'_0), R'$ denote the center and radius of the circle defined by each triplet. (The factors of the second triplet are denoted by primes). Predicate $P'_4 = x_0 + R < x'_0 + R'$ i.e., $P'_4 = AD' - A'D + \sqrt{\rho}D' - \sqrt{\rho'}D < 0$. Thus, $P'_4 \to \alpha^5 + \alpha^2\sqrt{\rho} - \alpha^2\sqrt{\rho'}$ $\to \alpha^{10} - (\alpha^2\sqrt{\rho} - \alpha^2\sqrt{\rho'})^2 \to \alpha^{10} - \alpha^4\sqrt{\rho}\sqrt{\rho'} \to \alpha^{20} - \alpha^8\rho\rho' \to \alpha^{20}$. (Used the *segregate and square* rule of Ref. [15]). □

5. The L_∞ Voronoi Diagram and the Critical Area Problem

In this section we present an important application of the L_∞ Voronoi diagram of segments in computing the *critical area* for shorts of a VLSI layout. The extraction of critical area forms the main computational bottleneck in predicting the *yield* of a VLSI chip.

Given a circuit layout C, the critical area is defined as $A_c = \int_0^\infty A(r)D(r)dr$ where $A(r)$ denotes the area in which the center of a defect of radius r must fall in order to cause circuit failure and $D(r)$ is the density function of the defect size. Defects are usually modeled as circles and the density function has been estimated to follow the "$1/r^3$" distribution.[11,20,25,27] Existing methods require substantial CPU time for large layouts. They can be summarized into three categories: 1) Geometric methods, where $A(r)$ is computed for several different values of r independently and then the results are used to approximate A_c. The methods to compute $A(r)$ are usually based on *shape-expansion* followed by *shape-intersection* (see Ref. [27] for references). For rectilinear layouts there is a more efficient plane-sweep method.[20] 2) Monte Carlo simulation, where a large number of defects is drawn with their radii distributed according to $D(r)$. The probability of fault is estimated by dividing the number of defects causing faults over the total number of defects.[28] 3) Grid-based approach, where an integer grid is assumed over the layout and the critical radius (i.e., the radius of the smallest defect causing a fault at this point) of every grid point is computed.[27] The time complexity is $O(I^{1.5})$, where I is the number of grid points and can be improved to $O(I)$ as shown in Ref. [19].

For rectilinear layouts, our solution to the critical area problem for shorts, assuming that defects are isothetic squares, appears in Ref. [19]. In Ref. [19], we showed that the critical area integral for rectilinear layouts and layouts including ± 1-slope edges can be written as a function of edges of the *2nd order L_∞ Voronoi diagram* of layout shapes, and thus the critical area computation for shorts can be done analytically once the 2nd order L_∞ Voronoi diagram is available. The *2nd order Voronoi diagram* of arbitrary polygons is a variation of the ordinary k-th order Voronoi diagram of points: The Voronoi cell of a polygon P is subdivided into finer regions by the Voronoi diagram of its neighbors. The 2nd order Voronoi region of an element s (edge or reflex vertex) within the Voronoi cell of P is defined as $reg_P(s) = \{x \mid d(s,x) \leq d(t,x), \forall t \in C - P\}$, where C is the given collection of polygons. The critical radius of every point $x \in reg_P(s)$ is determined by the distance of x from element s; we say that s is the owner of $reg_P(s)$.

Having the critical area computation done based on Euclidean Voronoi diagrams would be out of the question in practice due to the difficulty of computing the

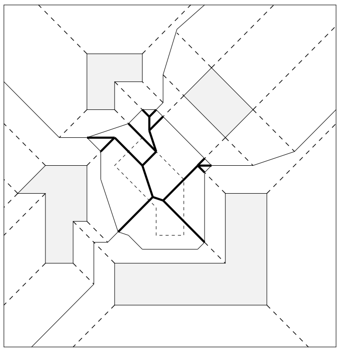

Fig. 12. The 2nd order subdivision within a Voronoi cell.

