Short Papers

Grid Quality and Its Influence on Accuracy and Convergence in Device Simulation

V. Axelrad

Abstract— **Convergence problems can significantly limit the practical value of numerical device simulation, especially where the ability to obtain stable results for a wide range of process conditions is crucial. Solving the semiconductor device equations is difficult for several reasons: solutions exhibit extremely rapid spatial variations in thin boundary layers at pn-junctions and inversion/depletion layers, equations are strongly nonlinear, a unique steady-state solution may not exist for a given set of bias conditions, loss of accuracy is possible when evaluating physical model equations. At the same time, the solution can be quite sensitive to the boundary conditions, making a certain level of accuracy necessary to ensure convergence of the nonlinear iteration.**

As a result, convergence problems are known to exist. Remedies for these problems are, if at all, usually found heuristically, rarely understanding the reasons for the problem or why a particular approach works better than others.

In this paper, an analysis of the situation is presented and an example demonstrates how improving grid quality can help solve convergence problems. A tradeoff between reducing discretization error and improving convergence and stability is demonstrated. The approach is justified on the grounds of basic finite element theory and demonstrated using a practical application (power MOS breakdown simulation).

Index Terms—**Accuracy, breakdown, convergence, device design, grid generation, numerical analysis, power semiconductor devices, semiconductor device simulation.**

I. Introduction

Convergence problems are a well-known yet little understood phenomenon in device simulation. Experienced users of device simulators know that certain device structures under certain bias conditions cause problems. Workarounds are known in some cases, although sometimes nothing seems to help.

Breakdown simulation is a particularly challenging problem in device simulation. Typically, a complex interplay of strongly coupled multidimensional electrical and thermal effects takes place leading to rapid changes in internal device behavior with changing bias conditions. This physically based sensitivity of the solution is often reflected by accuracy and convergence problems when a numerical solution is attempted. In effect, breakdown simulations often heighten the requirements to accuracy and numerical stability of the utilized numerical algorithms.

In this work, the situation is analyzed using a power MOS breakdown simulation. The structure and initial device grid were generated using two-dimensional (2-D) numerical simulation of an industrial process with the TSUPREM-4 process simulation program [1]. A straightforward attempt at a numerical solution results in poor convergence and consequently excessive simulation time.

Second breakdown of a power MOS is dominated by a rapid decrease in material resistivity at high current levels caused by self heating. This is a strongly nonlinear effect characterized by current snap-back and very high lattice temperatures approaching the eutectic temperature of contact material (aluminum) and therefore device failure.

We were able to dramatically improve the convergence and central processing unit (CPU) time of the solution by replacing the original simulation grid by one of better quality. There is an important tradeoff between solution accuracy measured by the discretization error and convergence and stability measured by the conditioning of the discretized equations. While the discretization error is improved (reduced) by refining the grid, the conditioning of the problem degrades. An estimate for the condition number appears to correlate well with convergence on different simulation grids in the same physical application. The same approach has been successfully applied to a number of other difficult applications and appears to be of general value.

In the following, an overview of the role of accuracy and conditioning in device simulation with an emphasis on grid effects is presented. A power MOS breakdown with a discussion of the physical effects taking place and their influence on the numerical solution is shown as an example.

II. Accuracy and Conditioning in Device Simulation

Steady-state numerical device simulation requires the solution of a coupled nonlinear system of second-order elliptic partial differential equations. A comprehensive review of the equations solved and common discretization techniques can be found in [2]. The differential equations are discretized in space leading to the need to solve a large nonlinear algebraic system of equations

$$F(x) = 0. \tag{1}$$

This equation is solved iteratively using Newton's method of successive linearization or modifications thereof

$$\left(\frac{\partial F}{\partial x}\right) \cdot \Delta x = -F(x_0), \quad \text{where } \Delta x = (x - x_0) \tag{2}$$

or

$$J \cdot \Delta x = -y_0. \tag{3}$$

A single Newton step (2) involves the calculation of a solution update Δx given a solution estimate x_0. During this process, accuracy can be lost in two places:

1) when calculating the right-hand sides $F(x_0) = u_0$;
2) when solving the linear system of equations $J \cdot \Delta x = -y_0$.

The propagation of numerical error during calculations is described by the condition number of the algorithm. The condition number for a function $y(x)$ with respect to x is defined as the amplification factor from the relative change in x to the relative change in y

$$C_x^y = \left(\frac{dy}{y}\right) \Big/ \left(\frac{dx}{x}\right) = \frac{x}{y} \cdot \frac{dy}{dx}. \tag{4}$$

The accuracy of the solution update can thus be estimated to

$$\varepsilon_{\Delta x} = C^J \cdot C^F \cdot \varepsilon_{x_0} \tag{5}$$

Manuscript received January 11, 1996; revised May 6, 1996. This paper was recommended by Associate Editor Z. Yu.

The author was with Technology Modeling Associates, Inc., Sunnyvale, CA USA. He is now with Sequoia Design Systems, Woodside, CA 94062 USA (e-mail: sequoia_design_systems@yahoo.com).

Publisher Item Identifier S 0278-0070(98)02549-4.

TABLE I
TYPICAL RANGES OF CONDITION NUMBER COMPONENTS IN TWO-DIMENSIONAL DEVICE SIMULATION

Jacobian condition number C_J	Right-hand-sides condition number C_F	Machine accuracy ε_0
$10^4 - 10^{10}$	$10^1 - 10^4$	10^{-16} (double precision)

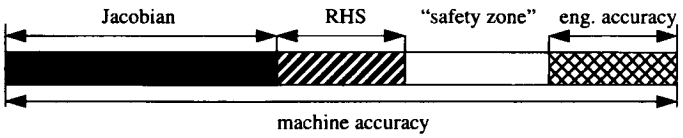

Fig. 1. Schematic representation of conditioning effects (log scale).

where C^J is the condition number of the Jacobian matrix, C^F is the condition number of the evaluation of the residuals, and ε_{x_0} is the so-called machine accuracy, that is the error inherent in the computer representation of a number. Having estimates for the three values on the right-hand side of (5) allows us to estimate the accuracy which can be at best expected in the solution updates. Values for all three can be estimated with varying levels of confidence. Machine accuracy is the easiest one, simply being a constant for a given computer architecture. Conditioning of the residuals evaluation [the right-hand sides of (3)] is probably the most difficult one, since it depends on implementation details of all physical models, device type, and bias conditions. Conditioning of the Jacobian matrix can be estimated to be no better than the conditioning of a much simpler partial differential equation (Laplace equation) discretized on the same triangular grid.

