Critical Area Computation -- A New Approach

Article · September 1999

DOI: 10.1145/274535.274548 · Source: CiteSeer

2 authors:

Evanthia Papadopoulou
University of Lugano
65 PUBLICATIONS **740** CITATIONS

SEE PROFILE

D. T. Lee
National Chung Hsing University
287 PUBLICATIONS **13,326** CITATIONS

SEE PROFILE

CRITICAL AREA COMPUTATION – A NEW APPROACH

Evanthia Papadopoulou[1] *D. T. Lee* [2]

[1]IBM TJ Watson Research Center, Yorktown Heights, NY 10598, USA
evanthia@watson.ibm.com
[2]Northwestern University, Evanston, IL 20208, USA
dtlee@ece.nwu.edu

ABSTRACT

In this paper we present a new approach for computing the critical area for shorts in a circuit layout. The critical area calculation is the main computational problem in VLSI yield prediction. The method is based on the concept of Voronoi diagrams and computes the critical area for shorts (for all possible defect radii, assuming square defects) accurately in $O(n \log n)$ time, where n is the size of the input. The method is presented for rectilinear layouts but it is extendible to general layouts. As a byproduct we briefly sketch how to speed up the grid method of Wagner and Koren [16].

1. INTRODUCTION

The critical area of a VLSI layout is a measure that reflects the sensitivity of the layout to defects occurring during the manufacturing process and it is widely used to predict the yield of a VLSI chip. Yield prediction is essential in today's VLSI manufacturing due to the growing need to control cost. Models for yield estimation are based on the concept of critical area which represents the main computational problem in the analysis of yield loss due to spot defects during fabrication. Spot defects are caused by particles such as dust and other contaminants in materials and equipment and are classified into two types: "extra material" defects causing shorts between different conducting regions and "missing material" defects causing open circuits. Extra material defects are the ones that appear most frequently in a typical manufacturing process and are the main reason for yield loss. For more information on yield estimation and spot defects see for example [4, 5, 7, 10, 11, 15, 13, 14, 16]

Quoting from [16], the yield of a chip, denoted by Y, is computed as $Y = \Pi_{i=1}^{m} Y_i$ where Y_i is the yield associated with the ith step of the manufacturing process. The yield of a single processing step is modeled as

$$Y_i = (1 + \frac{dA_c}{\alpha})^{-\alpha}$$

where d denotes the average number of defects per unit of area, α the clustering parameter, and A_c the critical area.

For 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. The defect density function has been estimated as follows[5, 7, 15, 16]:

$$D(r) = \begin{cases} cr^q/r_0^{q+1}, & 0 \le r_0 \\ cr_0^{p-1}/r^p, & r_0 \le r \le \infty \end{cases} \quad (1)$$

where p, q are real numbers (typically $p = 3, q = 1$), $c = (q+1)(p-1)/(q+p)$, and r_0 is some minimum optically resolvable size.

Existing methods for yield prediction and critical area computation can be classified into the following types:

1. Geometric methods: Compute $A(r)$ for several different values of r independently. Use the results to approximate A_c. The methods to compute $A(r)$ are usually based on *shape-expansion* followed by *shape-intersection* (see e.g [6]) and have a quadratic flavor. For rectilinear layouts there is also a scan-line method which computes critical areas in multiple layers. [13].
2. Virtual artwork approach: Build a virtual artwork having the same statistical features as the nominal IC layout. The virtual artwork is arranged in a form allowing easy calculation of the critical area as a function of the defect radius [9]. Accuracy is limited by differences in the detail of the nominal and virtual artworks.
3. Monte Carlo approach: Draw a large number of defects with their radii distributed according to $D(r)$, check for each defect if it causes a fault, and divide the number of defects causing faults by the total number of defects [18].
4. Grid approach: Assume an integer grid over the layout. Compute the critical radius (the radius of the smallest defect causing a fault at this point) for every grid point [16]. The method works in $O(I^{1.5})$ time, where I is the number of grid points. The accuracy depends on the density of the grid.