Voronoi diagram of segments. However, the computation in L_∞ is practical. Using the L_∞ metric is very natural for this problem; it simply amounts to modeling defects as squares instead of circles. In reality spot defects have any kind of shape thus, modeling defects as squares is good enough for all practical purposes. In the critical area literature defects have often been simplified to squares (e.g. Ref. [20]). In this paper we generalize the result of Ref. [19] to general layouts of shapes in arbitrary orientations and show that the critical area formula for shorts (assuming square defects) can be written as a function of 2nd order L_∞ Voronoi edges.

5.1. The Critical Area for Shorts as a Function of Voronoi Edges

We have a layer in a circuit layout consisting of a collection of disjoint simple polygons C in arbitrary orientations. Our goal is to evaluate the integral $A_c = \int_0^\infty A(r)D(r)dr$, where $D(r) = r_0^2/r^3$ and r_0 is a minimum optically resolvable size. Recall that $A(r)$ denotes the area of the *critical region* for square defects of radius r. The critical region for radius r is the locus of points where if the center of a square defect of radius r is placed it causes a short i.e., the defect overlaps with two disjoint polygons.

Consider an arbitrary point p in the layout. The *critical radius* of p for shorts is the radius of the smallest defect which if centered at p overlaps with two different polygons. Such a defect causes a short between the conducting regions represented by the two polygons. Clearly, any larger defect centered at p also causes a short. Thus, the critical radius of p is the distance of p from the second nearest polygon to p. Let's assume that we are given the 2nd order L_∞ Voronoi diagram of C as defined above. Fig. 8 shows the L_∞ Voronoi diagram of a set of shapes and Fig. 12 shows the 2nd order subdivision within a bounded cell. Then $A_c = \sum_V A_c(V)$ for all (2nd order) Voronoi cells V where $A_c(V)$ denotes the critical area within V. Note that $A_c(V) = \int_0^\infty A(r,V)D(r)dr$ where $A(r,V)$ denotes the area of the critical region for defect radius r within V. We will show that the integral can be discretized as a summation of (2nd order) L_∞ Voronoi edges.

Let's first concentrate on a single (2nd order) Voronoi cell V having as owner an

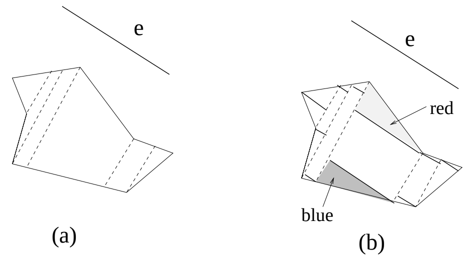

Fig. 13. The decomposition of V into trapezoids.

edge e of slope $\pm s, s \geq 0$. The Voronoi cell of a vertex is considered as the Voronoi cell of a vertical or horizontal edge. Consider a decomposition of V into trapezoids by drawing lines perpendicular to e (lines of slope $\pm 1/s$) emanating from the Voronoi vertices of V (see Fig. 13a). Each trapezoid $\mathcal{T}$ is further decomposed into orthogonal triangles and at most one rectangle $\mathcal{R}$ by drawing lines parallel to e (slope $\pm s$) through its vertices (see Fig. 13b). (In case $\mathcal{T}$ is a slanted parallelogram continue the decomposition recursively). We distinguish between two kinds of triangles, *red* and *blue*, depending on the relevant position of the hypotenuse and the orthogonal apex with respect to e. In particular, if the owner e and the orthogonal apex lie on opposite sides of the hypotenuse the triangle is colored red, otherwise it is colored blue. In Fig. 13b, the lightly shaded triangle is red and the darker shaded one is blue.

Given two vertices v_j and v_k such that v_j is closer to e than v_k, let r_j, r_k denote the corresponding critical radii i.e., the L_∞ distance of v_j and v_k from the line through e respectively. We derive the following formulas for the critical area within an element of the above decomposition.