Typical values for the condition numbers for 2-D device simulation problems are summarized in Table I, while a graphical interpretation of the situation is shown in Fig. 1. In essence, starting from machine precision, some accuracy is lost due to the conditioning of the Jacobian and some more due to the conditioning of the right-hand sides evaluation. What is left must be sufficient to provide engineering accuracy (e.g., 5%) plus a certain "safety zone" of 2–3 decimals. This "safety zone" should help make sure that the calculated Newton updates have the correct direction, which may be critical in certain "narrow-valley" cases where there is much stronger variation along specific directions. The goal in making one Newton step is always to reduce the residual, which requires incrementing the current solution vector in the appropriate direction. Machine accuracy times both condition numbers should therefore be smaller than around 1e-3 to 1e-6.

The error of the solution update can therefore be no better than machine precision used to represent the current solution estimate x_0, amplified by the product of condition numbers related to the assembly of the right-hand sides and the solution of the linear system. The remaining accuracy may or may not be sufficient to achieve convergence of the nonlinear iteration in a particular application. The success or failure of the simulation is thus determined by:

1) machine precision used for the calculations;
2) condition number of the right-hand sides evaluation;
3) condition number of the Jacobian matrix;
4) physical sensitivity of the problem.

The last three obviously can change in the course of the simulation. For instance, the accuracy with which the current density inside a given device can be obtained will strongly depend on the current level, i.e., at low currents the accuracy will be much worse. Similarly, the condition number of the Jacobian is likely to degrade near breakdown, thermal runaway, etc. This is due to the loss of uniqueness of the steady-state solution near breakdown, meaning that at breakdown, the current-voltage relationship is not unique. Finally, the physical sensitivity of the problem, i.e., how much accuracy is sufficient to ensure convergence of the nonlinear iteration is clearly a function of the device state and thus bias conditions.

TABLE II
EXAMPLE OF POOR CONVERGENCE (POWER MOSFET SIMULATION). ERROR COLUMNS DENOTE RELATIVE CHANGES IN THE ELECTRIC POTENTIAL (v), ELECTRON (n) AND HOLE (p) CONCENTRATIONS, AND THE LATTICE TEMPERATURE (t)

iter	v-error	n-error	p-error	t-error
1	3.5516E+00	8.1401E-01	1.5701E+00	4.3905E-02
2	1.5721E+00	2.3274E-01	2.0513E-01	2.8001E-03
3*	1.9608E-02	4.0258E-04	7.7408E-04	9.8210E-06
4	6.5516E-05	4.5868E-06	1.2278E-05	5.9540E-08
5	1.3987E-04	1.0190E-06	9.0222E-07	7.6241E-08
6	1.3411E-04	9.7333E-07	8.5525E-07	7.3397E-08
7	1.3469E-04	9.6415E-07	8.4647E-07	6.9815E-08
8	1.3609E-04	9.7396E-07	8.5544E-07	6.9715E-08

If the error of the solution update according to (5) is high enough to produce a significant disturbance in the solution, the convergence of the nonlinear iteration (2) can be compromised. Typical behavior in this case is: decreasing residuals and update norms for a number of iterations until a certain update norm is reached and no further improvement of the solution thereafter. An example is shown in the Table II.

In a case like the one shown above, it will not be possible to achieve accuracy better than approximately 1e-4 in potential. The actual value is determined by the simulation grid and bias conditions in addition to the physics of the application. According to (5), this error limit can only be improved by improving the machine precision, the conditioning of the physical problem, and the evaluation of the right-hand side or the conditioning of the Jacobian. Improving the conditioning of the physical problem may not be possible within the requirements of the simulation. (For instance, the conditioning of a silicon on insulator (SOI) problem can be improved by attaching the floating body to ground via a large resistor, the conditioning of an off-state impact ionization breakdown can be improved by reducing minority carrier lifetimes, etc. This, however, might not make physical sense.) What is left in this case is the conditioning associated with the Jacobian matrix. It is determined by the size and the quality of the simulation mesh and can be improved significantly leading to a solution of convergence problems in many difficult cases.

A. Estimating Jacobian Condition Numbers

The condition number of an algorithm is the error amplification factor which allows us to estimate the relative error in the result as a function of the relative error in the argument. For a function $y(x)$ the condition number is in general given by (4)

$$C_x^y = \frac{x}{y} \cdot \frac{dy}{dx}. \tag{6}$$

For certain algorithms, the condition number can be very large leading to a significant loss of accuracy. A classic example is subtraction of numbers of similar magnitude where the condition number approaches infinity

$$\lim_{a \to b} \left(C_a^{a-b} \right) = \lim_{a \to b} \left(\frac{a}{a-b} \right) \to \infty. \tag{7}$$

The condition number associated with the solution of a linear algebraic system of equations such as (3) is given by the condition number of the system matrix J which can be expressed as the ratio of its largest to its smallest eigenvalue

$$\mathrm{cond}(J) = \left|\frac{\lambda_{\max}}{\lambda_{\min}}\right|. \tag{8}$$

The condition number of the Jacobian arising in the finite element solution of second order partial differential equations on irregular elements can be estimated according to [3] as

$$\mathrm{cond}(J) \leq Q \cdot h_{\min}^{-2}. \tag{9}$$

The constant Q is a function of element quality and becomes large for degenerate elements, i.e., elements with small internal angles. To estimate the quantitative effect of element shape on the condition number, we use the inverse of the expression suggested in [4] for each element. This expression has the value of one, i.e., best possible conditioning, for equilateral triangles and is large for degenerate triangles (a is the area of the triangular element, h_i are the lengths of the edges)

$$q = \frac{h_1^2 + h_2^2 + h_3^2}{4a\sqrt{3}}. \tag{10}$$

The total condition number of the system is determined by the worst element, i.e., the whole system can become singular because of one degenerate element. We therefore combine the contributions of all elements to obtain the following estimate for the condition number of the Jacobian with $L_{\max}$ being the maximum domain dimension

$$\mathrm{cond}(J) \leq \max_i(q_i) \cdot \left(\frac{L_{\max}}{h_{\min}}\right)^2. \tag{11}$$

Expression (11) is the grid quality estimator proposed in this work. It takes into account three major effects which can influence the conditioning of the Jacobian, and thus, the quality of the triangulation 1) and 2) are lumped into the ratio $L_{\max}/h_{\min}$):

1) the total number of elements;
2) nonuniformity of the grid (does the grid density change much locally?);
3) geometric element quality (are degenerate elements present?).

In practical device simulations all three can be significant for certain types of grids. Grid modifications with the goal of improving its quality should attempt to achieve a balance between the need to minimize (9) and at the same time keep the discretization error sufficiently small to ensure meaningful results. As pointed out in [3], these are, in general, conflicting requirements since the discretization error improves with smaller element size and larger grid node counts while the conditioning degrades in this case.

