APIC'95, El Paso, Extended Abstracts, A Supplement to the international journal of Reliable Computing

How To Find Input Variables Whose Influence On The Result Is The Largest, or, How To Detect Defective Stages In VLSI Manufacturing?

Maria Beltran and Vladik Kreinovich

***Abstract*— Manufacturing processes are never absolutely accurate. The manufacturing parameters deviate from the desired values; as a result, the characteristics of the manufactured product may deviate from the values that we would have gotten for the ideal manufacturing. Interval methods can help in solving the *direct problem*: we know the possible deviations of the manufacturing process, and we want to know the resulting deviations in the product's characteristics. If these deviations are too huge, then we have an *inverse problem*: to find the parameters of the manufacturing process that need to be improved. In principle, this problem can be solved by computing the contribution of each manufacturing parameter. However, if there are many parameters (as in VLSI manufacturing), then this algorithm takes too long. In this paper, we propose a faster algorithm for detecting defective stages.**

I. Formulation of the Problem

A. A Manufacturing Product Must Satisfy Certain Conditions

In many real-life applications, we must guarantee that the values of some characteristics lie in between the given bounds. For example, we must guarantee that for several electrical, mechanical, and other characteristics of a VLSI circuit, the value y of each characteristic lies within the given bounds y_0^- and y_0^+.

B. We Must Design A Manufacturing Process That Will Guarantee The Desired Conditions

After a circuit is produced, the desired characteristics can be directly measured. If the values of some of these characteristics are outside the desired limits, then this circuit must be discarded. If the manufacturing process produces many defective chips, then we discard many of them, and, as a result, the cost of manufacturing a single good chip skyrockets.

It is therefore desirable to design a manufacturing process in such a way that the desired characteristics will be guaranteed to lie within the given limits.

C. If A Manufacturing Process Is Given, We Must Check Whether It Guarantees The Desired Conditions

Each characteristic y of the chip is uniquely determined by the values of the parameters $x_1, ..., x_n$ that describe different stages of the manufacturing procedure. If we know the exact values of these parameters x_i, then we can compute y by applying some algorithm f: $y = f(x_1, ..., x_n)$. For example, the resulting resistance of the chip can be computed based on the thicknesses of the wires and the resistances of the components. For VLSI manufacturing, algorithms that describe the yield of a chip as a function of the manufacturing parameters such as oxide thickness, placement, etc, are given in [1, 3].

Manufacturing is never a precise process. As a result, for each of the parameters x_i, we cannot guarantee its exact value; we can only guarantee that x_i belongs to the interval $\mathbf{x}_i = [\tilde{x}_i - \Delta_i, \tilde{x}_i + \Delta_i]$, where $\tilde{x}_i$ is a *nominal* value, and Δ_i is the accuracy of this particular manufacturing process.

If we have chosen a candidate for the manufacturing process (i.e, if we have already chosen the values $\tilde{x}_i$ and Δ_i), then the only remaining question is: Does this process guarantee that $y = f(x_1, ..., x_n)$ belong to the interval $[y_0^-, y_0^+]$? In mathematical terms, this question is equivalent to the following one: is it true that $[y^-, y^+] \subseteq [y_0^-, y_0^+]$, where by $[y^-, y^+]$, we denoted the interval

$$[y^-, y^+] = f(\mathbf{x}_1, ..., \mathbf{x}_n) =$$

$$\{f(x_1, ..., x_n) \,|\, x_1 \in \mathbf{x}_1, ..., x_n \in \mathbf{x}_n\}. \qquad (1)$$

Computing this interval (1) is what *interval computations* is about. Therefore, in order to check whether the manufacturing process guarantees the desired conditions, we can do the following:

The authors are with Computer Science Department, The University of Texas at El Paso, El Paso, TX 79968, email mbeltran@cs.utep.edu and vladik@cs.utep.edu. This work was partially supported by NSF grant No. CDA-9015006 and NASA Research Grant No. 9-757. The authors are thankful to Arnold Neumaier for valuable discussions.

- compute the interval (1);
- check whether $[y^-, y^+] \subseteq [y_0^-, y_0^+]$ (i.e., whether $y_0^- \le y^-$ and $y^+ \le y_0^+$).

