An Interactive VLSI CAD Tool for Yield Estimation

Israel A. Wagner and Israel Koren, *Fellow, IEEE*

Abstract—**The yield of a VLSI chip depends on the sensitivity of the chip to defects occurring during the fabrication process, among other factors. To predict this sensitivity, one usually needs to compute the so-called critical area (A_c), which reflects how many and how large the defects must be in order to result in a circuit failure. The main computational problem in yield estimation is to calculate A_c efficiently for complicated, irregular layouts. A novel approach is suggested for this problem that results in an algorithm that will solve it efficiently. This paper provides an interactive, accurate, and fast method for the evaluation of critical area as a design tool; the tool utilizes good visual feedback to allow layout improvement for higher yield. The algorithm is compared to other yield-prediction methods, which use either the Monte Carlo approach (VLASIC) or a deterministic approach (SCA); the algorithm is shown to be faster. It also has the advantage that it can graphically show a detailed 'defect sensitivity map' that can assist a chip designer in improving the yield of his/her layout.**

I. Introduction

FOLLOWING [4], [8], [9], and [13], the yield of a chip, denoted by Y, is computed as $Y = \prod_{i=1}^{m} Y_i$, where Y_i is the yield associated with the ith step of the manufacturing process. For convenience, the subscript will be omitted and Y will be referred to as the yield of a single processing step. Using the negative binomial distribution, the yield of a single processing step is modeled as

$$Y = \left(1 + \frac{dA_c}{\alpha}\right)^{-\alpha} \tag{1}$$

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

$$A_c = \int_0^\infty A(r)D(r)\,dr \tag{2}$$

where $A(r)$ is the area in which the center of a defect of radius r must fall in order to cause a circuit failure. For a given circuit layout C it is defined as

$$A(r) = \iint_{(x,y)\in C} \delta(x,y,r)\,dx\,dy \tag{3}$$

Manuscript received November 5, 1993; revised July 27, 1994. This work was supported in part by the National Science Foundation under Contract MIP-9305912. A preliminary version of this paper was presented at the *IEEE Workshop on Defect and Fault Tolerance in VLSI Systems*, Venice, Italy, October 27–29, 1993.

I. A. Wagner is with IBM Israel—Science and Technology, Matam, Haifa 31905, Israel.

I. Koren is with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01003 USA.

IEEE Log Number 9409813.

where

$$\delta(x,y,r) = \begin{cases} 1, & \text{a defect of radius } r \text{ at } (x,y) \\ & \text{causes a circuit failure} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where $D(r)$ is the density function of the defect size. Experimental data on defects in many wafers lead to the following formula [8], [9], [18]:

$$D(r) = \begin{cases} cr^q/r_0^{q+1}, & 0 \le r \le r_0 \\ cr_0^{p-1}/r^p, & r_0 \le r \le \infty \end{cases} \tag{5}$$

where p and q are real numbers (typically $p \approx 3, q \approx 1$ [18]), and c is given by $c = (q+1)(p-1)/(q+p)$ [3].

The problem in yield estimation is to calculate A_c for a complicated, irregular layout. Existing yield-prediction algorithms are based on one of the following methods:

- *Deterministic Approaches:* Compute $A(r)$ for several values of r by either the virtual artwork method [12], [13], scan-line [15], [16], shape-expansion [5], [17], or heuristic analytical expressions [2]. Equation (2) is then used to approximate A_c, and (1) is used to predict the yield.
- *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, then divide the number of defects causing circuit failures by the total number of defects [20], [21].

Algorithms based on the deterministic approach are usually faster, while the Monte Carlo approach yields more precise results but needs substantially more CPU time. Hence, neither method is suitable as a precise interactive design tool. In this paper a new analytic approach is suggested that leads to a faster algorithm for computing A_c. Such an algorithm can be used as the core of an interactive yield-prediction tool, which can help a VLSI designer to make yield-oriented decisions early in the design process, rather than the traditional approach of completing the layout and leaving the yield issues to the fabrication phase.

The rest of the paper is organized as follows: Section II includes the suggested analytic approach, Section III describes the algorithm, and in Section IV numerical and graphic results are presented.

II. Critical Area Analysis—A New Approach

A. Computing the Critical Area

In general, $A(r)$ is an area measure, and hence can be expressed as an integral, for a given cell C, as was shown in (3) and (4) above.

Let us denote by u an ordered set of coordinates (x, y). Thus,

$$A_c = \int_{r=0}^{\infty} D(r)\left(\int_{u\in C} \delta(u,r)\,du\right) dr. \tag{6}$$

Note that u is a 2-D point; hence du is an infinitely small area unit. The two integrations are independent, so their order can be changed, yielding

$$A_c = \int_{u\in C}\left(\int_{r=0}^{\infty} \delta(u,r)D(r)\,dr\right) du. \tag{7}$$

Let us denote by $S(u)$ the internal integral. In fact, $S(u)$ is the defect sensitivity at point u, i.e., the probability of a circuit fault caused by a defect at point u; it is given by

$$S(u) = \int_{r=0}^{\infty} \delta(u,r)D(r)\,dr. \tag{8}$$

We then obtain

$$A_c = \int_{u\in C} S(u)\,du. \tag{9}$$

Let us assume that $\delta(u,r)$ is monotonic in r; that is, if a defect of size r at point u results in a circuit fault, then a defect of size $r' > r$ at the same point will also result in a circuit fault. Further, let $r_c(u)$ denote the critical radius at point u, i.e., the minimal size of a defect causing a circuit fault at point u

$$r_c(u) = \min_r\{\delta(u,r) = 1\}$$

This will simplify the formula for $S(u)$ resulting in

$$S(u) = \int_{r=r_c(u)}^{\infty} D(r)\,dr. \tag{10}$$

At this point, the problem of calculating the critical area A_c has been reduced to that of calculating the critical radius at each point of the layout, since the integral in (10) can be precalculated for several values of r_c. In practice, if a simple formula like (5) is used, then its integral can be represented in a closed form.