The estimate in expression (11) can be pessimistic in some cases. On the other hand, it can also be optimistic since it does not include global grid properties such as how many poorly shaped elements are present and physical sources of ill conditioning discussed in the next section. Having said this, the estimate appears to be a valuable indicator for possible conditioning problems due to grid quality especially when used to compare the numerical quality of different triangulations of the same physical problem. Since the physical problem is given and cannot usually be modified to improve the numerical properties of its solution, improving the grid quality is often the only option to alleviate ill conditioning.

B. Physical Conditioning

Practitioners of device simulation know that certain types of devices cause more problems in applications than others. Examples are SOI-type devices, thyristors, diacs, or any device which contains floating regions. A mathematical analysis of the problem has been presented in [9], where it was shown that floating regions can in fact lead to very poor conditioning of the steady-state semiconductor equations. The same work also showed that if all p-type and n-type regions are contacted, the physical conditioning of the equations will be good.

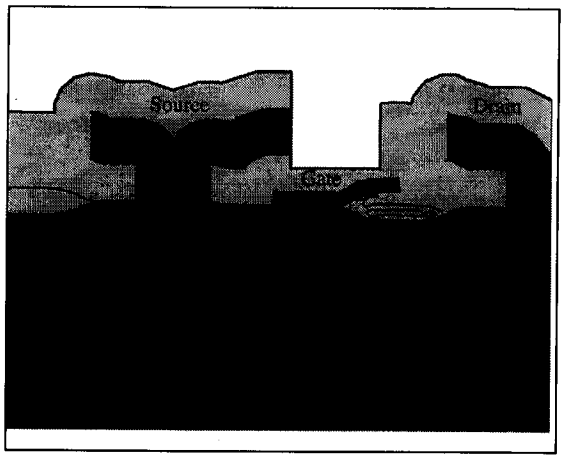

Fig. 2. Original power MOS structure (shading indicates material types) in the active area.

The situation becomes more complicated when lattice heating or energy balance are present, if the device undergoes a state transition such as breakdown, latch-up, etc. In some of these cases, a steady-state solution may not exist at all (for example, for a diode with a reverse bias higher than the breakdown voltage applied to its terminals, a solution does not exist).

Simulation of electrothermal breakdown is difficult for a combination of physical and numerical reasons. At breakdown, especially when snap-back current-voltage behavior is present, the device is unstable. Small changes in bias conditions at this point result in large changes in the internal conditions of the device. A state transition is taking place, with the transition point being the point of unstable equilibrium. Numerically this situation is characterized by degrading conditioning (i.e., quality) of the discretized algebraic equations, which are solved at each Newton step. This makes it harder to achieve convergence as the sensitivity to initial guesses increases, necessitating small bias increments near breakdown. This is a consequence of the fact that small bias changes may result in dramatic changes in internal device behavior.

In summary, physical effects can lead to increased sensitivity of the problem, leading to increased accuracy requirements on the solution. This can be a problem if a significant loss of accuracy is already present due to grid quality. If this is the case, improving grid quality can help improve the accuracy of the solution and achieve convergence.

III. Device Structure and Original Grid

The device under study is a power MOSFET. Its structure in the active region is shown in Fig. 2. The device was generated using the 2-D process simulator TSUPREM-4 [1]. Adaptive gridding was utilized during the process simulation. At the end of the simulation the grid has about 3800 nodes. As seen in Fig. 3, in addition to being rather large, the grid also contains a large number of poorly shaped triangles with small internal element angles. The condition number estimate of this structure is extremely high, according to (11)

$$\mathrm{cond}(J) \leq \max_i(q_i) \cdot \left(\frac{L_{\max}}{h_{\min}}\right)^2 \approx 700 \cdot 10^{13} = 7 \cdot 10^{15}. \tag{12}$$

There are apparently several sources of problems with this grid.

1) The grid contains extremely small elements (some as small as 0.1 nm).

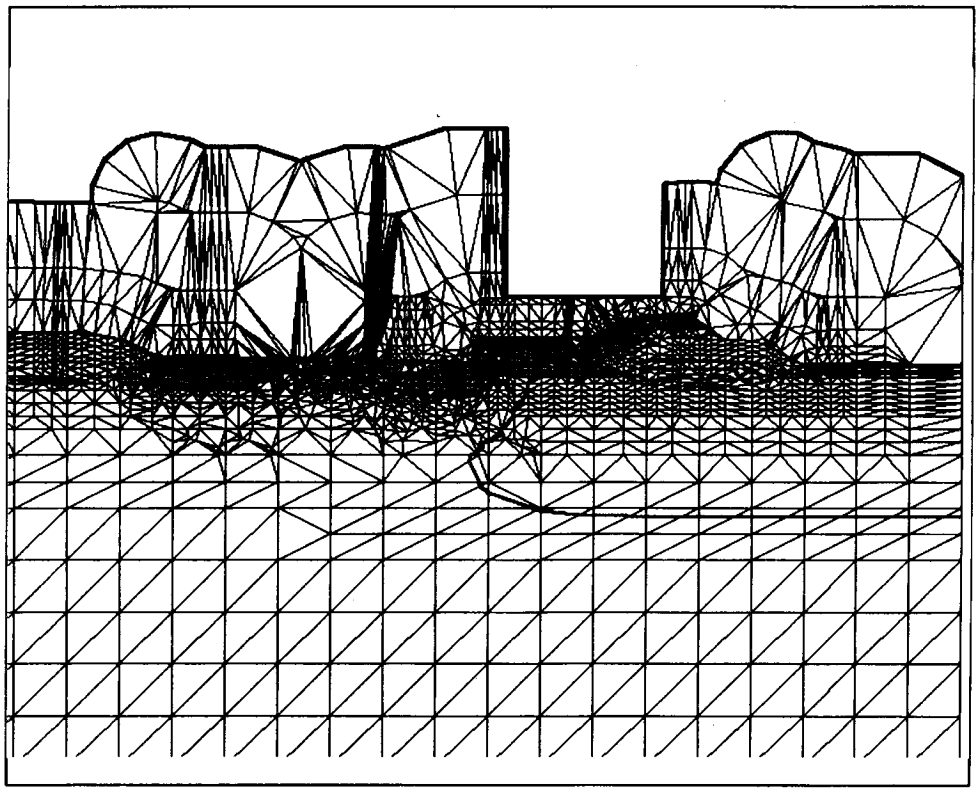

Fig. 3. Original power MOS grid in the active area $\text{cond}(J) \leq 10^{15}$.

2) The domain is large, aggravating the element size problem (500 μm).
3) Poorly shaped elements are present ($\max_i(q_i) = 700$).