Lemma 8 *The critical area within a rectangle $\mathcal{R}$, a red triangle T_{red}, and a blue triangle T_{blue} is given by the following formulas, using the "r_0^2/r^3" defect density distribution:*

$$A_c(\mathcal{R}) = \frac{r_0^2 S}{2}\left(\frac{l}{r_j} - \frac{l}{r_k}\right) \tag{5}$$

$$A_c(T_{red}) = \frac{r_0^2 S}{2}\left(ST\ln\left(\frac{r_k}{r_j}\right) - \frac{l}{r_k}\right) \tag{6}$$

$$A_c(T_{blue}) = \frac{r_0^2 S}{2}\left(\frac{l}{r_j} - ST\ln\left(\frac{r_k}{r_j}\right)\right) \tag{7}$$

where l is the size of the edge of $\mathcal{R}, T_{red}$, and T_{blue} parallel to e, $r_k, r_j, r_k > r_j$, are the maximum and the minimum critical radius of their vertices, $S = \frac{s+1}{\sqrt{s^2+1}}$, where s is the absolute value of the slope of e, and $T = \frac{ts+1}{|t-s|}$, where t is the absolute value of the slope of the hypotenuse (i.e, the slope of the corresponding Voronoi edge). For $s = \infty$, $S = 1$ and $T = t$.

Proof. Let's assume, without loss of generality, that the slope of e is positive. Consider a red triangle T_{red} of minimum critical radius r_j. Let a and b denote the

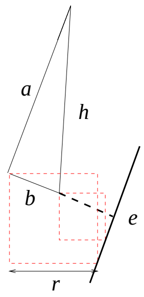

Fig. 14. A red triangle in the Voronoi cell of e

side parallel and perpendicular to e respectively for a defect radius $r, r_j < r \leq r_k$ (see Fig. 14). By Lemma 1, $b = \frac{s+1}{\sqrt{s^2+1}}(r - r_j)$. To determine the length of a let's rotate the coordinate system by an angle ϕ so that a becomes horizontal. $\phi = -\theta$ where $\tan\theta = s$. Let $t = \tan w$ denote the slope of the hypotenuse h (h is a Voronoi edge). In the new coordinate system the slope of h is is $t' = \tan(w - \theta) = \frac{\tan w - tan\theta}{1 + tanw \tan\theta} = \frac{t-s}{1+ts}$. Then $|a| = |b|/t'$.

Let $S = \frac{s+1}{\sqrt{s^2+1}}$ and $T = \frac{1+ts}{|t-s|}$. Then $A(r, T_{red}) = |a||b|/2 = S^2T(r - r_j)^2/2$ for $r < r_k$. For defect radius $r \geq r_k$, the whole triangle is critical and thus, $A(r, T_{red}) = S^2T(r_k - r_j)^2/2$. Hence, the critical area within T is $A_c(T_{red}) = \int_0^\infty A(r, T_{red})D(r)dr = \frac{r_0^2S^2T}{2}(\int_{r_j}^{r_k}(r-r_j)^2/r^3dr + \int_{r_k}^\infty(r_k-r_j)^2/r^3dr) = \frac{r_0^2S^2T}{2}(\int_{r_j}^{r_k}(1/r - 2r_j/r^2 + r_j^2/r^3)dr + (r_k-r_j)^2\int_{r_k}^\infty 1/r^3dr) = \frac{r_0^2S^2T}{2}(\ln(\frac{r_k}{r_j}) + (\frac{2r_j}{r_k} - \frac{2r_j}{r_j}) + (-\frac{r_j^2}{2r_k^2} + \frac{r_j^2}{2r_j^2}) + \frac{(r_k-r_j)^2}{2r_k^2}) = \frac{r_0^2S^2T}{2}(\ln(\frac{r_k}{r_j}) + \frac{2r_j}{r_k} - 2 - \frac{r_j^2}{2r_k^2} + \frac{1}{2} + \frac{(r_k-r_j)^2}{2r_k^2}) = \frac{r_0^2S^2T}{2}(\ln(\frac{r_k}{r_j}) + \frac{r_j}{r_k} - 1)$. But $l = ST(r_k - r_j)$. Thus, $A_c(T_{red}) = \frac{r_0^2S}{2}(ST\ln(\frac{r_k}{r_j}) - \frac{l}{r_k})$.