D. What If We Are Not Satisfied With The Given Manufacturing Process?

If $[y^-, y^+] \subseteq [y_0^-, y_0^+]$, then the manufacturing process guarantees the desired properties, and it should be adapted.

But what if $[y^-, y^+] \not\subseteq [y_0^-, y_0^+]$? In this case, we must make change some stages of the manufacturing procedures to more accurate ones. We cannot make all stages more accurate, because more accurate stages usually cost much more. Therefore, we must find the stages that must be improved.

If the original manufacturing process has been designed by an experienced expert, then usually, the accuracies of the manufacturing stages are well within the necessary limits. The reason why, in spite of that, the overall accuracy of the manufacturing process is not satisfactory, is that the intuition (and heuristic ideas) of human experts are not perfect, and therefore, on some of the stages (one, two, usually a few), an expert might have been too lax.

As a result, we arrive at the following situation: we know that out of n parameters $x_1, ..., x_n$, there are a few of them whose influence on the resulting accuracy is too huge (e.g., they contribute $\ge 10\%$ of the resulting inaccuracy). Our goal is to find these parameters, and to improve the corresponding manufacturing stages.

E. The Problem Can Be Simplified If We Take Into Consideration That Errors Are Usually Small

The values y^+ and y^- are the maximum and the minimum of the expression

$$f(x_1, ..., x_n) = f(\tilde{x}_1 + \Delta x_1, ..., \tilde{x}_n + \Delta x_n),$$

where by $\Delta x_i = x_i - \Delta_i$, we denoted the difference between the actual value x_i of i–th parameter and its nominal value $\tilde{x}_i$ (i.e., the *manufacturing uncertainty* that correspond to i–th parameter). We know that

$$|\Delta x_i| \le \Delta_i \qquad (2).$$

Usually, the inaccuracies Δ_i that correspond to different parameters are relatively small ($\Delta_i \ll |\tilde{x}_i|$). Therefore, we can neglect quadratic and higher order terms in the expansion of the function

$$f(\tilde{x}_1 + \Delta x_1, ..., \tilde{x}_n + \Delta x_n),$$

and arrive at the expression

$$f(x_1, ..., x_n) = \tilde{y} + \sum_{i=1}^{n} f_{,i} \Delta x_i, \qquad (3)$$

where:

- by $\tilde{y} = f(\tilde{x}_1, ..., \tilde{x}_n)$, we denoted the *nominal* value of y (i.e., the value that corresponds to the ideal case when all the parameters x_i take the nominal values), and
- $f_{,i}$ denotes a partial derivative:

$$f_{,i} = \frac{\partial f}{\partial x_i}(\tilde{x}_1, ..., \tilde{x}_n).$$

For a linear function (3), maximum and minimum under the condition (2) can easily be computed: the result is:

$$y^{\pm} = \tilde{y} \pm \Delta,$$

where

$$\Delta = \sum_{i=1}^{n} |f_{,i}| \Delta_i. \qquad (4)$$

In this formulation, we have clearly separated the influence of each variable x_i, and so, we can reformulate this problem as follows: find i for which

$$|f_{,i}| \Delta_i \ge p \cdot \Delta, \qquad (5)$$

where p is a given number (e.g., $p = 0.1 = 10\%$).

How can we find such i?

F. If We Already Know The Partial Derivatives Of f, Then This Problem Is Easy To Solve

If for computing Δ, we use a numerical method that already computes the values of the partial derivatives, then it is very easy to check whether the condition (5) is true for each of these derivatives, and thus find all the values i for which the corresponding manufacturing stage must be improved.

G. The Situation Is Different If We Use Methods That Are Not Based On Estimating Derivatives

If a method of estimating Δ is based on computing all n partial derivatives, then this method requires at least n computation steps. In many real-life cases (e.g., in VLSI design), the number of parameters n is in the thousands. In this case, methods that require computing partial derivatives take too long. For such situations, other methods have been proposed that give an estimate for Δ mush faster, without computing all the derivatives. In particular, a method based on Monte-Carlo simulation is described in [2].