An on-state breakdown simulation of this device was attempted using a DC bias sweep of the drain with the gate kept at 5 V. A curve-tracing or continuation method was used [8] to trace the snap-back current-voltage curve without the need to switch between voltage and current boundary conditions at the drain. Thermal boundary conditions were reflective at the sides and the top and constant 300 K at the bottom of the device.

As indicated by the high condition number, simulation of the device using the grid shown in Fig. 3. is nearly impossible. Convergence problems of the type shown in Table II occur even at low biases. The problem becomes worse as the device approaches thermal breakdown when high temperatures cause silicon to become intrinsic, conductivity rises rapidly with increasing current, and the current-voltage characteristic exhibits a snap-back behavior.

IV. Remeshing the Device

Replacing the grid by a completely new one is the most drastic attempt at improving its quality. In severe cases where the original grid is of very poor quality it may not be possible at all to improve it in any other way. The mesh generation strategy is based on a novel advancing front technique. It produces locally quasi-orthogonal anisotropic meshes of high quality with well-controlled element sizes and shapes. The implementation of the algorithm described in this paper utilizes an unstructured triangular grid generator called TRI [5], which was derived from the piecewise linear finite element multigrid (PLTMG) package [4].[1]

[1] Implementations based on "point-cloud" type Delaunay triangulations have been shown to be substantially more flexible and robust [11].

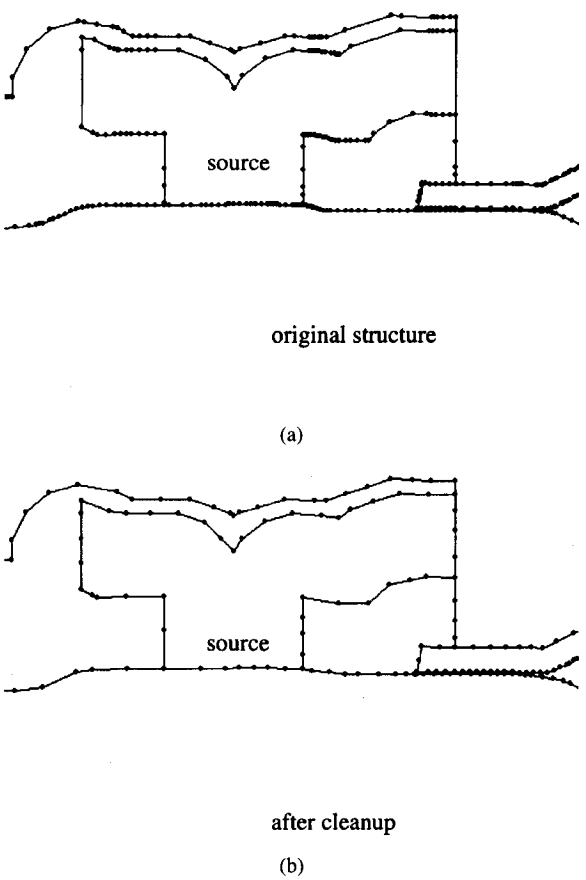

Fig. 4. The cleanup algorithm. (a) Portion of the original structure and (b) the modified structure.

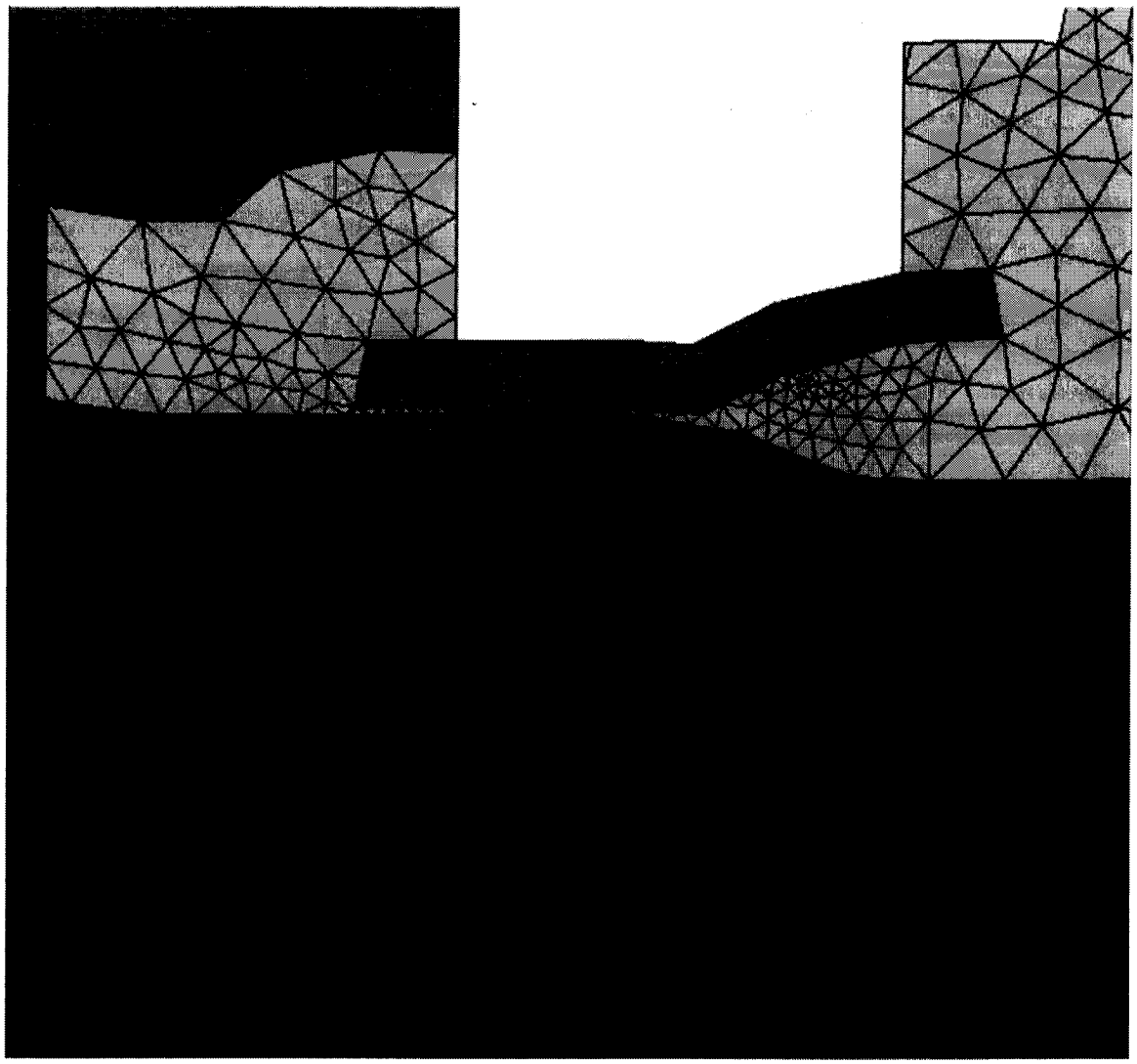

Fig. 5. Isotropic power MOS grid, 1153 nodes, $\mathrm{cond}(J) \leq 10^{10}$, 11 min CPU time for an on-state breakdown simulation.