In order to be practical, let us consider the area of the layout as a set of grid points, each of which may be either empty or belong to a shape in the layout. The grid resolution is γ, which has to be small enough to cover all significant shape variations. If our layout has width W and height H, then there are WH/γ^2 points in it. The layout is described as a set of shapes $L = \{P_1, P_2, \cdots, P_k\}$, where the P denotes sets of points, and k is the number of shapes in the layer under consideration. Henceforth, 'points' has the meaning of 'grid points'.

Based on the above analysis, one can employ a naive algorithm for calculating $r_c(p)$, as depicted in Fig. 1. The problem with ALG1 is its high complexity. There are WH/γ^2 points, and for each point we need to scan all the other points (in the worst case, when there are only 2 small polygons near the edge of the area). Hence, the complexity of running ALG1 on the whole cell is $O(WH/\gamma^2)^2$. For example, if the layout is 100 μm $\times$ 100 μm and we set $\gamma = 0.1$ μm, then we need $O(10^{12})$ computation steps. In the next section, a mathematical observation is presented that can be used to substantially speed up this computation.

```
Function r_c(u):real;
var r: real; begin
   r := 0;
   repeat
      r := r + γ;
      scan the points in the r-vicinity of u
   until two points, u1, u2 that belong to two different shapes are found;
   r_c(u) := max{|u − u1|, |u − u2|}
end;
```

Fig. 1. Naively computing $r_c(u)$ (ALG1).

B. Computing $r_c(u)$ Efficiently—A Ring Theorem

Let us first discuss the 'short-circuit' type faults. We shall later show that the same approach holds for 'open-circuit' faults. In this context, $r_c(u)$ is the minimal radius of a circle around u that intersects more than one shape. A defect with such a radius will cause a short circuit.

We denote by $|u - u'|$ the distance between the points u and u'. (Although we use the Euclidean distance in this paper, the theory holds for other measures as long as the triangle inequality holds.) Also, we denote by $r_i(u)$ the distance from u to the ith nearest shape of the layout; we will be interested in $r_1(u)$ and $r_2(u)$, the distances to the nearest and second nearest metal polygons, respectively. From the above definitions, the critical radius at point u is the radius of the smallest circle around u that covers points from two (or more) polygons, hence $r_c(u) = r_2(u)$.

The corollary of the following theorem states that the critical radii of two points cannot differ by more than the distance between these points.

Theorem 1: Let u and u' be two points in a layout. Then

$$\forall i \in \{1,2\}\colon |r_i(u) - r_i(u')| \le |u - u'|.$$

In other words, the functions $r_1(\cdot), r_2(\cdot)$ are 'Lipschitz continuous' with the Lipschitz constant equal to 1 (See [1], [11] for definition and applications of Lipschitz continuity.)

Corollary 1: Any polygon that may affect $r_c(u)$ intersects either the ring:

$$\{v|r_1(u') - |u - u'| \le |v - u| \le r_1(u') + |u - u'|\}.$$

or the ring:

$$\{v|r_2(u') - |u - u'| \le |v - u| \le r_2(u') + |u - u'|\}.$$

The proofs of Theorem 1 and Corollary 1 can be found in the appendix. Note that the same arguments that are used in the analysis of short-circuit type defects can also be used for open-circuit type defects, by considering the *borders* of conducting shapes rather than the shapes themselves, as shown in Fig. 4. This is achieved in the following way:

1) for each net N_i, replace its polygon by several edges named $N_{1000i}, N_{1000i+1}, \cdots$;
2) to the resulting layout, apply procedure YMAP, but look for "shorts" only between nets whose index difference is smaller than 1000. (It is assumed arbitrarily that the number of edges in a polygon is smaller than 1000.)

```
Function r_c(u):real;
var   d: real;
begin
   d := |u - u'|
   repeat
      scan the points in the two rings:
         R1:{v | r1(u') - d ≤ |v - u| ≤ r1(u') + d}
         R2:{v | r2(u') - d ≤ |v - u| ≤ r2(u') + d}
   until two points, u1, u2 that belong to two different shapes
      and are nearest to u are found;
   r1(u) := min{|u - u1|, |u - u2|};
   r2(u) := max{|u - u1|, |u - u2|};
   r_c(u) := r2(u);
end;
```

Fig. 2. Efficiently computing $r_c(u)$ using $r_1(u')$, $r_2(u')$ (ALG2).

```
Procedure YMAP(L : layout; ε_max, r_g : real);
Input:  * A layout L with width W and height H.
        * ε_max - A bound on the error in A_c.
        * r_g - The minimal ground rule for the layer under test.
Output: * The defect sensitivity S(x, y) for each grid-point (x, y) ∈ L.
        * A_c = The critical area of L.
var     A_c, γ, r_c: real;
        i, j, m, n: integer;
begin
   γ := 1.061 · ε_max · r_g;   /* this number results from Theorem 2 */
   A_c := 0, m := ⌈W/γ⌉, n := ⌈H/γ⌉
   for i := 1 to m,   j := 1 to n do
      r_c := r_c(iγ, jγ);
      S(iγ, jγ) := ∫_{2r_c}^{∞} D(t)dt;
      A_c := A_c + γ²S(iγ, jγ);   /* γ² is an area element for piecewise integration*/
end;
```

Fig. 3. Computing the defect-sensitivity $S(\cdot)$ and the critical area A_c (YMAP).

Fig. 5 shows a layout example illustrating the above.

Using Theorem 1, one can devise a better algorithm for calculating $r_c(u)$, which makes use of the previously calculated values $r_1(u')$ and $r_2(u')$, where u' is a neighbor of u on the grid.

In ALG2, shown in Fig. 2, we perform an exhaustive search for only one point. Then, for all the other points we need only scan a ring whose radius cannot exceed the edge of the cell, and whose width is 2γ. Hence, the worst case complexity of running ALG2 on the whole cell is $O(WH/\gamma^2)^{1.5}$, which implies that, in the example above, ALG2 is 1000 times faster than ALG1. A more sophisticated analysis leads to an even better bound of $O(WH\overline{r_c}/\gamma^3)$, where $\overline{r_c}$ is the average critical radius of the layout. It is interesting to note that the complexity is inversely proportional to the layout density; as the number of shapes grows, $\overline{r_c}$ decreases, as does the complexity of ALG2.