In this paper we present a new approach for computing the critical area for shorts in a single layer. The method is based on the concept of *Voronoi diagrams* and can accurately compute the total critical area A_c for square defects in $O(n \log n)$ time where n is the size of the input. In this paper we only deal with rectilinear layouts. However, our method is extendible to non-rectilinear layouts. To the best

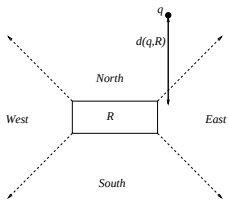

Figure 1. The 45° rays emanating from the corners of R partition the plane into four quadrants.

of our knowledge this is the first low polynomial algorithm to compute critical area for shorts accurately in irregular rectilinear layouts.

2. SQUARE DEFECTS AND THE L_∞ METRIC

In the critical area literature, defects have consistently been modeled as circles due to the common use of Euclidean geometry. To simplify the computation, in this paper, a spot defect of size r is modeled as a square of side $2r$ (i.e., a square of radius r). In reality spot defects have any kind of shape thus, modeling defects as squares is as good as the circle model. Moreover, in existing methods critical area is approximated in a way that would not make much difference if the defects were assumed squares or circles. (In [13] defects are also modeled as squares.) Modeling defects as squares corresponds to computing critical area in the L_∞ metric instead of the Euclidean. In L_∞, the distance between two points $p = (x_p, y_p)$ and $q = (x_q, y_q)$ is the maximum of the horizontal and the vertical distance between p and q i.e., $d(p, q) = \max\{|x_p - x_q|, |y_p - y_q|\}$. Intuitively, the L_∞ distance is the size of the smallest square touching p and q. Note that the "unit circle" in the L_∞ metric is a square of side two (i.e., a square of radius 1). The L_∞ distance between a point p and a line l is $d(p, l) = \min\{d(p, q), \forall q \in l\}$. In the following, unless otherwise noted, we always imply the L_∞ distance and thus we skip the term for brevity when there is no confusion.

3. THE VORONOI DIAGRAM OF POLYGONS

In L_∞, the bisector of two elements (lines or points) is the locus of points at equal L_∞ distance from the two elements. It can be regarded as the locus of points corresponding to centers of squares touching the two elements. Figures 3 and 4, illustrate the L_∞ bisector of two points and two lines respectively. Consider a rectangle R and the four 45° rays[1] emanating away from the vertices of R (see Figure 1). They partition the plane into four quadrants. For any point q in the north or south (resp. east or west) quadrant the L_∞ distance to R simplifies to the vertical (resp. horizontal) distance between q and the line through the respective horizontal (resp. vertical) edges of R. The four 45° rays through the corners of R are regarded as bisectors between the edges of R.

In Figure 2, the bisector of two rectilinear polygons is illustrated. It consists of portions of bisectors between edges of the two polygons. The border of each portion is induced by the 45° rays emanating from the corners of the polygons (illustrated by dashed lines in Figure 2.)

The Voronoi diagram of a set C of polygons is a partition of the plane into regions, called *Voronoi cells*, each of which is associated with a polygon, called the *owner*, of the cell. The Voronoi cell of a polygon P, denoted as $reg(P)$, is the locus of points closer to P than to any other polygon (see

[1] A 45° ray is a ray of slope +1 or −1.

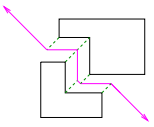

Figure 2. The L_∞ bisector of two rectilinear polygons.

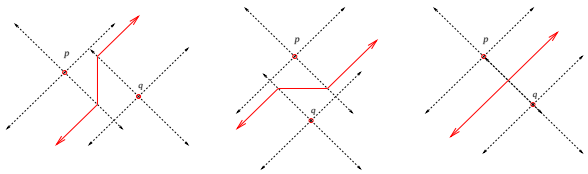

Figure 3. The L_∞ bisector of two points.

for example Figure 5). The Voronoi cell of P is further partitioned into finer cells each of which is associated with an element (edge or vertex) of P. The Voronoi cell of an element $e \in P$ is the locus of points closer to e than to any other element. Clearly, $reg(e) \subset reg(P)$. The boundary that borders two Voronoi cells is called a *Voronoi edge*, and consists of portions of *bisectors* between the owners of the cells. The point where three or more Voronoi edges meet is called a *Voronoi vertex*. For more information on Voronoi diagrams in general see [1, 2]. In the Euclidean metric the boundary of a Voronoi cell contains parabolic curves which makes the computation difficult in practice. For this reason we perform computations in the L_∞ metric.