The grid generator creates a high quality mesh based on polygonal boundaries of each material region present in the device and user-supplied target grid densities in certain areas. Two critical parts of the approach (cleanup and creation of anisotropic layers) are described below. The cleanup step is essential to remove boundary edges not relevant for the device under study, such as collinear edges and small edges in electrically uncritical device regions. Anisotropic grid layers are an absolute necessity for an accurate discretization of highly anisotropic inversion layers where the variation of carrier concentrations and electric potential is many orders of magnitude higher across the inversion layer than it is along the layer. The new algorithm which is briefly summarized below provides a general, robust, and computationally efficient approach to generate such grid layers. It is applicable to any geometry including V-grooves, trenches, shallow trench isolation (STI) corners, etc., and allows a high degree of control over the produced grid in the anisotropic layer.

A. *Preparation Algorithm (Cleanup)*

A preparation step is carried out before generation of the new grid to remove boundary edges according to user-specified criteria. This is important for the present example in which extremely small elements are present in areas where a dense grid is not needed for device simulation. In our case, the reason was primarily the roughness of the Al-Si interface at the source contact due to certain processing steps. Since the silicon under this rough surface is highly doped, the electric behavior of this part of the device is easily described by a much coarser grid. Thus the description of this boundary can be significantly simplified without any effect on solution accuracy.

On the other hand, it is extremely important to preserve geometry details in other, more electrically critical areas of the device. In particular, any modifications of the boundary must not alter the thickness and shape of the gate oxide. Our preparation step therefore preserves geometry details near the gate while cleaning up the geometry near the source electrode. Results are demonstrated in Fig. 4. The original structure had a large number of extremely small boundary edges along the boundary of the aluminum source electrode. These were not only irrelevant for an accurate device simulation, they also had a highly detrimental effect on grid quality and its size and were therefore removed.

B. *Creating Anisotropic Grid Layers*

The new gridding algorithm allows substantial user control over the density and quality of the grid in selected areas. An important

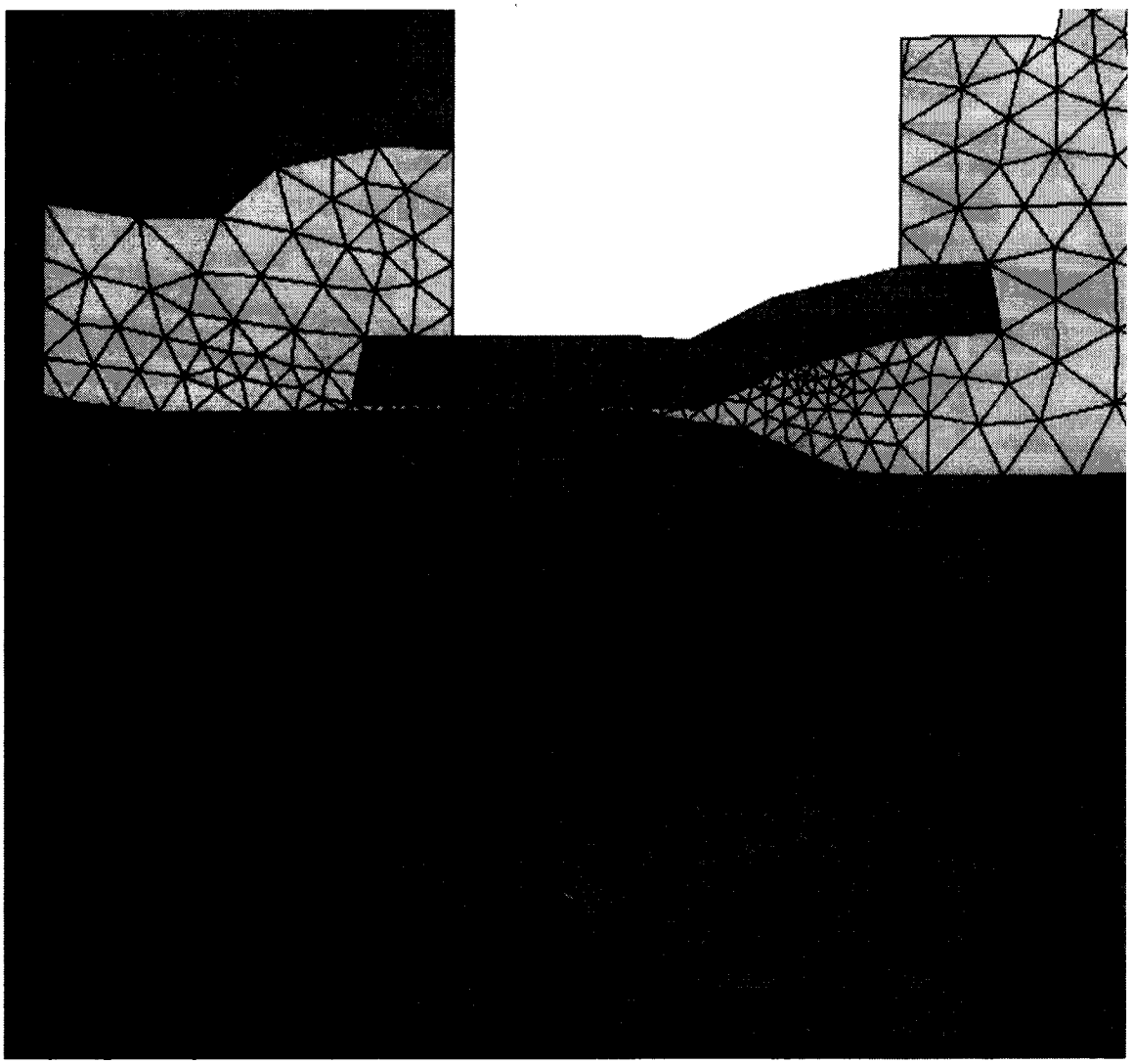

Fig. 6. Anisotropic power MOS grid resolving the inversion layer. 1341 nodes, $\mathrm{cond}(J) \leq 10^{12}$. 20 min CPU time for an on-state breakdown simulation.

requirement on grids for device simulation is resolving inversion layers in MOS-type devices. This is difficult to achieve using a general unstructured Delaunay-type grid generator such as [4] and [5] which normally generates isotropic grids with the goal of creating nearly equilateral triangles. On the other hand, quad-tree type grid generators, which can generate anisotropic grids, fail when the orientation of the layers to be resolved does not coincide with the preferential directions of the quads (vertical or horizontal). This makes quad-tree algorithms not robust for general applications where the device structure is created by a process simulation and is therefore not necessarily planar. They are also intrinsically inapplicable to device structures such as a V-groove MOS (VMOS) or a vertical trench MOSFET where the channel orientation is neither exactly vertical nor horizontal.