Using (9) and (10) above, and the procedure ALG2 to calculate $r_c(x, y)$, the sensitivity and critical area for an arbitrary cell can be approximated as depicted in Fig. 3. In order to get a good approximation for A_c, one must choose a γ such that (1) γ is small enough so that we will not miss any significant variation in the critical radius between grid points and, (2) the number of test-points, given by WH/γ^2, is not too large in order to keep the algorithm efficient in time and space.

The theorem that follows shows the linkage between γ and the maximal error.

Theorem 2: Denote by A_c the critical area of the layout, and by A_c' the critical area as computed by YMAP. Then, for a typical distribution of defect size, the relative error satisfies

$$\left|\frac{A_c' - A_c}{A_c}\right| \leq \frac{0.943\gamma}{r_g} \tag{11}$$

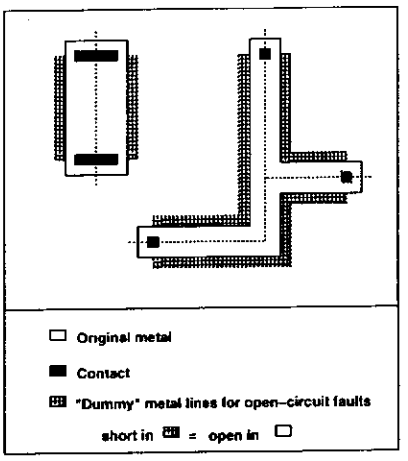
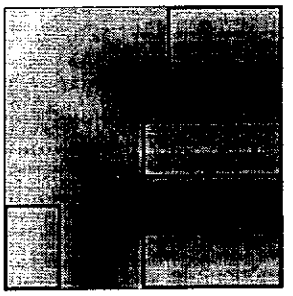


Fig. 4. Transforming an open-circuit problem to an analogous short-circuit problem.

(a)



(b)

Fig. 5. Analysis of a cell for open-circuit fault sensitivity (a) and short-circuit fault sensitivity (b).

where r_g is the minimum ground rule distance for the layer under consideration.

For example, if $r_g = 1.0$ μm and γ is selected to be 0.01 μm, then the relative error is bounded by 0.943%. In other words, the runtime needed to have the error bounded by ϵ is

$$T(\epsilon) = O\left(\frac{A}{(\epsilon \cdot r_g)^2}\right)^{1.5}. \tag{12}$$

It is implied by (12) that the error, ϵ, decreases as $1/\sqrt[3]{T}$, where T is the runtime. This implies that YMAP is a polynomial complexity approximation scheme.

III. Implementation and Results

YMAP was implemented in the C programming language, and run on an RS/6000 IBM workstation. It has been integrated

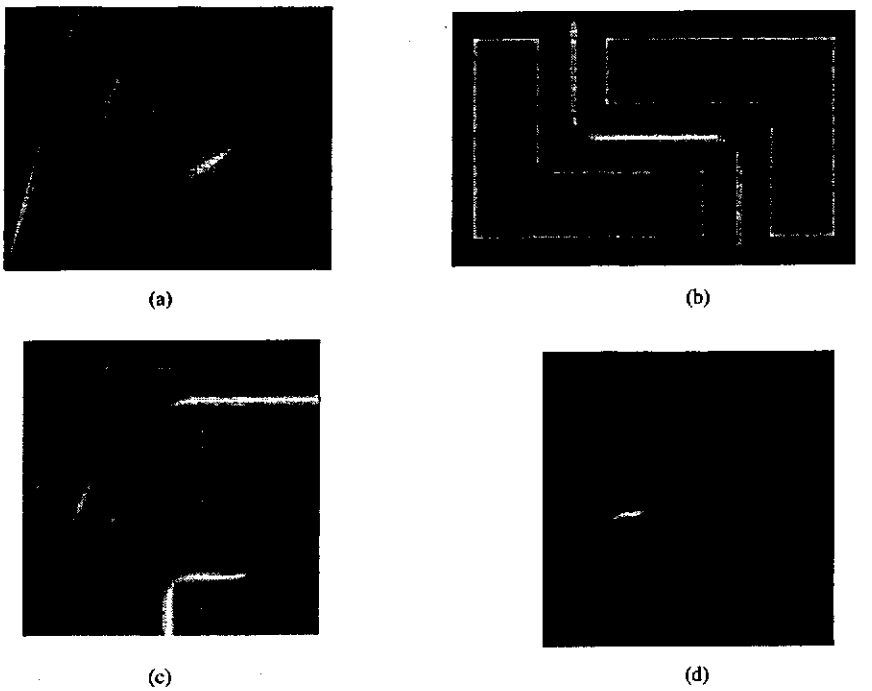


Fig. 6. Examples of sensitivity maps produced by YMAP for various layout configurations.

into the IBM design system [6]. Fig. 6 shows examples of metal polygons with the corresponding defect-sensitivity maps for short-circuit faults.

A. Usage of the YMAP Algorithm

The values calculated by the algorithm can be used in two ways:

1) *Yield-Meter*: for a given layout cell L, compute the critical area, then use (1) to evaluate the yield of the cell,
2) *Yield-Pointer*: draw a 'topographic map' of L, such that the intensity of color at point u is proportional to $S(u)$ as defined in (8) (See Fig. 6.)

If $Y(L)$ is too small, one can use the sensitivity map to find how the yield can be improved.

B. Comparison with Other Yield Estimators

We compared the YMAP program to two yield simulators: VLASIC [20], [21] and SCA [15], [16]. Simple layouts were considered for which analytic formulas of the critical area could be derived. For each such layout, we ran YMAP, VLASIC, and SCA for several time limits, and then compared the results. The time limit was set by either the number of defects (VLASIC), the step size of the defect radius (SCA), or the grid size (YMAP). The results are normalized, i.e., we take one case as having unit critical area and then normalize the other cases accordingly. This was done for two reasons. First, it is important for an interactive yield simulator tool to be able to compare several layout configurations for their predicted yield. Second, the three programs have quite different assumptions on units of area, defect size, etc., and this "numerical noise" may be filtered out by normalization.