For a rectangle $\mathcal{R}$, the area of the critical region within $\mathcal{R}$ for defect radius r where $r_j \leq r \leq r_k$ is $A(r, \mathcal{R}) = |b|l = Sl(r - r_j)$. For a defect radius $r \geq r_k$ the whole rectangle $\mathcal{R}$ is critical and thus the area is $A(r, \mathcal{R}) = Sl(r_k - r_j)$. Hence, the critical area within $\mathcal{R}$ is $A_c(\mathcal{R}) = \int_0^\infty A(r, \mathcal{R})D(r)dr = r_0^2Sl(\int_{r_j}^{r_k}\frac{r-r_j}{r^3}dr + \int_{r_k}^\infty\frac{(r_k-r_j)}{r^3}dr) = r_0^2Sl(\int_{r_j}^{r_k}(\frac{1}{r^2} - \frac{r_j}{r^3})dr + \int_{r_k}^\infty\frac{(r_k-r_j)}{r^3}dr) = r_0^2Sl(-\frac{1}{r}\ |_{r_j}^{r_k} + \frac{r_j}{2r^2}\ |_{r_j}^{r_k} - \frac{r_k-r_j}{2r^2}\ |_{r_k}^\infty) = r_0^2Sl(\frac{1}{r_j} - \frac{1}{r_k} + \frac{r_j}{2r_k^2} - \frac{r_j}{2r_j^2} + \frac{r_k-r_j}{2r_k^2}) = r_0^2Sl(\frac{1}{2r_j} - \frac{1}{2r_k}) = \frac{r_0^2S}{2}(\frac{l}{r_j} - \frac{l}{r_k})$.

For a blue triangle T_{blue} consider the red triangle T_{red} obtained by flipping T_{blue} around the hypotenuse. T_{blue} and T_{red} form a rectangle $\mathcal{R}$. For any defect radius $r \geq r_j$ the area of critical region within T_{blue} is $A(r, T_{blue}) = A(r, R) - A(r, T_{red})$. Note that for $r > r_k$ the whole triangle is critical. Thus, $A_c(T_{blue}) = \int_0^\infty(A(r, R) - A(r, T_{red}))D(r)dr = \int_0^\infty A(r, R)D(r)dr - \int_0^\infty A(r, T_{red})D(r)dr$. Thus, $A_c(T_{blue}) = A_c(\mathcal{R}) - A_c(T_{red})$. The formula is derived by arithmetic substitution. $\square$

We can derive the critical area within V by adding up the critical areas within every rectangle and triangle in the above decomposition of V. Because of the summation, terms of the form Sl/r_l corresponding to internal decomposition edges cancel out. Similarly, for logarithmic terms involving endpoints of the decomposition other than Voronoi vertices. Thus, the critical area within V can be written as a function of Voronoi edges. Let's color the Voronoi edges that induce the hypotenuse of red triangles as red, and those inducing the hypotenuse of blue triangles as blue. Voronoi edges that are incident to a rectangle or induce an orthogonal edge of a triangle must be either parallel or perpendicular to the owner of the cell. Those Voronoi edges that are parallel to the owner are referred to as *prime* and they are portions of bisectors of two parallel edges. Prime Voronoi edges are colored red if the interior and the owner of the cell lie on opposite sides of the edge; otherwise they are colored blue. Those Voronoi edges that are perpendicular to the owner are colored *neutral* with respect to that owner. In the formulas of Lemma 8, terms corresponding to red edges get added while terms corresponding to blue edges get subtracted. Neutral edges do not contribute in the formulas. The boundary of the layout is assumed to be a rectangle whose intersection points with the Voronoi diagram are treated as Voronoi vertices. Boundary edges are treated as Voronoi edges but they contribute at most once in the formulas.

We derive the following theorem.

Theorem 2 *Given the 2nd order L_∞ Voronoi diagram of polygons of a layer in a circuit layout C and assuming that defects are squares following the "r_0^2/r^3" defect density distribution, the critical area for shorts in that layer is given by the following formula:*

$$A_c = r_0^2\Big(\sum_{\substack{red,\\ prime\ e_i}} \frac{S_i l_i}{r_i} - \sum_{\substack{blue,\ prime\\ e_m}} \frac{S_i l_i}{r_i} + \frac{1}{2}\sum_{\substack{red, nonprime\\ e_j,\ wrt\ e}} S_e^2 T_e \ln\frac{r_k}{r_j} - \frac{1}{2}\sum_{\substack{blue, nonprime\\ e_j,\ wrt\ e}} S_e^2 T_e \ln\frac{r_k}{r_j} + \frac{B}{2}\Big)$$

where l_i and r_i denote the length and the critical radius of a prime Voronoi edge e_i, and $r_k, r_j, r_k > r_j$ denote the maximum and the minimum critical radius of a non-prime Voronoi edge e_j. The factors S_e and T_e are as defined in Lemma 8 for each owner e of the non-prime edge e_j. B is the sum of the blue terms for prime boundary edges and appears as a correction factor since boundary edges contribute at most once.