The new algorithm achieves highly anisotropic grids matched to arbitrarily shaped boundary surfaces including V-grooves and trenches. For the present study, two different simulation grids were used: one which is isotropic in the channel area shown in Fig. 5. and one with an anisotropic layer of elements shown in Fig. 6. The number and thickness of layers is controlled by the user.

The basic approach is illustrated in Fig. 7: it takes advantage of the fact that a general Delaunay triangulation algorithm respects region boundaries. To force the creation of element layers following the surface, artificial region boundaries are introduced in a sequence of layers with increasing thickness as shown in Fig. 7. All layers follow the shape of the generating surface using an *advancing front* approach similar to a string-based etching algorithm. The "etching" velocity, i.e., the increments between successive layers, is controlled by the user and in this example was chosen smaller in the channel area than near the contacts.

The actual triangulation is performed by a general Delaunay-type algorithm based on [5]. In contrast to previously reported boundary layer approaches [6] no iteration procedure is required and the total CPU time for the generation of the anisotropic layer and the triangulation of the entire structure in the examples was well under 1 min.

The condition numbers of the Jacobian for the isotropic and anisotropic grids are estimated to be approximately 10^{10} and 10^{12}, respectively, up to five orders of magnitude better than for the original grid, although still quite high. This is due both to the improved shape of the elements as well as a significantly reduced ratio of domain size to smallest element size. The total node counts are 1153 and 1341, with most grid nodes concentrated near the channel of the device in both cases where a high grid density is important to

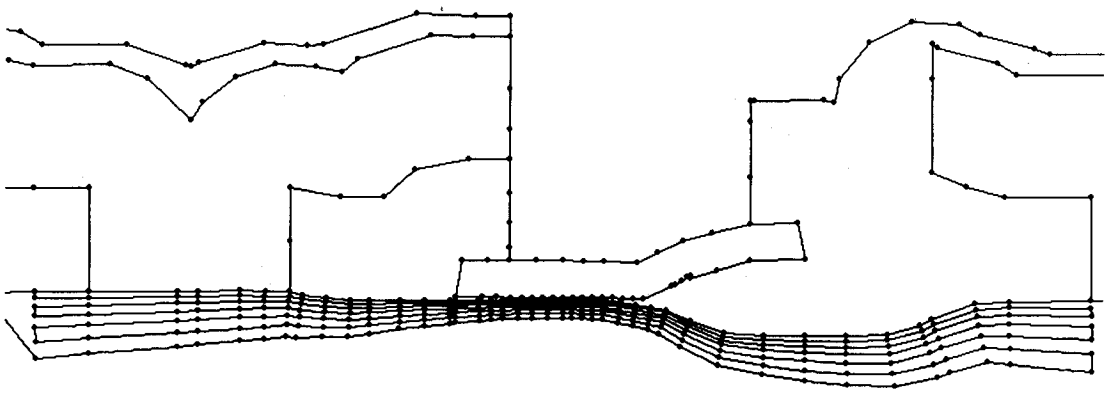

Fig. 7. Algorithm for the generation of anisotropic layers in complex structures: boundary layer before triangulation. Final grid is shown in Fig. 6.

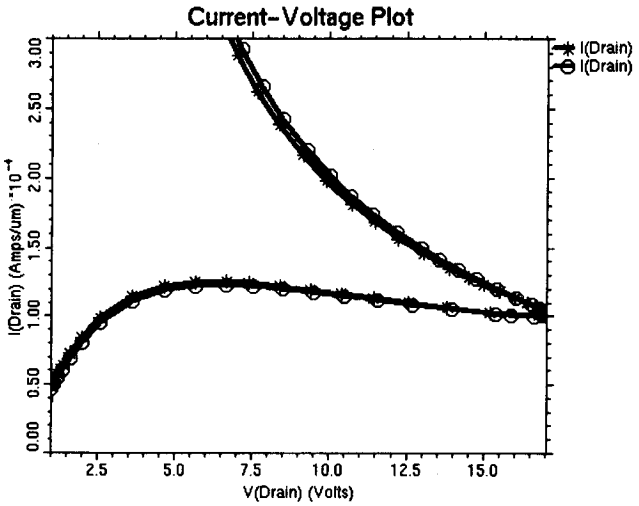

Fig. 8. Simulated power MOSFET breakdown curve (asterisks—anisotropic grid, circles—isotropic grid).

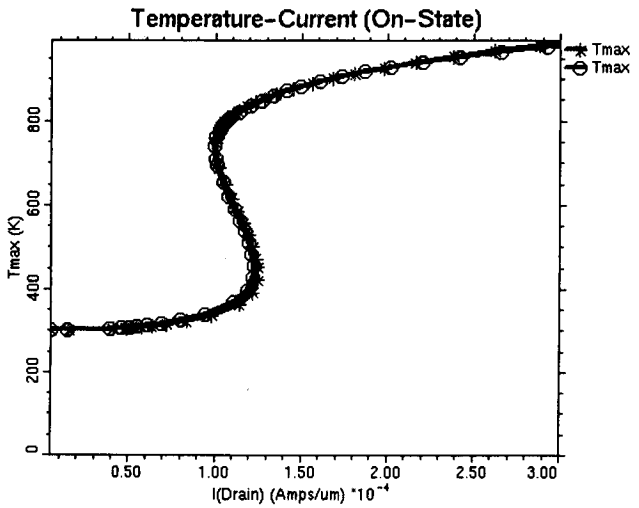

Fig. 9. Simulated temperature versus current curves (asterisks—anisotropic grid, circles—isotropic grid).

ensure sufficient accuracy. Both grids are substantially smaller than the original simulation grid (3705 nodes).

C. Simulation Results

Four simulations were carried out: an off-state ($V_{\text{gate}} = 0$ V) and an on-state ($V_{\text{gate}} = 5$ V) breakdown simulation for each of the simulation grids shown in Figs. 5 and 6. As mentioned above, it was not possible to carry out a simulation using the original grid due to convergence problems. An important observation is that there was a measurable difference in the convergence of the simulation using the two new grids. Although the node counts are similar, the condition number estimate is about two orders of magnitude higher (worse) for the anisotropic grid. While such a grid is clearly desirable from the point of view of reducing the discretization error, avoiding cross-wind diffusion in the channel, etc., the tradeoff between accuracy and stability is evident.