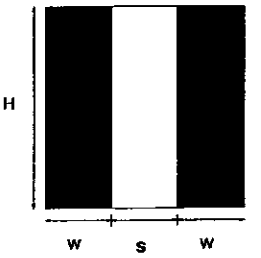


Fig. 7. Cell with two parallel lines.

1) Test Case 1: Short-Circuit Faults in Two Parallel Lines: For the case of short-circuit faults between two parallel lines (see Fig. 7) of width W, spacing S, and height H, with defect distribution as in (5) (with $p = 3$ and $q = 1$), it can be shown (see Appendix) that the critical area is

$$A_c = H \cdot \left(\frac{2}{S} - \frac{1}{W+S} - \frac{W+S/2}{R_{\max}^2} \right) \qquad (13)$$

where $(2W + S)$ and H are the horizontal and vertical dimensions of the cell, respectively, and $R_{\max}$ is the maximum possible radius of a defect for the given layout. In Table I we compare the convergence speed of the three algorithms, that is, how fast each algorithm converges to the precise ratio of $A_c(par_2)/A_c(par_1)$, where par_1, par_2 are two cells with parallel lines but different S, W combinations. The advantage of using YMAP is demonstrated in Fig. 11, where the convergence rates of the programs are compared graphically.

2) Test Case 2: Short-Circuit Faults for Two Implementations of a NAND: In CMOS design it frequently happens that several topologies can be used for the same circuit. Fig. 8 shows

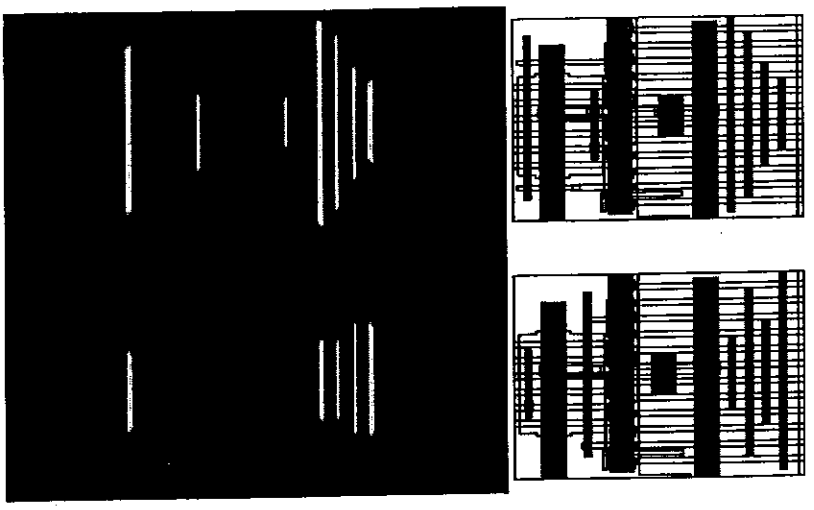


Fig. 8. Two implementations of NAND with different wiring schemes and the corresponding maps.

TABLE I
COMPARING THE AVERAGE ERRORS (IN %) OF YMAP, VLASIC AND SCA FOR THE THREE TEST CASES

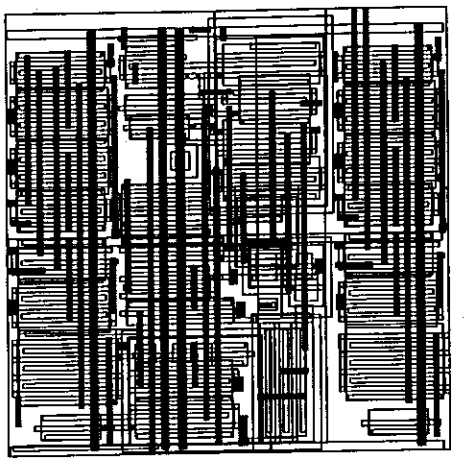
Program	Case \ CPU time (sec)	0.5	1.8	3.7	9.4	16	32	64
YMAP	PAR	5.90	3.04	1.54	0.54	0.19	0.21	0.03
	NAND	9.82	5.77	1.38	1.33	1.33	0.07	0.00
	WP	13.24	6.79	1.77	1.95	1.83	0.03	0.59
VLASIC	PAR	36.51	31.43	32.20	11.57	8.57	2.49	4.00
	NAND	100.00	11.46	12.89	6.06	4.22	5.42	0.21
	WP	52.89	33.93	10.60	6.10	1.74	5.89	1.45
SCA	PAR	84.42	51.46	27.31	23.31	17.98	13.05	0.43
	NAND	34.15	35.26	6.82	5.00	5.25	4.33	0.14
	WP	46.53	47.76	45.82	36.36	26.14	9.98	0.03

two possibilities for the wiring of a NAND gate, together with their corresponding defect-sensitivity maps. The first, *nand1*, is wired in a straightforward manner while in the second, *nand2*, the yield has been taken into account and the spacing between wires was increased as much as possible. This modification, which does not cost any additional area, has reduced the critical area in the M1-layer by more than 31%.

The error in the ratio $A_c(nand1)/A_c(nand2)$ is based on the value calculated by VLASIC in a very long run ($4 \cdot 10^7$ random defects). See Table I and Fig. 11 for the convergence rates of YMAP, VLASIC, and SCA.

3) Test Case 3: Short-Circuit Faults for Two Implementations of the WP-logic Cell: WP-logic is a large cell with 73 transistors and fairly complicated metal routing. Figs. 9 and 10 depict two possible implementations of the wiring with the corresponding defect-sensitivity maps. In the original design (*wp1*), wires have been put on a predefined grid, while in the second implementation (*wp2*) ån effort has been made to achieve better spacing, which results in a 21% reduction in critical area of the M1-layer.