The L_∞ Voronoi diagram of a set of polygons is a skeleton of straight line segments of linear combinatorial complexity [12]. In the special case of rectilinear polygons, the L_∞ Voronoi diagram is a particularly simple skeleton [12]. It consists of line segments in only four orientations, vertical, horizontal, and lines of slope (+1) and (−1) [12]. Figures 9 and 5, illustrate the Voronoi diagram of a set of rectangles and a set of rectilinear polygons respectively. Solid edges illustrate the Voronoi edges separating the cells of different polygons and dashed edges illustrate Voronoi edges induced by the edges of the same polygon. Note that a Voronoi vertex is the meeting point of at least three Voronoi edges (bisectors) and thus it is equidistant from at least three elements. An upper bound on the size of the L_∞ Voronoi diagram of a set of rectilinear polygons is $(2n - 2)$ where n is the number of vertices [12].

The Voronoi diagram encodes nearest neighbor information for every point in the plane. For any point $t \in reg(e)$, where e is an element of polygon P, t is closer to P than to any other polygon. Moreover, t is closer to e than to any other polygonal element (in the L_∞ sense). We say that polygon P and in particular element e is the *nearest neighbor* of t. Several algorithms are available in the literature for computing the Euclidean Voronoi diagram of line segments in time $O(n \log n)$ in expected or worst case (see [2] for a survey). Most of them could be modified to compute the simpler L_∞ diagram, however not all are simple to implement. In Section **??** we give a simple plane sweep

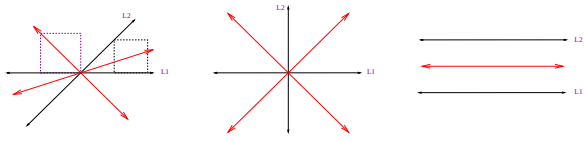

Figure 4. The L_∞ bisector of two lines.

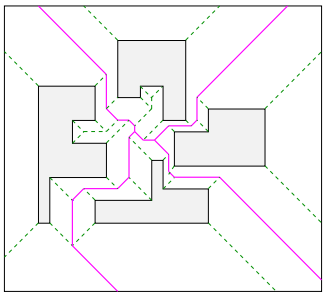

Figure 5. The L_∞ Voronoi diagram of four polygons.

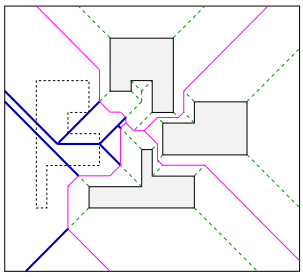

Figure 6. The 2nd order Voronoi diagram in $reg(P)$.

algorithm to compute the L_∞ Voronoi diagram of a set of rectilinear polygons.

4. VORONOI DIAGRAMS FOR CRITICAL AREA

Let's assume the set of polygons C represents a layer of a VLSI layout. The *critical radius* of a point t is the radius of the smallest defect centered at t which overlaps with at least two different polygons (shapes) in different nets. The critical radius of t reflects the size of the smallest defect that would cause a short at point t. To determine the critical radius of a point t whose nearest polygon is P we need the second nearest polygon to t, say Q (assuming that P and Q belong to different nets). If w is the element of Q nearest to t, the critical radius of t is $d(w,t) = d(Q,t)$.

Consider now the partitioning of the plane obtained as follows: For every polygon $P \in C$ partition the interior of $reg(P)$ by the Voronoi diagram of $C - P$. The subdivision derived in this manner is called the *2nd order Voronoi diagram* of C and provides 2nd nearest neighbor information. In Figure 6, the thick lines illustrate the *2nd order Voronoi diagram* restricted in the interior of $reg(P)$ where P is shown in dotted lines. More formally, the 2nd order Voronoi region of an element $s \in C - P$ within the Voronoi cell of P is defined as $reg_P(s) = \{x \mid d(s,x) \leq d(t,x), \forall t \in C - P\}$. Region $reg_P(s)$ is *owned* by the unique pair (P,s). The critical radius of every point $x \in reg_P(s)$ is the distance between x and s. Because s is the element inducing the critical radius for every point $x \in reg_P(s)$, we drop P and say that s is the *owner* of $reg_P(s)$. The size of the *2nd order Voronoi diagram* cannot be more than twice the size of ordinary Voronoi diagram i.e., it is linear in the size of the input. This 2nd order Voronoi diagram of polygons is based on the concept of *kth order Voronoi diagram* of line segments [8] which is defined as a planar subdivision such that each region is closest to k line segments.