Proof. By the above discussion and Lemma 8, it is clear that the critical area within one (2nd order) Voronoi cell V of owner e is given by a summation of terms derived by the bounding Voronoi edges. A prime Voronoi edge e_i contributes $(r_0^2 S_i l_i)/(2r_i)$, and a non-prime, non-neutral Voronoi edge e_j contributes $(r_0^2 S_e^2 T_e \ln\frac{r_k}{r_j})/2$. A neutral Voronoi edge does not contribute to critical area. Every prime Voronoi edge bounds exactly two cells and receives the same color with respect to both cells; thus, it contributes the same term twice. Boundary edges contribute the respective term once. Note that a boundary edge perpendicular to the owner is colored neutral and does not contribute at all to critical area. The formula is derived by adding up the critical area within every Voronoi cell. □

The critical area computation problem for shorts is now reduced to the 2nd order L_∞ Voronoi diagram of polygons. Note that the L_∞ Voronoi diagram of polygons is not necessarily unique; it depends on the convention adapted for breaking ties. However, the critical area result is the same no matter which version of the Voronoi diagram is used: given a point t equidistant from at least two polygonal elements,

$r_c(t)$ is the same no matter which element is considered to be the owner of t. Once the 2nd order L_∞ Voronoi diagram of layout shapes is computed we only need to determine the coloring of Voronoi edges. The coloring of a Voronoi edge can be determined by attributes associated with the Voronoi edge without any need to answer additional high degree predicates. We thus summarize with the following theorem:

Theorem 3 *Assuming square defects following the "r_0^2/r^3" defect density distribution, the critical area integral for shorts on a layer of a VLSI design can be computed accurately by plane sweep in $O(n \log n)$ time and degree 7, where n is the total number of edges of shapes on that layer.*

Proof. It only remains to be shown that the coloring of a Voronoi edge f can be determined with no need for additional predicates of high degree. Let e_i, e_j be the owners of f, where e_i is the owner of the bordering cell under consideration. Let l_i and l_j denote the lines implied by e_i and e_j respectively. Voronoi edge f is portion of the bisector $b(l_i, l_j)$. To determine whether f is prime it is enough to determine whether l_i and l_j are parallel i.e., degree 2 is sufficient. For a prime f, $b(l_i, l_j)$ consists of a single line. For a non-prime f, $b(l_i, l_j)$ is partitioned into four branches by the intersection point of l_i, l_j. Since e_i and e_j do not intersect, f must belong entirely to exactly one branch. Which branch can be easily determined by comparing the sides of l_i (resp. l_j) where e_j (resp. e_i) lie i.e., degree 2 is sufficient. Once the branch of $b(l_i, l_j)$ containing f is determined the color of f can be easily derived by known attributes of f involving which side of f lies l_i compared to the bordering cell of e_i. □

In practice VLSI designs consist typically of segments in fixed orientations with slopes given by small constants. In this case the degree of the algorithm to compute critical area is only 1.

6. Concluding Remarks

We have shown that the L_∞ Voronoi diagram of segments is a simple straight-line skeleton of combinatorial complexity similar to its Euclidean counterpart. It is particularly well suited for applications involving proximity in VLSI layout where shapes are "mostly" rectilinear. In the presence of many orthogonal edges the combinatorial complexity of the L_∞ Voronoi diagram is slightly smaller than the Euclidean one. We have presented a simple plane-sweep algorithm of degree 7 to compute the L_∞ Voronoi diagram of segments in $O(n \log n)$ time. Although the degree is low compared to the Euclidean case, a robust implementation for arbitrary segments would require the use of *arithmetic filters* (see e.g. Refs. [4,7,13]). In the special case of segments in fixed orientations with slopes given by small constants the algorithm is of degree 1 and thus can be implemented robustly by ordinary means. This is typically the case for shapes in VLSI designs. Using the (2nd order) L_∞ Voronoi diagram of shapes in one layer of a VLSI layout we addressed the problem of computing the critical area for shorts. We have also shown that the critical area in one layer of a layout can be written as a function of Voronoi edges and thus it can be computed analytically once the (2nd order) Voronoi diagram is available. In future research we plan to investigate potential uses of the L_∞ Voronoi diagram of segments in other problems in the VLSI layout and manufacturing area.