Here again, the error in the ratio $A_c(wp1)/A_c(wp2)$ is based on the value calculated by VLASIC in a very long run. The convergence rates are depicted in Table I and Fig. 11.



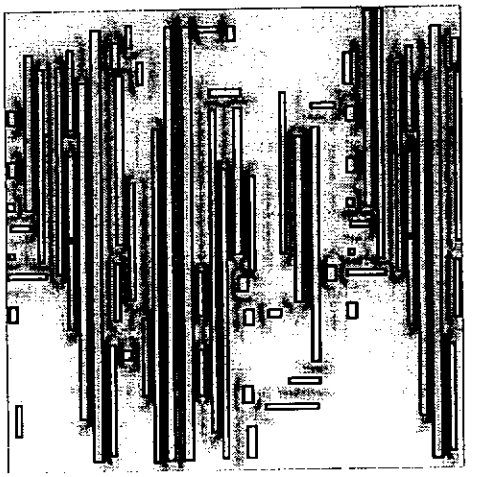


Fig. 9. The original WP-logic cell with its sensitivity map.

IV. DISCUSSION

Based on the above examples, it is clear that all three methods converge to similar results, but YMAP performs better if the computation time must be limited, since its error decreases faster. Hence, YMAP is suitable to serve as an accurate interactive design tool.

It should be noted here that, in general, (see [7]) Monte Carlo methods (like VLASIC) have their error (i.e., the vari-

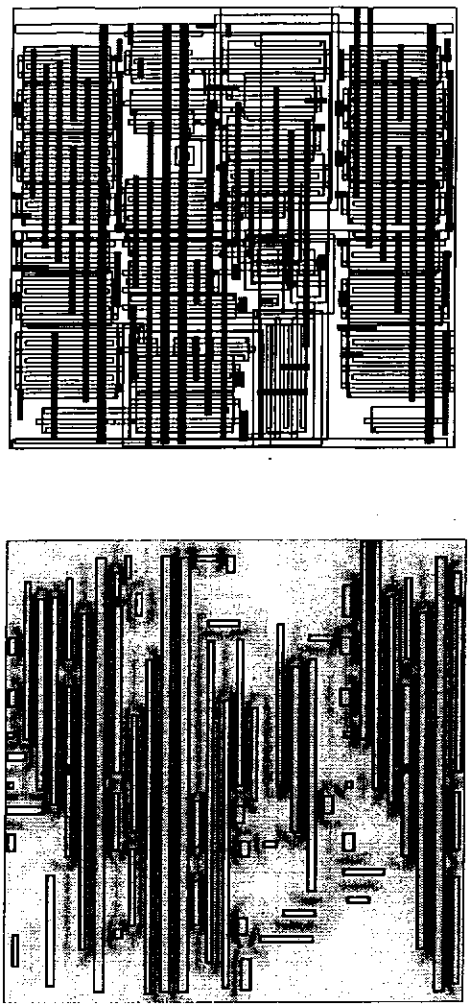


Fig. 10. The WP-logic with improved wiring cell with its sensitivity map.

ance) decrease as $1/\sqrt{N}$, where N is the number of samples, while the error in YMAP is only decreasing as $1/\sqrt[3]{N}$, where N is the number of points. However, our bound on the error (see (11) and (12)) is *deterministic* and hence, guaranteed to be satisfied while in Monte Carlo methods one may have significant deviations in the first period of computation. These random deviations also explain the nonmonotonicity in VLASIC-behavior in some of the above examples. On the other hand, the SCA algorithm has quite a good error bound, but with no error-time tradeoff, while in YMAP one can achieve any level of accuracy at the cost of additional runtime. It also turns out that if we calculate r_c only for a subset of the points, and interpolate for the others, then the time complexity reduces sharply, but the error increases only slightly. The reason for this seems to be that the function $r_c(\cdot)$ is Lipschitz continuous and hence can be approximated by a relatively small number of samples.

Besides the computational complexity, we feel that YMAP suggests a different way of thinking for a layout designer due to its good visual feedback.

Another point of interest is the generalization to 3–D defects. For such defects, the concept of *critical area* can be naturally extended to that of *critical volume* (see [14]). Currently, our program implements the algorithm for planar, single layer defects. However, it is straightforward to generalize it to 3-D objects, by using an (x, y, z) lattice rather than the planar one.

V. Conclusions

All three algorithms (VLASIC, SCA, and YMAP) eventually reach the correct results. It seems that the Monte Carlo procedure used by VLASIC requires many more iterations for the results to be precise. On the other hand, SCA and YMAP rapidly converge to their final result (YMAP's speed is somewhat better), and are deterministic, so a single run is sufficient.

Probably, the VLASIC approach is more suitable for a whole chip analysis when a long, batch type execution is acceptable. YMAP, on the other hand, can be more useful as a part of a cell/macro CAD system, when a physical design engineer needs a fast and precise yield prediction of his/her layout.

If the chip is a memory unit built from a repeated basic block, then YMAP constitutes a very efficient tool. Another advantage of YMAP is that it provides a deterministic bound on its error, so the user has a better knowledge about the precision of the results.

Appendix

A. *Proof of Theorem 1*

Assume that, as in Fig. 12, $p_1(u)$ is the metal point nearest to u, and $p_1(u')$ is the metal point nearest to u'. Clearly,

$$r_1(u) = |u - p_1(u)|, r_1(u') = |u' - p_1(u')|$$

and

$$|u' - p_1(u)| \geq r_1(u') \tag{14}$$

otherwise $p_1(u)$ would become the nearest point to v. The layout space is Euclidean, hence all points obey the triangle inequality. In particular,