If we use this method to estimate Δ, then we do not know the values $f_{,i}$. If we directly try to check condition (5) for $i = 1, ..., n$, then we will need to actually compute all n derivatives: this is exactly what we wanted to avoid. So, we arrive at the following problem:

H. Formulation Of The Problem

To Find Input Variables x_i Whose Influence on the Result is the Largest (Without Computing All n Partial Derivatives).

When computing, we can assume that the function f is linear (in the vicinity of the point $(\tilde{x}_1, ..., \tilde{x}_n)$), but the coefficients $f_{,i}$ of this linear function are unknown.

In this paper, we propose such an algorithm.

II. Proposed Algorithm

A. Main Idea

The main idea of this algorithm is as follows:

- First, we choose a number C, and divide the parameters $x_1, ..., x_n$ into C approximately equal groups $g_1, ..., g_C$. To be more precise: we compute the group size $s = \lceil (n/C) \rceil$, and we assign a value i to $k-$th group if $(k-1)s < i \leq ks$.
- For each of these groups, we compute the value $\Delta(g_k)$ that corresponds to the situation when only parameters from this group can be imprecise, and all the other are assumed to be absolutely accurate. In other words, to compute $\Delta(g_k)$, we fix the values of the parameters $x_i = \tilde{x}_i$ for $i \notin g_k$, and let parameters x_i with $i \in g_k$ to take arbitrary values from the interval $\mathbf{x}_i$. So, we estimate the interval

  $$f(\tilde{x}_1, ..., \tilde{x}_{s(k-1)}, \mathbf{x}_{s(k-1)+1}, ..., \mathbf{x}_{sk}, \tilde{x}_{sk+1}, ...).$$

  This can be done, e.g., by applying a method from [2].
- The resulting values satisfy the formula

  $$\Delta(g_k) = \sum_{i \in g_k} |f_{,i}| \Delta_i. \qquad (6)$$

  Since each element i belongs to exactly one group, we can compare (6) with (4) and conclude that

  $$\Delta = \sum_{k=1}^{C} \Delta(g_k). \qquad (7)$$

  If $k-$th group contains the value i that satisfies the inequality (5), then we have

  $$\Delta(g_k) \geq |f_{,i}| \Delta_i \geq p \cdot \Delta.$$

  Because of this result, we will call groups that satisfy this inequality *suspicious* (in the sense that these groups may contain parameters that satisfy (5)). Because of (7), there can be no more than $1/p$ terms $\Delta(g_k)$ that are $\geq p \cdot \Delta$ (else, the sum of the values $\Delta(g_k)$ would exceed Δ). Therefore, out of the original C groups, only $\geq 1/p$ can contain bad parameters (i.e., parameters for which (5) is true). Each group contains $s \approx n/C$ paratemers. As a result, we now have to consider only the parameters from these suspicious groups. The total number n_1 of these parameters is $\leq (1/p) \cdot s \approx (1/p)(n/C) = n/(Cp)$. If $Cp > 1$, then we have $n_1 < n$.
- We started with n suspicious parameters, and we were able to select n_1 of them. Now, we can start the same procedure with n_1 suspicious parameters: divide them into groups, compute $\Delta(g_k)$ for each of the new groups, and keep as suspicious only those parameters that belong to suspicious groups. As a result, instead of n_1 originally suspicious parameters, we have $n_2 \leq n_1/(Cp)$ ones. Since $n_1 \leq n/(Cp)$, we have $n_2 \leq n/(Cp)^2$.
- After I iterations of this procedure, we get $\leq n/(Cp)^I$ suspicious parameters. So, if $n/(Cp)^I \leq 1$, we will find all "bad" parameters.

These iterations continue until we get $s = 1$. At this point, we have $\leq (1/p)$ suspicious elements. For a one-element group $g = \{i\}$, $\Delta(g) = |f_{,i}|\Delta_i$. Therefore, if a one-element group is suspicious it means that its only element satisfies the desired inequality (5).