To derive the 2nd order diagram within the Voronoi cell of each polygon P we only need to consider the Voronoi neighbors of $reg(P)$. In fact, as shown in Figure 6, the 2nd order Voronoi edges in $reg(P)$ are obtained by extending the 1st order edges incident to the boundary of $reg(P)$ into $reg(P)$.

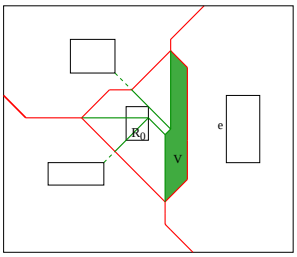

Figure 7. 2nd order Voronoi cell of vertical edge e

A detail worth pointing out in the computation of critical area for shorts is that a pair of distinct polygons may be part of the same net via a connection through a different layer. In this case a defect overlaping with these polygons does not cause a short and thus should not be contributing to critical area. In the above approach it is easy to accommodate this point. Let P, Q be two disjoint polygons that belong to the same net. If the Voronoi cells of P and Q are not neighboring then any defect overlaping P and Q must also overlap with some other polygon. If on the other hand the regions are neighboring we can simply unite $reg(P)$ and $reg(Q)$ into $reg(P \cup Q)$ by removing the adjacent Voronoi edges and treat the resulting region as any other when computing the 2nd order diagram.

5. COMPUTING CRITICAL AREA

We have a layer in a circuit layout consisting of a collection of rectilinear polygons C. We assume that overlaping polygons have been unified into single shapes and thus we can assume that all polygons are disjoint. Our goal is to compute the critical area for shorts i.e., to evaluate the integral $A_c = \int_0^\infty A(r)D(r)dr$, where the defect density function is given by Eq.(1). Using typical values for p, q and c we derive the widely used defect size distribution $D(r) = r_0^2/r^3$. (Since r_0 is typically smaller than the minimum feature size we ignore $D(r)$ for $r < r_0$). 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 polygons in different nets.

Let's assume that we are given the 2nd order L_∞ Voronoi diagram of C (with the modification described in the previous section to accommodate polygons of the same net). It is not hard to see that $A_c = \sum_V A_c(V)$ 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.

Let's first concentrate on computing critical area within a single (2nd order) Voronoi cell V having as owner an edge e. Let's assume without loss of generality that e is a vertical edge. Since the edges of V are bisectors induced by e they must be vertical or 45°. In Figure 7 the shaded region illustrates a 2nd order Voronoi cell of the vertical edge e.

Consider a decomposition of V into trapezoids by drawing the horizontal lines emanating from the Voronoi vertices of V (see Figure 8a). Each trapezoid $\mathcal{T}$ can be further decomposed into triangles and at most one rectangle $\mathcal{R}$ by

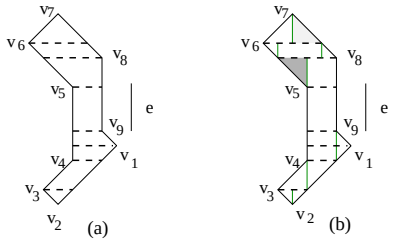

Figure 8. The decomposition of V into trapezoids

drawing the vertical lines through its vertices. (In case $\mathcal{T}$ is a slanted parallelogram continue the decomposition recursively, see Figure 8b). We distinguish between two kinds of triangles in the above decomposition depending on the relevant position with respect to e of their vertical edge and opposite apex. If the apex lies between e and the vertical edge (comparing abscissae) the triangle is called *diverging*; otherwise it called *converging*. In Figure 8, the lightly shaded triangle is diverging and the darker shaded one is converging. 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 defect radii i.e., $r_j = x_e - x_j$ and $r_k = x_e - x_k$ where x_j, x_k and x_e denote the x-coordinates of v_j, v_k and e respectively.