$$r_1(u) < |u' - p_1(u)| - |u - u'|. \tag{15}$$

Multiplying inequality (15) by -1 and adding to inequality (14) one gets

$$r_1(u) - r_1(u') < |u - u'|. \tag{16}$$

In just the same way, by interchanging u and u', one can get

$$r_1(u') - r_1(u) < |u - u'| \tag{17}$$

and, all in all

$$|r_1(u') - r_1(u)| < |u - u'|. \tag{18}$$

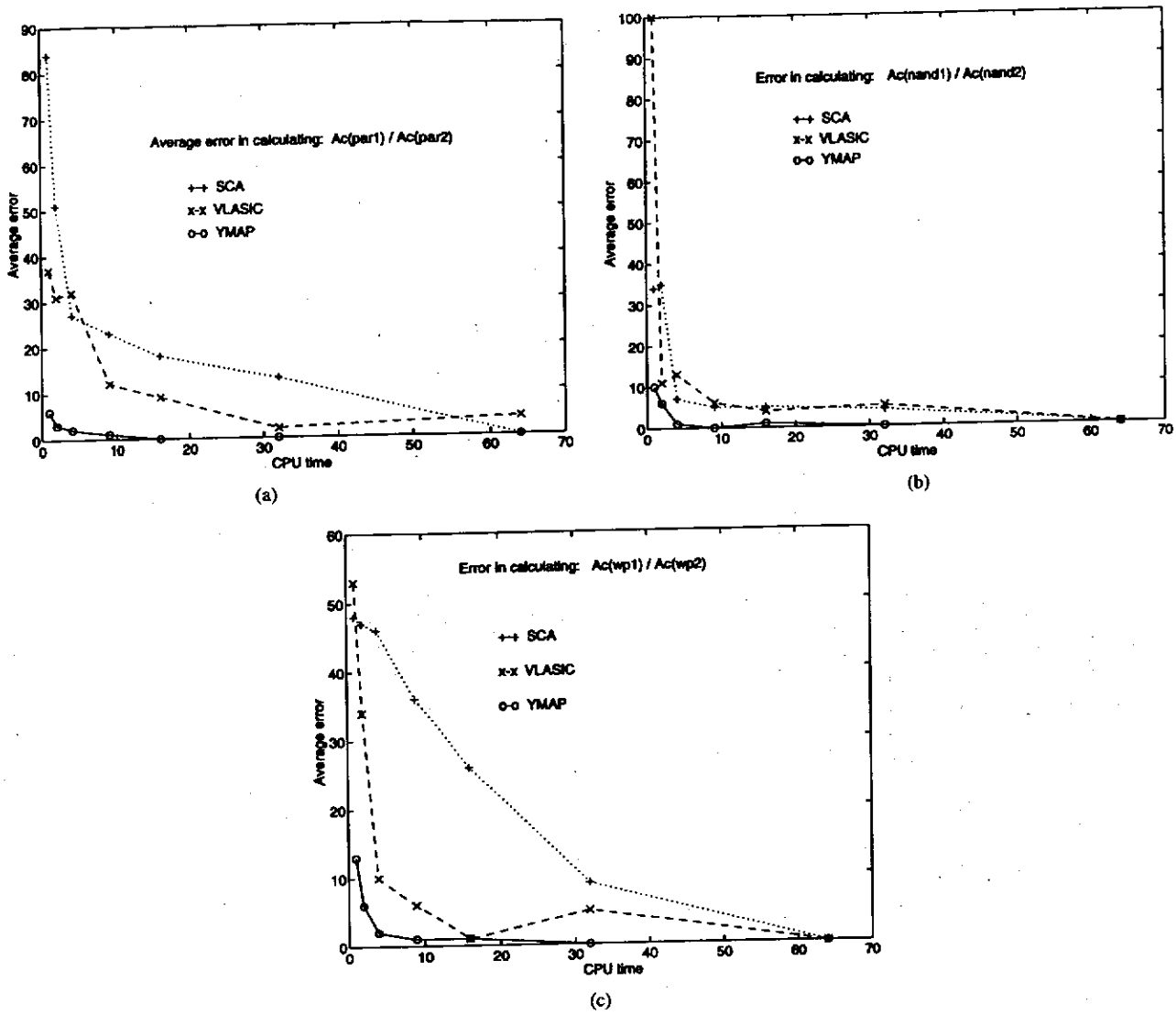


Fig. 11. Convergence rates of VLASIC, SCA, and YMAP in the three test cases.

All the above arguments apply to r_2 as well as for r_1. Hence we also have

$$|r_2(u') - r_2(u)| < |u - u'|. \tag{19}$$

Q.E.D.

As explained before, the critical radius at point u is $r_c(u) = r_2(u)$. It is implied by the theorem that any point that may affect the critical radius of point u is in one of the rings

$$\{v | r_1(u') - |u - u'| \le |v - u| \le r_1(u') + |u - u'|\}$$
$$\{v | r_2(u') - |u - u'| \le |v - u| \le r_2(u') + |u - u'|\}$$

as sketched in Fig. 13. Hence, Corollary 1 follows.

Another corollary of Theorem 1 is that r_c has a bounded derivative:

Corollary 2: Denote an arbitrary unit vector by p. Then

$$\forall u : \left| \frac{d}{dp} r_c(u) \right| \le 1.$$

Proof: By definition

$$\frac{d}{dp} r_c(u) = \lim_{\Delta p \to 0} \frac{r_c(u + \Delta p) - r_c(u)}{\Delta p}.$$

Using Theorem 1 we know that

$$\forall u : |r_c(u) - r_c(u')| \le |u - u'|.$$

By setting u' equal to $u + \Delta p$ we complete the proof.

Q.E.D.

B. Proof of Theorem 2

Using Theorem 1, one can see that the value of $r_c(u)$ as computed by YMAP in any nongrid point u cannot exceed its value at the nearest grid-point u_g by more than the distance between them. The worst case is when $r_c(u)$ is the same for all four vertices of a grid square, but takes a maximum in the center of the square. The average error for this square is the volume of this square-based pyramid, divided by the area of

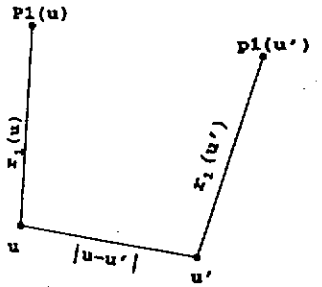


Fig. 12. A sketch of the situation in Theorem 1.