References

1. O. Aichholzer, F. Aurenhammer, "Straight Skeletons for general Polygonal Figures in the Plane", *Proc. 2nd Int. Computing and Combinatorics Conference,1996* Lecture Notes in Computer Science 1090, 117-126.
2. F. Aurenhammer, "Voronoi diagrams: A survey of a fundamental geometric data structure," *ACM Comput. Survey,* 23 1991, 345-405.
3. F. Aurenhammer and R. Klein, "Voronoi Diagrams" chapter 18, *Textbook on Computational Geometry*, J.Sack and G. Urrutia (eds), Elsevier Science Publishing 2000, 201-290.
4. F.Avnaim, J.D. Boissonnat, O. Devillers, F. Preparata, M. Yvinec, "Evaluating signs of determinants using single precision arithmetic", *Algorithmica*, 17 1997, 11-132.
5. J.D. Boissonnat, F. Preparata, "Robust plane sweep for intersecting segments", *SIAM J. Computing*, to appear.
6. C. Burnikel, "Exact Computation of Voronoi diagrams and Line Segment Intersections", *Ph.D. Dissertation*, Universität des Saarlandes, Saarbrücken Germany, March 1996.
7. C. Burnikel, K, Mehlhorn S. Schirra, "How to compute the Voronoi Diagram of Line Segments: Theoretical and Experimental Results" *Proc. 2nd Annu. European Symp. on Algorithms, 1994*, LNCS 855, 227-239.
8. D. Coppersmith, D.T. Lee and C.K. Wong, "An Elementary Proof of Nonexistence of Isometries Between l_k^p and l_k^q," *IBM J. Research and Development*, Nov. 1979, 696-699.
9. F. Dehne and R. Klein, "The Big Sweep": On the power of the Wavefront Approach to Voronoi Diagrams", *Algorithmica*(1997), 17, 19-32.
10. A.V. Ferris-Prabhu, "Modeling the Critical Area in Yield Forecast", *IEEE J. of Solid State Circuits*, vol. SC-20, No4, Aug. 1985, 874-878
11. A.V. Ferris-Prabhu, "Defect size variations and their effect on the critical area of VLSI devices", *IEEE J. of Solid State Circuits*, vol. SC-20, No4, Aug. 1985, 878-880.
12. S. J. Fortune, "A sweepline algorithm for Voronoi diagrams", *Algorithmica*, 2, 1987, 153-174.
13. S. J. Fortune, "Numerical Stability of Algorithms for 2-d Delaunay triangulations", *International J. Comput. Geom. Applic.*, 5(1), 1995, 193-213.
14. I. Koren, "The effect of scaling on the yield of VLSI circuits", *Yield Modeling and defect Tolerance in VLSI circuits* W.R. Moore, W. Maly, and A. Strojwas Eds., Bristol UK: Adam-Hilger Ltd., 1988, 91-99
15. G. Liotta, F.P. Preparata, and R. Tamassia, "Robust Proximity Queries: An illustration of degree-driven algorithm design," SIAM Journal on Computing, 28, 3, 1998, 864-889.
16. W. Maly, "Computer Aided Design for VLSI Circuit Manufacturability", *Proc. IEEE*, Feb. 90, 356-392.
17. W. Maly, and J. Deszczka, "Yield Estimation Model for VLSI Artwork Evaluation", *Electron Lett.* vol 19, no.6, 226-227, March 1983