When using the grid shown in Fig. 6, convergence behavior similar to what is shown in Table II was observed, i.e., the residuals decreased rapidly (quadratically) to a small value beyond which further improvement was not possible. We therefore carried out all simulations using a relaxed set of convergence criteria (error tolerances). Interestingly, it proved sufficient to increase the error tolerances by two orders of magnitude (from 1e-5 to 1e-3), which corresponds to the difference in the condition number estimates between Figs. 5 and 6.

D. On-State Breakdown

Complete on-state ($V_{\text{gate}} = 5$ V) current-voltage curves were generated in 11 and 20 min, respectively, on a DEC Alpha. The difference in CPU time is mainly due to different convergence behavior since the difference in grid size is only about 20%. The current-voltage curve shows typical snap-back behavior with a breakdown voltage of approximately 16 V (Fig. 8, asterisks—anisotropic grid, circles—isotropic grid). The curves also show negative differential resistance due to decreasing mobility at higher temperatures, a self-heating effect also observed in SOI devices. This is confirmed by the S-shaped temperature versus current curve shown in Fig. 9. Snap-back occurs at approximately 800 K when the device becomes intrinsic and its conductivity increases exponentially with further increasing temperature. This interpretation is supported by Fig. 10 which shows current flow through the bulk of the device, where at normal operating conditions no current flow is present. It should be pointed out that the current flow lines are quite smooth throughout the device. This is due to the lack of preferential directions in the grid which allows unrestricted current flow according to the physics of the problem. It is an advantage of Delaunay grids such as shown in Figs. 5 and 6. Rectangular or quad-tree grids often show "stair-stepping" artifacts in their solutions because there the current flow is restricted to vertical or horizontal only (there is no connectivity along the hypotenuse of a right-angle triangle).

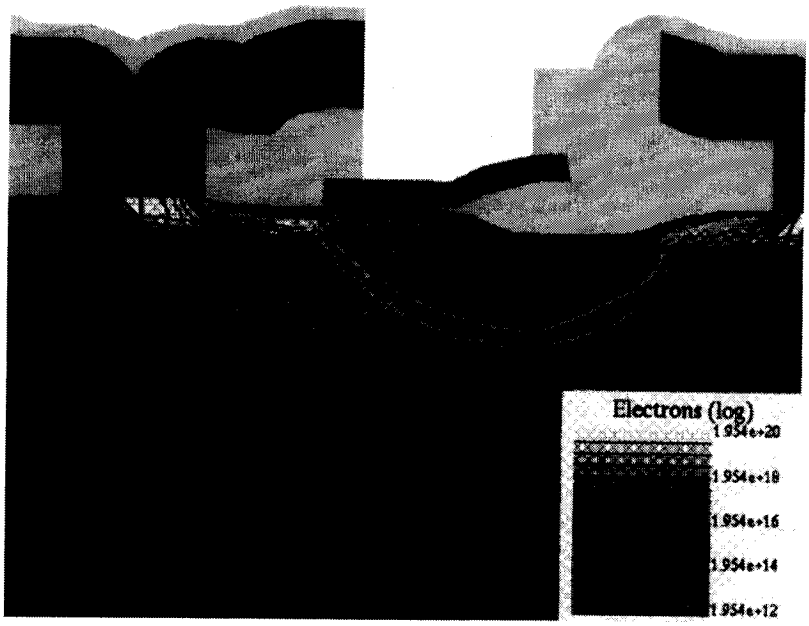

Fig. 10. Electron concentration and current flow lines after breakdown.

Both curves shown in Fig. 8 indicate that small bias increments were taken by the continuation method [8] near the snap-back point due the state transition taking place with a drastic change in device conductivity, current path, carrier concentrations, etc. Interestingly, there is not much difference between the two solutions obtained using the two grids (asterisks—anisotropic grid, circles—isotropic grid). The breakdown voltage, defined as the highest source-drain potential difference during the simulation, is 16.98 V (isotropic mesh) and 17.05 V (anisotropic mesh).

E. Off-State Breakdown

Results of the off-state breakdown simulation are shown in Fig. 11. The situation is somewhat different here from the on-state breakdown simulation. In the off-state case it is not necessary to consider device self-heating since avalanche breakdown occurs at low current levels, thus a standard three equation drift-diffusion model is used (Poisson's and continuity equations for electrons and holes). As a result, the solution time per bias point is smaller. On the other hand, there are more bias points so that the total simulation times are similar to the on-state breakdown case (15 and 22 min, respectively).

The off-state breakdown simulation is significantly easier to carry out than the on-state breakdown discussed in the previous section. In the off-state case, it is possible to perform a simulation using the original TSUPREM-4 grid, which provides an interesting reference: a breakdown voltage of 101.6 V is obtained at approximately 60 min CPU time (solid line in Fig. 11).

All three breakdown curves (original, isotropic, and anisotropic grids) are remarkably similar up to the onset of avalanche breakdown. Defining breakdown voltage as the highest source-drain potential difference during the simulation, breakdown voltages of 104.7 V for the isotropic grid and 104 V for the anisotropic one are obtained about 3% higher than the solution obtained using the original grid. Finer grids usually result in a lower breakdown voltage because of better resolution of the electric field. Since the field is highly nonuniform in the drain junction where avalanche breakdown is initiated, a coarse grid underestimates the peak field value and thus overestimates the breakdown voltage.

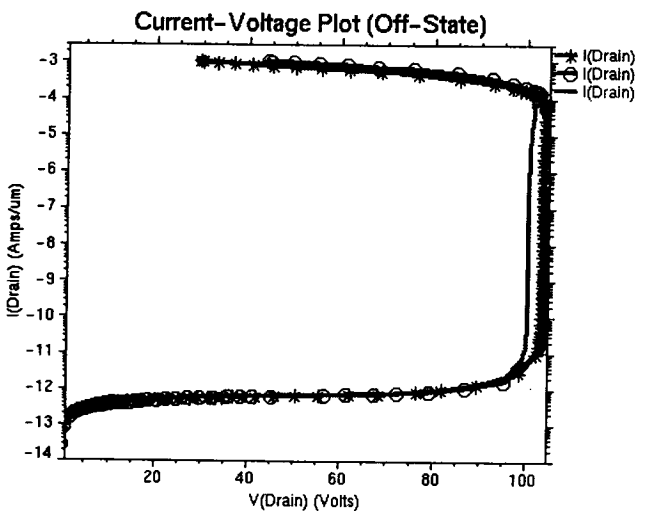

Fig. 11. Off-state breakdown curves (solid line—original grid (TSUPREM-4), circles—isotropic grid, asterisks—anisotropic grid).