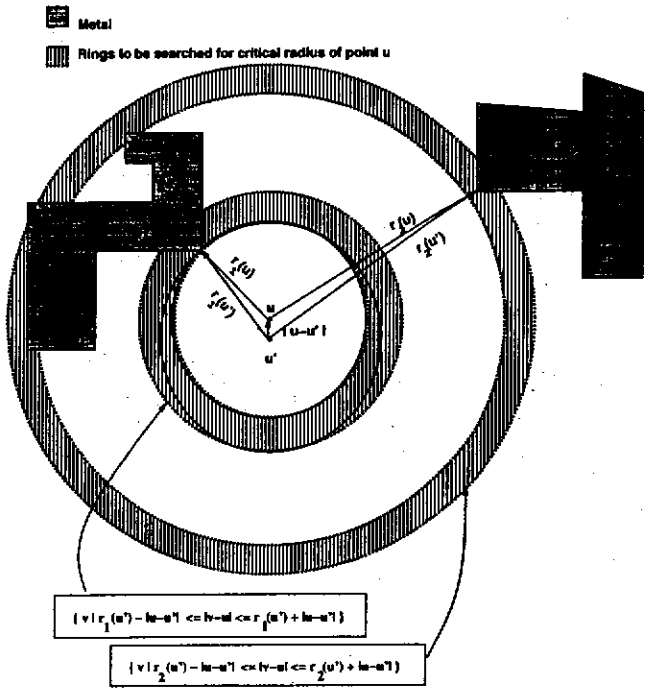


Fig. 13. Only two rings need to be searched for $r_c(u')$.

its basis. Due to corollary 2 the steepness of this pyramid is bounded by 45°. Hence, the worst-case relative error in $r_c(u)$ within a grid-square is bounded by

$$\mathrm{Err}[r_c(u)] = \left|\frac{r_c(u_g) - r_c(u)}{r_c(u)}\right| \le \frac{\frac{1}{3}\gamma^2\frac{\gamma}{\sqrt{2}}}{\gamma^2} = \frac{\gamma}{3\sqrt{2}}.$$

Since the local defect sensitivity is defined by

$$S(u) = \int_{r=r_c(u)}^{\infty} D(r)\,dr$$

the absolute error in the calculated value $S'(u)$ is bounded from above by

$$|S'(u) - S(u)| \le \int_{r_c(u)}^{r_c(u)+(\gamma/3\sqrt{2})} D(r)\,dr.$$

For practical considerations we can assume that the layout obeys the ground rules, hence $r_c(u) > r_0$ and then we obtain from (5) that

$$D(r) = cr_0^{p-1}/r^p.$$

Denote $\int_r^{\infty} D(r)\,dr$ by $I(r)$. Then, in the region of interest

$$I(r) = \frac{cr_0^{p-1}}{(p-1)r^{p-1}}.$$

The relative error, which is the ratio between the error and the precise value, is

$$\begin{aligned}\mathrm{Err}[S(u)] &= \left|\frac{S'(u) - S(u)}{S(u)}\right| \\ &\le \frac{I(r) - I\left(r + \frac{\gamma}{3\sqrt{2}}\right)}{I(r)} \\ &= 1 - \frac{I\left(r + \frac{\gamma}{3\sqrt{2}}\right)}{I(r)} \\ &= 1 - \frac{\frac{cr_0^{p-1}}{(p-1)\left(r + \frac{\gamma}{3\sqrt{2}}\right)^{p-1}}}{\frac{cr_0^{p-1}}{(p-1)r^{p-1}}} \\ &= 1 - \left(\frac{r}{r + \frac{\gamma}{3\sqrt{2}}}\right)^{p-1} \\ &\le \frac{(p-1)\gamma}{3\sqrt{2}r}\end{aligned}$$

Substituting a typical value of $p = 3$, and using the fact that the critical radius of any point cannot be smaller than half the minimal ground-rule, we get

$$\mathrm{Err}[A_c'] = \left|\frac{A_c' - A_c}{A_c}\right| \le \max_u\{\mathrm{Err}[S'(u)]\} \le \frac{0.943\gamma}{r_g} \tag{20}$$

where r_g is the minimal distance between shapes at the layer under consideration.

Q.E.D..

C. Critical Area for Two Parallel Lines

Consider the case of two parallel lines with width W, separation S, and height H (see Fig. 7). It has been shown (e.g., [4]) that the critical area for radius r is

$$A_c(r) = \begin{cases} 0, & \text{if } r \le S/2 \\ H \cdot (2r - S), & \text{if } S/2 < r \le S + W \\ H \cdot (2W + S), & \text{if } S + W < r \le R_{\max} \end{cases} \tag{21}$$

where $(2W + S)$, H are the horizontal and vertical dimensions of the cell, respectively. $R_{\max}$ is the maximal possible radius

of a defect for the given layout. The total critical area is

$$\begin{aligned} A_c &= \int_{S/2}^{R_{\max}} A_c(r) c r^{-3}\, dr \\ &= cH\left[\int_{S/2}^{S+W} (2r^{-2} - Sr^{-3})\, dr \right. \\ &\quad \left. + \int_{S+W}^{R_{\max}} (2W+S) r^{-3}\, dr\right] \\ &= cH\left[\left(-2r^{-1} + \frac{s}{2}r^{-2}\right)\Big|_{s/2}^{2w+s} \right. \\ &\quad \left. + (2w+s)\left(-\frac{1}{2}r^{-2}\Big|_{2W+S}^{R_{\max}}\right)\right] \end{aligned}$$

where

$$c = (q+1)(p-1)/(q+p).$$

For $p = 3$ and $q = 1$, c equals 1 and as a result

$$A_c = H \cdot \left(\frac{2}{S} - \frac{1}{W+S} - \frac{W+S/2}{R_{\max}^2}\right). \qquad (22)$$

ACKNOWLEDGMENT

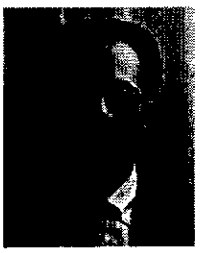
The authors would like to thank the anonymous referees for their enlightening comments, which helped to improve the performance of the YMAP program and the presentation of this paper.