18. S. N. Meshkat and C. M. Sakkas. " Voronoi diagram for multiply-connected polygonal domains II: Implementation and application", *IBM J. of Research and Development*, Vol. 31, No. 3, May 1987
19. E. Papadopoulou and D.T. Lee, "Critical Area computation via Voronoi diagrams", *IEEE Trans. on Computer-Aided Design*, vol. 18, No. 4, April 1999, 463-474.
20. J. Pineda de Gyvez, C. Di, "IC Defect Sensitivity for Footprint-Type Spot Defects", *IEEE Trans. on Computer-Aided Design*, vol. 11, no 5, 638-658, May 1992
21. K. Sugihara, Y. Ooishi, and T. Imai, "Topology oriented approach to robustness and its applications to several Voronoi-diagram algorithms. In *Proc. 2nd Canad. Conf. Comput. Geom.*, pages 36-39, 1990
22. V. Srinivasan Personal Communication.
23. V. Srinivasan, L.R. Nackman, " Voronoi diagram for multiply-connected polygonal domains II: Algorithm", *IBM Journal of Research and Development*, Vol. 31, No. 3, May 1987
24. V. Srinivasan, L.R. Nackman, J.M. Tang, and S.N. Meshkat, "Automatic Mesh Generation Using the Symmetric Axis Transformation of Polygonal Domains", *Proceedings of the IEEE*,Vol. 80, No. 9, Sept. 1992, 1485-1501.
25. C.H. Stapper, "Modeling of Defects in integrated circuits photolithographic patterns", *IBM J. Research and Development*, vol.28, no.4, 461-475, 1984.
26. C. H. Stapper and R. J. Rosner, "Integrated Circuit Yield Management and Yield Analysis: Development and Implementation" *IEEE Trans. on Semiconductor Manufacturing* Vol. 8, No.2, 1995, 95-101.
27. I. A. Wagner and I. Koren, "An Interactive VLSI CAD Tool for Yield Estimation", *IEEE Trans. on Semiconductor Manufacturing* Vol. 8, No.2, 1995, 130-138.
28. H. Walker and S.W. Director, " VLASIC: A yield simulator for integrated circuits", *IEEE Trans. on Computer-Aided Design*, vol. CAD-5, no 4, 541-556, Oct. 1986.

Appendix A:

We shall use the definitions, notation, and results of Ref. [5] (appendix). Let $p(x_1, \ldots x_n)$ be a multivariate homogeneous polynomial and let $I, |I| \geq 2$, be a subset of variables in p of degree 1. Given a variable x_i of degree 1, p can be expressed as $p = p_i x_i + p_{i0}$, where p_i, p_{i0} are polynomials in all variables except x_i. Theorem 13 in Ref. [5] states that if p has degree at least 3, $|I| \geq 2$, and for some $i, j \in I$ coefficients p_i and p_j are distinct and irreducible, then p is irreducible. Recall that polynomial p is reducible[5] if it can be written as $p = \phi\psi$, where ϕ, ψ are multivariate homogeneous polynomials over the same variables satisfying $1 \leq degree(\phi),\ degree(\psi) < degree(p)$. Polynomials differing only by a multiplicative constant are not distinct. If p has degree 2 we have the following lemma.

Lemma A.1 *Let $p(x_1, \ldots, x_n)$ be a multivariate homogeneous polynomial of degree 2 , with $|I| \geq 2$. If for some $i, j \in I$ coefficients p_i and p_j are distinct and do not contain variables x_i nor x_j, then p is irreducible.*

Proof. Suppose for a contradiction that p is reducible. Then p can be written as $p = \phi\psi$, where ϕ, ψ are polynomials over the same variables, both of degree 1. A polynomial of degree 1 is a linear combination of all variables, $x_1, x_2, ..., x_n$, i.e., $\Sigma_{l=0}^{n}(q_l * x_l)$, where $q_l, l = 0, 1, ..., n$ are constants, $x_0 = 1$, and at least one of

$q_1, q_2, ..., q_n$, is nonzero. Let $\phi = \Sigma_{l=0}^{n}(q_l * x_l)$, and $\psi = \Sigma_{l=0}^{n}(r_l * x_l)$. Since p does not contain terms x_i^2 and x_j^2, q_i and r_i (resp. q_j and r_j) cannot both be non-zero. Let's first assume that $q_i \neq 0$ and $q_j \neq 0$. Then $r_i = r_j = 0$ thus, $p_i = q_i\psi$ and $p_j = q_j\psi$ i.e., p_i, p_j are not distinct; contradiction. Similarly for $r_i \neq 0$ and $r_j \neq 0$. Let's now assume that $q_i \neq 0$ and $r_j \neq 0$ (i.e., $r_i = 0, q_j = 0$). Then $p_i = q_i\psi$ and $p_j = r_i\phi$. But then the coefficient of x_j in p_i and the coefficient of x_i in p_j are non-zero; contradiction. □