The required number of iterations is thus determined from the condition $n/(Cp)^I = 1$, i.e., $n = (Cp)^I$, and $I = \ln(n)/\ln(Cp)$. On each iteration, we need to find $\Delta(g_k)$ for C different groups. Therefore, we need C applications of the error-estimation program. Totally, we thus need

$$I \cdot C = C \cdot \frac{\ln(n)}{\ln(Cp)}$$

applications of this program.

Our main goal is to save computation time. So, we will choose the parameter C of this algorithm from the condition that

$$I \cdot C = C \cdot \frac{\ln(n)}{\ln(Cp)} \to \min.$$

If we differentiate the minimized function w.r.t. C and equate the derivative to 0, we will end up with the value $\ln(Cp) = 1$, i.e., $Cp = e$ and $C = e/p$. For this C, we need

$$C \cdot \frac{\ln(n)}{\ln(Cp)} = C \cdot \ln(n) = \frac{e}{p} \ln(n)$$

applications of the error estimation program.

The value $C = e/p$ is not necessarily integer, so, a reasonable idea is to choose as C the integer that is the closest to e/p.

As a result, we arrive at the following algorithm:

III. Resulting Algorithm

A. Input

(1) An integer n (called *the number of parameters*).
(2) n pairs of real numbers $(\tilde{x}_i, \Delta_i)$; $1 \le i \le n$; the value $\tilde{x}_i$ is called a *nominal value*, and Δ_i is called the *tolerance* of i–th parameter.
(3) A program f that computes $y = f(x_1, ..., x_n)$, given n real numbers $x_1, ..., x_n$; we assume that this function is linear (i.e., is described by the formula (3)), but the actual values of the coefficients $f_{,i}$ are unknown. This program is called the *data processing algorithm*.
(4) a program which given:
   - an integer n;
   - n pairs of real numbers $(\tilde{x}_i, \Delta_i)$;
   - a (linear) algorithm $f(x_1, ..., x_n)$,

   returns Δ (determined by the formula (4)); this program is called the *error estimating algorithm*.

B. Objective

To find all i for which the inequality (5) is true.

C. Main Result

We can find all such i by applying the error estimating algorithm $\le C \ln(n)$ times.

D. Algorithm

First, we do some pre-computing: we compute C as an integer that is the closest to e/p.

Then, we apply some iterations. This algorithm we will work with an (ordered) list L of indices of suspicious parameters x_i. In the computer, the list can be stored by describing its first, its second, ..., elements, and the total number of elements in this list.

In the beginning, this list includes all i from 1 to n. The algorithm consists of several iterations, on each of which the list will shrink, until we only have the desired elements in the list. This shrinking is done as follows:

- Divide the list L into C groups $g_1, ..., g_C$. To do that, we compute $s = \lceil (n/C) \rceil$. Then, we assign the first s elements of the list L to the first group g_1, the next s elements to the second group, etc, and, finally, we assign the remaining elements to the group g_C.
- For each group g_k, we apply the given error estimating algorithm to a function of s variables $x_i, i \in g_k$, that is obtained from f if we only vary parameters from the group g_k. Thus, we compute $\Delta(g_k)$.
- Elements of the groups g_k for which $\Delta(g_k) < p\Delta$ are deleted from the list.
- With the remaining list, the procedure continues.

The values of s becomes smaller and smaller with each iteration. The iteration for which $s = 1$, is the last one; after this iteration, we stop. The resulting list of suspicious indices now contains exactly the numbers of the "bad" parameters, i.e., parameters for which (5) is true.

References

[1] S. W. Director, K. Krishna, and P. Feldmann, "Introduction to Parametric Yield Optimization," In: S. W. Director and W. Maly (eds.), **Statistical Approach to VLSI**, North Holland, N.Y., 1994, pp. 103–110.

[2] V. Kreinovich, A. Bernat, E. Villa, and Y. Mariscal, "Parallel computers. estimate errors caused by imprecise data," *Interval Computations*, 1991, No. 2, pp. 41–42.

[3] W. Maly, "Design for Manufacturability of VLSI Circuits," In: S. W. Director and W. Maly (eds.), **Statistical Approach to VLSI**, North Holland, N.Y., 1994, pp. 81–82.