REFERENCES

[1] C. M. Bender and S. A. Orszag, *Advanced Mathematical Methods for Scientists and Engineers*. New York: McGraw-Hill, 1978.
[2] A. R. Dalal, P. D. Franzon, and M. J. Lorenzetti, "A layout-driven yield predictor and fault generator for VLSI," *IEEE Trans. Semicond. Manufact.*, vol. 6, no. 1 pp. 77–82, 1993.
[3] A. V. Ferris-Prabhu, "Defect size variations and their effect on the critical area of VLSI devices," *IEEE J. Solid-State Circuits*, vol. SC-20, pp. 878–880, 1985.
[4] ——, *Introduction to Semiconductor Device Yield Modeling*. Norwood, MA: Artech House, 1992.
[5] S. Gandemer, B. C. Tremintin, and J. J. Charlot, "Critical area and critical levels calculation in IC yield modeling," *IEEE Trans. Electronic Devices*, vol. 35, no. 2, pp. 158–166, Feb. 1988.
[6] *IBMS—Integrated Book and Macro System*. IBM Science and Technology, Haifa, Israel.
[7] R. M. Karp, M. Luby, and N. Madras, "Monte-Carlo approximation algorithms for enumeration problems," *J. Algorithms*, pp. 429–448, 1989.
[8] I. Koren, "The effect of scaling on the yield of VLSI circuits," in *Yield Modeling and Defect Tolerance in VLSI Circuits*, W. R. Moore, W. Maly and A. Strojwas Eds., Bristol, UK: Adam Hilger Ltd., 1988, pp. 91–99.
[9] I. Koren and A. D. Singh, "Fault tolerance in VLSI circuits," *IEEE Comput. Mag.*, pp. 73–83, July 1990.
[10] I. Koren and C. H. Stapper, "Yield models for defect tolerant VLSI circuits: A review," in *Defect and Fault Tolerance in VLSI Systems*, I. Koren, Ed., New York: Plenum, 1989, pp. 1–21.
[11] S. Lefschetz, *Differential Equations: Geometric Theory*. New York: Interscience Publishers, 1957.
[12] W. Maly, "Modeling of lithography related yield losses for CAD of VLSI circuits," *IEEE Trans. Computer-Aided Design*, vol. CAD-4, no. 3, pp. 166–177, July 1985.
[13] ——, "Computer aided design for VLSI circuit manufacturability," *Proc. IEEE*, pp. 356–392, Feb. 1990.
[14] J. Pineda de Gyvez and S. M. Dani, "Modeling of 3-dimensional defects in integrated circuits," in *Proc. IEEE Int. Workshop on Defect and Fault Tolerance in VLSI Systems*, 1992, pp. 197–206.
[15] ——, "IC defect sensitivity for footprint type spot defects," *IEEE Trans. Computer-Aided Design*, vol. 11, no. 5, pp. 638–658, May 1992.
[16] J. Pineda de Gyvez, *SCA—A Defect-Sensitivity Evaluation System for IC Layout*. Texas: Texas A&M University, 1992.
[17] M. Rivier, "Random yield simulation applied to physical circuit design," in *Yield Modeling and Defect Tolerance in VLSI Circuits*, W. R. Moore, W. Maly, and A. Strojwas, Eds., Bristol, UK: Adam Hilger Ltd., 1988, pp. 111–120.
[18] C. H. Stapper, "Modeling of defects in integrated circuits photolithographic patterns," *IBM J. Res. Develop.*, vol. 28, no. 4, pp. 461–475, 1984.
[19] I. A. Wagner and I. Koren, "An interactive yield estimator as a VLSI CAD tool," in *Proc. IEEE Int Workshop Defect and Fault Tolerance in VLSI Systems*, 1993, pp. 167–174.
[20] D. M. H. Walker, "VLASIC system user manual, release 1.3," CMU, Aug. 1990.
[21] H. Walker and S. W. Director, "VLASIC: A catastrophic fault yield simulator for integrated circuits," *IEEE Trans. Computer-Aided Design*, vol. CAD-5, no. 4, pp. 541–556, Oct. 1986.

Israel A. Wagner received the B.Sc. degree in computer engineering from the Technion-Israel Institute of Technology, Haifa, Israel, in 1987, and the M.Sc. degree in computer science from the Hebrew University, Jerusalem, Israel, in 1990.

In 1990 he joined the IBM Haifa Reasearch Group, where he has been staff member since 1991. He was also a teaching assistant at the Hebrew University and a research engineer at General Microwave, Jerusalem, Israel. His research interests include manual and automatic VLSI design, computational geometry, and models for yield and performance of VLSI chips.



Israel Koren (S'72–M'76–SM'87–F'91) received the B.Sc., M.Sc., and D.Sc. degrees from the Technion-Israel Institute of Technology, Haifa, in 1967, 1970, and 1975, respectively, all in electrical engineering.

He is currently a Professor of Electrical and Computer Engineering at the University of Massachusetts, Amherst. Previously he was with the Departments of Electrical Engineering and Computer Science at the Technion-Israel Institute of Technology. He also held visiting positions with the University of California at Berkeley, University of Southern California, Los Angeles, and University of California, Santa Barbara. He has been a consultant to several companies including Intel, Digital Equipment Corp., National Semiconductor and Tolerant Systems. His current research interests are fault-tolerant VLSI architectures, models for yield and performance, floor-planning of VLSI chips, and computer arithmetic. He published extensively in the IEEE TRANSACTIONS. He was a Co-Guest Editor for the IEEE TRANSACTIONS ON COMPUTERS, special issue on High Yield VLSI Systems, April 1989. Since January 1992 he has served on the Editorial Board of these TRANSACTIONS. He also served as Program Committee member for numerous conferences. He has edited and co-authored the book, *Defect and Fault-Tolerance in VLSI Systems*, vol. 1, Plenum, 1989. He is the author of the textbook *Computer Arithmetic Algorithms*, (Englewood Cliffs, NJ: Prentice-Hall, 1993).