To prove that the polynomials P_2, P_3, P_4 involved in Predicates $2, 3, 4$ are irreducible we apply Theorem 13 of Ref. [5] recursively to pairs of variables until we derive irreducible polynomials of degree 2. To show that a polynomial of degree 2 is irreducible we apply Lemma A.1. Schematically, the recursive application of Theorem 13 to a polynomial of degree k results in a binary tree of height $k-2$ where a node at level i corresponds to a polynomial of degree $k-i+1$. The root is the original polynomial of degree k and the leaves are irreducible polynomials of degree 2. Each node, holding a polynomial p, has two children, each one obtained by differentiating p with respect to a variable. Any two sibling nodes contain distinct polynomials.

Lemma A.2 *P_2, P_3, and P_4 are irreducible over the rationals.*

Proof. Let $(x_{m1}, y_{m1}), (x_{m2}, y_{m2})$ denote the endpoints of element $e_m, m = i, j, k$ or $m = 1, 2, \ldots, 6$. We only give the details for Predicate 2. For the remaining predicates we only give the sequence of pairs of variables used at each level of the derivation.

$P_2 = y_p D(L_t, e_j, e_k) - M_1(L_t, e_j, e_k)$, $I = \{x_{m1}, y_{m1}, x_{m2}, y_{m2}, y_p, m = j, k\}$. P_2 can also be written as $P_2 = p_{j1}y_{j1} + p_{j10}$ where p_{j1}, p_{j10} do not contain y_{j1}. $p_{j1} = (t(-x_{k1}+x_{k2}) - x_{j2}(-x_{k1}+x_{k2}) - x_{k2}y_{k1} + x_{j2}(y_{k1}-y_{k2}) + x_{k1}y_{k2} + (-x_{k1}+x_{k2})y_p)$. We can apply Theorem 13 of Ref. [5] to pair (y_p, y_{j1}). To show that $D(L_t, e_j, e_k)$ is irreducible we apply Lemma A.1 for variables (x_{j1}, y_{j1}). p_{j1} is of degree 2 and can be shown to be irreducible using Lemma A.1 for variables (x_{k1}, x_{k2}). Thus, the polynomial involved in Predicate 2 is irreducible.

$P_3 = -D(e_i, e_j, e_k)x_p + N_2(e_i, e_j, e_k)$. P_3 can also be written as $P_3 = p_i x_{i1} + p_{i0}$ (apply Theorem 13 of Ref. [5] to (x_p, x_{i1})). We can show that $D(e_i, e_j, e_k)$ is irreducible by applying Theorem 13 to (y_{i1}, x_{k1}) to derive two degree 2 irreducible polynomials, $p_{i1} = (-x_{j1} + x_{j2})(-x_{k1} + x_{k2}) + (-x_{k1} + x_{k2})(y_{j1} - y_{j2}) - (-x_{j1} + x_{j2})(y_{k1} - y_{k2})$ and $p_{k1} = -(-x_{j1} + x_{j2})(y_{i1} - y_{i2}) + (-x_{i1} + x_{i2})(y_{j1} - y_{j2}) - (y_{i1} - y_{i2})(y_{j1} - y_{j2})$. We show that p_{i1} is irreducible by applying Lemma A.1 to (x_{k1}, y_{k1}). We show that p_{k1} is irreducible by applying Lemma A.1 to (x_{i1}, y_{i1}). p_i is shown to be irreducible by applying Theorem 13 to (y_{k1}, y_{j1}).

$P_4 = N_2(e_1, e_2, e_3)D(e_4, e_5, e_6) - N_2(e_4, e_5, e_6)D(e_1, e_2, e_3)$ and $I = \{x_{m1}, y_{m1}, x_{m2}, y_{m2}, m = 1, 2, \ldots 6\}$. We apply Theorem 13 to the following sequence of pairs of variables: (x_{11}, x_{12}), (x_{31}, x_{32}), (y_{51}, y_{52}), (x_{61}, x_{62}), and (y_{21}, y_{22}). Each pair is applied to one level of the derivation tree, starting with (x_{11}, x_{12}). In each branch we end up with two distinct polynomials of degree 2 which are shown to be irreducible by applying Lemma A.1. □