Lemma 1 *The critical area within a rectangle $\mathcal{R}$, a diverging triangle T_1 and a converging triangle T_2 is given by the following formulas, using the "r_0^2/r^3" defect density distribution:*

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

$$A_c(T_1) = \frac{r_0^2}{2}\left(\ln\left(\frac{r_k}{r_j}\right) - \frac{l}{r_k}\right) \tag{3}$$

$$A_c(T_2) = \frac{r_0^2}{2}\left(\frac{l}{r_j} - \ln\left(\frac{r_k}{r_j}\right)\right) \tag{4}$$

where l is the size of the vertical side of $\mathcal{R}, T_1$ and T_2, and $r_k, r_j, r_k > r_j$ are the maximum and the minimum critical radius of their vertices.

We 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 l/r_l corresponding to internal decomposition edges cancel out. Similarly, for logarithmic terms involving endpoints of the decomposition other than Voronoi vertices. Let's classify the edges of V as *converging* or *diverging* as follows. A vertical edge e_v is called diverging (resp. *converging*) if the interior V and the owner e lie at opposite sides (resp. same side) of e_v. A 45° edge is called *diverging* (resp. *converging*) if it is incident to a diverging (resp. converging) triangle. In the formulas of Lemma 1, terms corresponding to diverging edges get added while terms corresponding to converging edges get subtracted. Every Voronoi edge bounds exactly two cells and receives the same classification (diverging or converging) with respect to both cells. We derive the following theorem.

Theorem 1 *Given the 2nd order L_∞ Voronoi diagram of rectilinear 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:*

$$A_c = r_0^2\left(\sum_{dvrg\ e_i}\frac{l_i}{r_i} - \sum_{conv\ e_m}\frac{l_m}{r_m} + \sum_{dvrg\ e_{45}}\ln\frac{r_k}{r_j} - \sum_{conv\ e_{45}}\ln\frac{r_k}{r_j}\right)$$

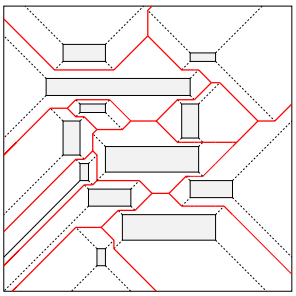

Figure 9. The L_∞ Voronoi diagram of rectangles.

where l_i and r_i denote the length and the critical radius of an orthogonal Voronoi edge, and $r_k, r_j, r_k > r_j$ denote the maximum and the minimum critical radius of a 45° Voronoi edge. The first and second summations are taken over all diverging and all converging orthogonal Voronoi edges. The third and forth summations are taken over all diverging and all converging 45° Voronoi edges respectively.

6. A SWEEP-LINE ALGORITHM TO COMPUTE THE L_∞ VORONOI DIAGRAM

In this section we give a plane-sweep algorithm to compute the L_∞ Voronoi diagram of a rectangular layout consisting of n rectangles (see Figure 9). We consider rectangles instead of rectilinear polygons for simplicity of presentation; the same algorithm can be easily generalized to rectilinear layout. The time complexity is $O(n \log n)$ and it is based on the wavefront paradigm introduced by Dehne and Klein [3] for the Voronoi diagram of points.

Each rectangle R_i is decomposed into four elements, R_i^s, R_i^n, R_i^e and R_i^w, representing south, north, east and west edges of R_i. Emanating from each corner there is a 45° ray *directed* from the corner vertex outwards which is the bisector between two consecutive elements of R_i.

We shall apply a vertical sweep-line $\mathcal{L}$ sweeping across the entire plane from left to right. The set of rectangles at any instant of the sweeping process will be partitioned into three subsets, S_l, S_m and S_r, corresponding to those that lie totally to the left of $\mathcal{L}$, intersect $\mathcal{L}$, and lie totally to the right of $\mathcal{L}$, respectively. The sweep line, which is cut by the rectangles in S_m, is decomposed into $|S_m|+1$ sweepline segments, two of which are unbounded.

We shall maintain a partial Voronoi diagram computed so far for the rectangles in S_l, the portion of the rectangles in S_m that lie to the left of $\mathcal{L}$, and the sweepline segments. At any instant of the sweeping process the boundary of the Voronoi cell associated with a sweepline segment is called a *wavefront curve* and consists of bisectors induced by the sweepline segment. The collection of the wavefront curves for all sweepline segments is referred to as the *wavefront*. In Figure 10a, the wavefront is shown in dashed-lines. The elements of rectangles defining a bisector in the wavefront are referred to as *wavefront elements*. The Voronoi edges that have an endpoint common with the wavefront are called *spike bisectors*. In Figure 10a, the wavefront elements are shown in bold face and the spike bisectors are shown in solid and thicker lines. As the sweepline moves to the right, the wavefront bisector will shift to the right and so do the spike bisectors.