V. Conclusions

Convergence problems can be a significant obstacle to industrial applications of device simulation tools. In a number of difficult cases, these problems turned out to be related grid quality and could be resolved by improving the quality of the simulation grid. A quantitative measure to evaluate grid quality is obtained as an estimate for the condition number of the Jacobian of the discretized semiconductor equations. This estimate can be used for *a priori* judgement of suitability of certain triangulations for device simulation.

A new grid generation algorithm has been developed which combines the robustness of a general unstructured Delaunay triangulation with highly anisotropic grid structures mandatory in the simulation of MOS-type structures. Improved quality of the grid after remeshing offers sufficient stability to perform a simulated experiment under a variety of process conditions to optimize the device according to practical requirements. Similar results have been achieved in several

other applications including SOI, MOS and Bipolar breakdown, insulated gate bipolar transistor (IGBT), etc.

In addition to greatly improving grid quality which solved the observed convergence problems, the new grid generation algorithm allows a drastic reduction in grid size without sacrificing solution accuracy. Since there is a superlinear dependence of CPU time on grid size (approximately $O(N^{1.5-1.75})$ in 2-D), the resulting speed up can be significant even in less critical cases where convergence problems are not observed with the larger original grid. In the shown example with the original grid size of 3700 and grid size after remesh of 1300, a speed up of more than 3–4 times per Newton iteration would be expected (total CPU time may vary if automatic biasing is utilized).

The discussed application example is an electrothermal breakdown simulation of a power MOS device, which could not be simulated using the original grid. The suggested remeshing strategy achieved an improvement in grid quality by 3–5 orders of magnitude and allowed an entire breakdown current-voltage curve to be generated in under 0.5 h on a workstation. Similar observations were made for an off-state avalanche breakdown of the same device.

Since the original advancing front mesh generation algorithm was first demonstrated in the work reported here, a more general and robust version of it has been implemented as pdMesh, a tool from PDF Solutions [10], [11]. Subsequently, it was also used by Technology Modeling Associates, Inc. (TMA) in the so-called adaptive boundary conforming (ABC) mesh [12] and most recently by Integrated Systems Engineering (ISE) in their MDRAW-ISE product [13].

ACKNOWLEDGMENT

The author would like to thank S. Verdonckt-Vandebroeck from Xerox Research who provided the initial motivation for this work as well as the power MOSFET structure where the convergence problems were more severe than usual. Many valuable discussions on this subject with colleagues at TMA are also greatly appreciated, in particular with J. G. Rollins and V. Boksha.

REFERENCES

[1] *TSUPREM-4 User's Manual*, Technology Modeling Associates, Palo Alto, CA, 1995.

[2] S. Selberherr, *Analysis and Simulation of Semiconductor Devices*. New York: Springer-Verlag, 1984.

[3] G. Strang and G. J. Fix, *An Analysis of the Finite Element Method*. Englewood Cliffs, NJ: Prentice-Hall, 1973.

[4] *PLTMG User's Guide 7.0*, R. E. Bank, Soc. Ind. Appl. Math. (SIAM), 1994.

[5] M. R. Pinto, "Comprehensive semiconductor device simulation for ULSI," Ph.D. dissertation, Stanford University, CA, 1990.

[6] S. Kumashiro and I. Yokota, *A Triangular Mesh Generation Method Suitable for the Analysis of Complex MOS Device Structures*, NUPAD V, Honolulu, 1994.

[7] V. Axelrad and R. Klein, "Electrothermal simulation of an IGBT," in ISPSD-92, Tokyo, Japan, 1992.

[8] *MEDICI User's Manual*, Technology Modeling Associates, Palo Alto, CA, 1995.

[9] U. Ascher, P. A. Markowich, C. Schmeiser, H. Steinrueck and R. Weiss, "Conditioning of the steady-state semiconductor device problem," Computer Sci., Univ. British Columbia, Vancouver, Can., Tech. Rep. 86-12, 1986.

[10] V. Axelrad, "pdMesh White Paper," PDF Solutions, San Jose, CA, Dec. 1996.

[11] V. Axelrad, G. Long, and P. Kuepper, *Elimination of Meshing Noise in Statistical TCAD*, in Int. Workshop Semiconductor Metrology (IWSM), Kyoto, Japan, June 1997.

[12] V. Moroz, S. Motzny, and K. Lilja, *A Boundary Conforming Mesh Generation Algorithm for Simulation of Devices with Complex Geometry*, in Simulation Semiconductor Devices Processes (SISDEP), Boston, Sept. 1997.

[13] Integrated Systems Engineering (ISI), "ISI Newsletter 97," Integrated Systems Eng. (ISE AG), Switzerland, Dec. 1997.

A Controller Redesign Technique to Enhance Testability of Controller-Data Path Circuits

Sujit Dey, Vijay Gangaram, and Miodrag Potkonjak

***Abstract*— We study the effect of the controller on the testability of sequential circuits composed of controllers and data paths. We show that even when all the loops of the circuit have been broken by using scan flip-flops (FF's) and the control and data path parts are individually 100% testable, the composite circuit may not be easily testable by gate-level sequential automatic test pattern generation (ATPG). Analysis shows that a primary problem in test pattern generation of combined controller-data path circuits is the correlation of control signals due to implications imposed by the controller specification. A design-for-testability (DFT) technique is developed to redesign the controller such that the implications which may produce conflicts during test pattern generation are eliminated. The DFT technique involves adding extra control vectors to the controller. Experimental results show the ability of the controller DFT technique to produce highly testable controller-data path circuits, with nominal hardware overhead.**

***Index Terms*— Data path, design for testability, controller, high-level synthesis, high-level testability.**

I. INTRODUCTION

Several existing scan-based design-for-testability (DFT) techniques use heuristics based on the topology of a circuit like breaking all loops, except self-loops, and reduction of sequential depth as ways to make sequential automatic test pattern generation (ATPG) of circuits easy [1]–[3]. However, it has been observed that for many circuits, even when all loops are broken using scan flip-flops (FF's) and the sequential depth is low, the circuit remains difficult for sequential ATPG. This paper introduces a new DFT technique to supplement topology-based DFT techniques like loop-breaking and sequential depth reduction. The proposed technique uses high-level

Manuscript received February 23, 1996. This paper was recommended by Associate Editor S. Reddy.

S. Dey was with C & C Research Laboratories, NEC, Princeton, NJ 08540 USA. He is now with the Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA 92093 USA (e-mail: dey@ece.ucsd.edu).

V. Gangaram was with C & C Research Laboratories, NEC, Princeton, NJ 08540 USA. He is now with Design Technologies of Intel Corporation, Folsom, CA 95630 USA.

M. Potkonjak was with C & C Research Laboratories, NEC, Princeton, NJ 08540 USA. He is now with the Computer Science Department, University of California, Los Angeles, CA 90095 USA.

Publisher Item Identifier S 0278-0070(98)04511-4.