Associated with the plane-sweep algorithm there are two major components: an *event list* and a *sweep-line status*. The event list, implemented as a *priority queue* Q, consists of abscissae, called *priority values* at which the sweep-line

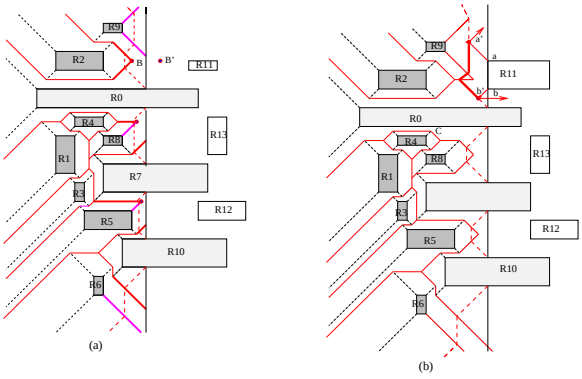

Figure 10. Construction of the Voronoi diagram by plane sweep method.

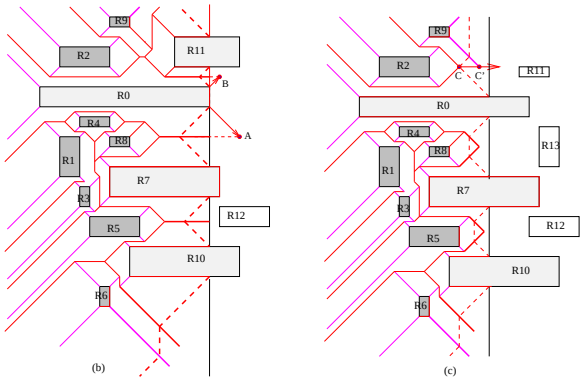

Figure 11. Events in the plane sweep: b) Right edge of a rectangle. c) Potential Voronoi vertex.

status will change. The sweep-line status, implemented as a *height-balanced* binary tree $\mathcal{T}$, implicitly maintains the wavefront. In particular, we keep the spike bisectors and the north and south edges of those rectangles in S_m ordered according to their intersections with the wavefront. To simulate the wavefront movement to the right as the plane sweep proceeds we parameterize the right endpoints of spike bisectors with respect to the abscissa of the sweep line t. In other words we keep the abscissa of the right endpoint of a spike bisector as a linear function of t. This function is derived by the property that the endpoints are equidistant from the sweep-line and the elements inducing the bisector. The abscissa of the right endpoint of a north or south edge in $\mathcal{T}$ is given by t. In the event list we have two kinds of events: 1) events corresponding to vertical edges of rectangles referred to as *edge events* and 2) events corresponding to the intersection point of two neighboring spike bisectors referred to as *spike events*. The priority value of an edge event is the abscissa of the edge. A spike event corresponds to a potential Voronoi vertex that may or not appear in the final Voronoi diagram. The priority value of a spike event I of abscissa x_i is $x_i + d(e_j, I)$ where $I \in reg(e_j)$ For example, in Figure 10a, the spike event associated with the potential Voronoi vertex B has as priority value the abscissa of the point marked as B' In other words, the spike event corresponding to B occurs when the sweepline reaches B'.

Let us discuss three cases to handle the event points.

1. Left edge of rectangle R_i (Figure 10b). Let the edge be denoted $\overline{a, b}$ where a and b are the northwest and southwest corners of R_i respectively. We construct two 45° rays emanating from a and b respectively i.e., bisectors $B(R_i^w, R_i^n)$, $B(R_i^s, R_i^w)$
   (a) Identify the intersection points a' and b' of $B(R_i^w, R_i^n)$ and $B(R_i^s, R_i^w)$ with the wavefront respectively. This can be done by binary search in $\mathcal{T}$. Let e_a and e_b denote the wavefront elements owning a' and b'.
   (b) Update the partial Voronoi diagram by explicitely constructing the bisectors of the wavefront between a' and b'.
   Update $\mathcal{T}$ by removing those spike bisectors between a' and b'. The potential Voronoi event points associated with these spike bisectors can be either *deleted* from the priority queue now or *ignored* later when they get extracted from the priority queue.
   (c) Two new spike bisectors $B(e_a, R_i^n)$ and $B(R_i^s, e_b)$ are created with startpoints a' and b' respectively. At most two new potential Voronoi vertices defined by the intersection of bisector $B(R_i^s, e_b)$ and its lower neighbor and by the intersection of bisector $B(e_a, R_i^n)$ and its upper neighbor will be created and inserted to the priority queue. Note that the priority values of these potential Voronoi vertices should be recorded correctly.
   (d) Update the sweep-line status by inserting R_i^n and R_i^s, and two new spike bisectors $B(e_a, R_i^n)$ and $B(R_i^s, e_b)$. In Figure 10b, these two new spike bisectors, $B(R_9^e, R_{11}^n)$ and $B(R_{11}^s, R_0^n)$ are shown with an arrowhead.
2. Right edge of rectangle R_i (figure 11b). In this case the two elements R_i^s and R_i^n must be present in T in consecutive order. These two elements in $\mathcal{T}$ need to be deleted from $\mathcal{T}$ and replaced by two new bisector elements, $B(R_i^e, R_i^s)$ and $B(R_i^n, R_i^e)$ respectively. As in case 1(c), these two newly created bisectors should be checked against their neighboring elements to see if they would create potential Voronoi vertices to be inserted in the priority queue and the appropriate priority values are recorded.
3. Potential event point p which is the intersection of two spike bisector elements, say $B(R_k^u, R_j^v)$ and $B(R_j^v, R_i^w)$ for some wavefront rectangles $R_i, R_j, R_k \in S_l$ (Figure 11c). Note that this point is equidistant from three rectangles. If either of the two bisector elements was marked previously due to the construction of the Voronoi diagram (case 1(b)), then we discard it, as it is no longer relevant. Otherwise, the two bisector elements $B(R_k^u, R_j^v)$ and $B(R_j^v, R_i^w)$ must be adjacent in $\mathcal{T}$. Both elements should be deleted from $\mathcal{T}$ and replaced by a new spike bisector element $B(R_k^u, R_i^w)$. (The element R_j^v ceases to be a wavefront element.) This new spike bisector should also be checked against its neighboring elements for possible intersections, creating potential Voronoi vertices to be inserted in the priority queue with the appropriate priority values.

Lemma 2 *The Voronoi diagram in the L_∞-metric for a set of rectangles can be computed in $O(n \log n)$ time.*

Once the Voronoi diagram is available we can compute the 2nd order diagram in each Voronoi cell in a similar fashion. Due to the page limit we skip the details. Note however that as soon as an individual Voronoi cell is computed we can directly compute the 2nd order subdivision within the cell and derive the corresponding critical area.

7. A MORE EFFICIENT GRID METHOD

In [16], Wagner and Koren consider the area of the layout as a set of I grid points of resolution γ. The input polygons (shapes) are also treated as collection of grid points. They compute the critical radius at every grid point by visiting on average $\sqrt{I}$ other grid points i.e, total time complexity is $O(I^{1.5})$. Once the critical radius at every grid point is known the integral of the total critical area can be easily computed; the error depends on the grid resolution γ.

Imagine now that the shape grid points are sources of waves that start propagating at the same time with the same speed. The wave has the id of the net where the source grid point belongs. Waves of the same id are considered identical (although they come from different sources). The wave propagation will assign to every grid point t a pair of distance labels (d_1, d_2), where d_1 indicates the distance from the source grid point whose wave is the first one to reach t and d_2 indicates the distance from the source grid point of different id whose wave is the second one to reach t. Clearly d_2 is the critical radius at point t. During the propagation we allow exactly two distinct waves to overlap. When a grid point p is visited by wave w the following happens: 1) if p has already been visited by wave w or if p has already been visited by two other distinct waves the propagation stops at p. 2) Otherwise, p updates its labels (updates label d_1 if w is the first wave to reach p or label d_2 if w is the 2nd wave to reach p) and propagates w to its immediate neighbors (grid points) increasing the distance label by 1.

This construction is an implicit mechanical construction of the 2nd order Voronoi diagram and thus correctness follows from the previous discussion. Every grid point is visited by the waves exactly twice and thus the critical radius at every grid point can be computed in $O(I)$ time. Then the critical area can be computed as described in [16].

8. CONCLUSION

We have shown how to compute and use the second order Voronoi diagram of rectilinear polygons to efficiently compute critical area for shorts in a single layer. The value of critical area can then be used in evaluating yield. Furthermore we can use the Voronoi diagram to identify the places in the layout that are most vulnerable to spot defects. We can readily obtain information about those edges that contribute most to critical area. By slightly perturbing such edges we may be able to reduce the value of critical area and thus increase yield. Note that after moving an edge the Voronoi diagram can be updated dynamically in time proportional to the changes caused to the diagram because of the move. Thus the Voronoi diagram approach is suitable for an interactive critical area tool. Note also that the *topographic map* of [16] essentially provides a bitmapped version of the more critical Voronoi edges.

This method can also be extended to non-rectilinear layouts. The definition of the L_∞ Voronoi diagram of non-rectilinear polygons remains the same. The difference is that the diagram will consist of edges in more than four orientations. For example, if the layout contains edges of slope ± 1 the L_∞ Voronoi diagram will consist of edges in eight orientations: vertical, horizontal, (± 1)-slope, $(\pm 1/3)$-slope and (± 3)-slope lines [12].

Acknowledgment. The authors would like to thank Dr. Dan Ostapko for reading through an earlier draft and providing useful comments. The second author's research is supported in part by the Office of Naval Research under the Grants No. N00014-95-1-1007 and No. N00014-97-1-0514.

REFERENCES

[1] F. Aurenhammer, "Voronoi diagrams: A survey of a fundamental geometric data structure," *ACM Comput. Survey,* 23 1991, 345-405.

[2] F. Aurenhammer and R. Klein, "Voronoi Diagrams" chapter 18, *Textbook on Computational Geometry*, J.Sack and G. Urrutia (eds).

[3] F. Dehne and R. Klein, "The Big Sweep": On the power of the Wavefront Approach to Voronoi Diagrams", *Algorithmica*(1997), 17, 19-32.

[4] 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

[5] 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, No 4, Aug. 1985, 878-880.

[6] S. Gandemer, B.C. Tremintin, and J.J. Charlot, "Critical Area and critical levels calculation in IC yield modeling", *IEEE J. of Solid State Circuits*, vol.35, No 2, Feb. 1988, 158-166.

[7] 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

[8] D. T. Lee, "On k-nearest neighbor Voronoi diagrams in the plane," *IEEE Trans. Comput.,* Vol. C-31, No. 6, June 1982, 478-487.

[9] W. Maly, "Modeling of lithography related yield losses for CAD of VLSI circuits" *IEEE Transactions on Computer-Aided Design*, vol. CAD-4, no 3, 166-177, July 1985.

[10] W. Maly, "Computer Aided Design for VLSI Circuit Manufacturability", *Proc. IEEE*, Feb. 90, 356-392.

[11] W. Maly, and J. Deszczka, "Yield Estimation Model for VLSI Artwork Evaluation", *Electron Lett.* vol 19, no.6, 226-227, March 1983

[12] E. Papadopoulou and D.T. Lee, "L_∞ Voronoi Diagrams and Applications to VLSI Layout and Manufacturing", Manuscript.

[13] 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

[14] 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.

[15] C.H. Stapper, "Modeling of Defects in integrated circuits photolithographic patterns", *IBM J. Research and Development*, vol.28, no.4, 461-475, 1984.

[16] 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.

[17] H. Walker, "VLASIC system user manual, release 1.3", CMU, Aug 1990

[18] 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.

CRITICAL AREA COMPUTATION – A NEW APPROACH

Evanthia Papadopoulou[1] *and D. T. Lee* [2]

[1]IBM TJ Watson Research Center, Yorktown Heights, NY 10598, USA
evanthia@watson.ibm.com
[2]Northwestern University, Evanston, IL 20208, USA
dtlee@ece.nwu.edu

In this paper we present a new approach for computing the critical area for shorts in a circuit layout. The critical area calculation is the main computational problem in VLSI yield prediction. The method is based on the concept of Voronoi diagrams and computes the critical area for shorts (for all possible defect radii, assuming square defects) accurately in $O(n \log n)$ time, where n is the size of the input. The method is presented for rectilinear layouts but it is extendible to general layouts. As a byproduct we briefly sketch how to speed up the grid method of Wagner and Koren [16].