Using a Genetic Algorithm for exploring a Semiconductor Fabrication Model

Rodrigo Sandoval
TCAD Technical Lead
Synopsys R&D Center
Santiago, Chile
rodrigo.sandoval@synopsys.com

***Abstract*—Initial iterations or generations of this particular Genetic Algorithm (GA) implementation for optimizing a semiconductor fabrication process, not only search for an optimal solution, but also, with the aid of a visual tool, help understanding the complex characteristics of the model. Finally, thanks to this visual analysis, some adjustments are proposed to the algorithm execution in order to drive the search more effectively, as explained using a reference TCAD simulation-based example.**

I. Introduction

Typically, TCAD (Technology CAD) is used for parametric optimization: Each simulation is described by a set of parameters, whose values can be changed to represent different conditions for the simulation. The simulation yields one or more responses that result from the parameter combination.

The optimization is focused on finding the best possible combination of parameter values that results in a set of responses optimized with different criteria. Several algorithms can and are used for this optimization, among them, the Genetic Algorithm. One of the key aspects of the optimization of these models is a proper definition of the goal function.

A. *The optimization goal function*

This GA implementation considers that each response might be optimized with a different criterion (maximization, minimization, and close to a given value)., therefore requires that these responses values are normalized in a joint goal function, producing a single comparable value, to allow the evaluation of each experiment and their comparative optimality.

This global desirability function considers that [1]:

- A normalization function is applied to each simulation response, producing a value in the range of -1 to 1.
- Different desirability functions are used depending on whether the simulation response is to be minimized, maximized, or has an assigned target value.
- A user-defined weight is assigned to each response, representing its relative importance in the goal function result.
- These normalized and desirability-adapted values are summed up for the final goal function result, multiplying each of them by the corresponding weight.

To illustrate the use of an optimization model using this GA implementation, a semiconductor topography simulation example will be used for searching an “optimal” or ideal etch deposition, using one of the available deposition models.

B. *An example: Optimization of a deposition process*

This example is focused on understanding and ultimately finding the best conditions for a specific semiconductor topography resulting from operating a given deposition model. For this example, it is relevant to know that the deposition model is based on four factors (sputc1, sputc2, sputc4, and time) in an angle-dependant formula. Changes in these three factors affect the resulting shape of deposition layer, in this example a Nitride layer. Devices specifications define only certain deposition results as valid, and moreover, a resulting layer as shown in the next image.

To describe this Nitride layer shape, some measurements are made in specific points. In this case, three points are chosen for measurement reference and are shown in the next image. These values are labeled Ta_1, Ta_2, and Ta_3, considered as the model “responses” for the optimization approach used here.

Therefore, for optimization purposes and goal function evaluation, a formula which includes these four factors is used:

$$T = F(sputc_1, sputc_2, sputc_3, time)$$

where:

T is a vector including the three dimension values Ta_1, Ta_2, and Ta_3.

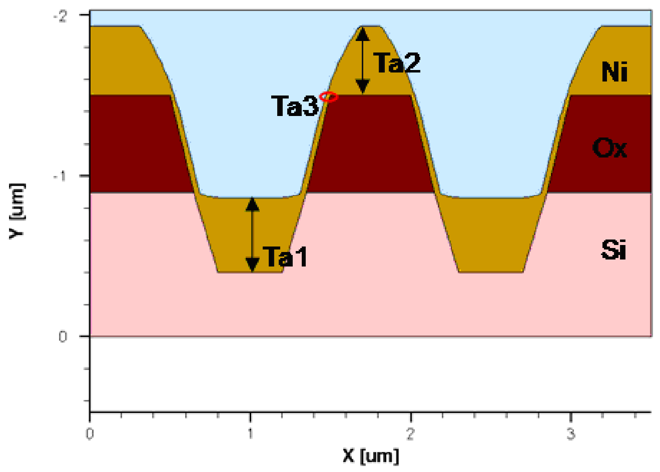

Figure 1. The Nitride deposition resulting layer, being measured in three different points.

The optimization problem is finding the best combination of the 4 parameters that produces the desired deposition layer shape, defined by measuring Ta_1, Ta_2, and Ta_3. These 3 target values are obtained from a TCAD simulation, using Synopsys Sentaurus Topography [2], reproducing the physical behavior of the deposition model, and are initially equally-weighted for goal evaluation purposes, meaning that all of them are equally important when they are considered in the optimization goal function.

II. A GENETIC ALGORITHM IMPLEMENTATION

The Genetic Algorithm is an optimization approach, part of what is known as "Evolution Programming", and has been around for more than 40 years [3]. Also, this method offers a powerful way to explore the domain space of possible solutions, while other algorithms perform better when approaching a local search [4].

The main idea of this algorithm is that an initial population of chromosomes represents an initial set of possible parameter combinations or experiments. Iteratively, this initial population evolves to a better set of chromosomes, including a best solution, considered as the optimum. There are several ways to execute this evolution, including crossover, and mutation, among others. The optimality of each chromosome is evaluated using a fitness function.

For this type of models, some adaptations were made to the GA implementation: Each chromosome is a multi-parameter experiment and produces more than one result, which are then combined in the previously mentioned normalizing goal function that finally returns the fitness value used for optimization. In this case the target is minimizing this goal function result.

Following the standard GA implementation, next generation is produced by combining elite population, formed by the best chromosomes in previous generation surviving to the next, crossover, consisting of a child resulting of the weighted average of two semi-random parents, and mutation, which is the result of a new experiment version resulting as a variation of some gene in the previous version. The algorithm runs for a user-defined number of generations.

For this implementation, given that the standard GA will perform differently for different models, a few algorithm parameters were added to refine the behavior and the way in which each new generation is computed. The most relevant parameters are:

- *Initial generation*. While often the initial generation is produced randomly, in this case a Design of Experiments (DoE) is used, considering that different designs cover the solutions domain differently [7], and it is up the user to define a compromise between domain coverage and simulation cost.
- *Number of generations*. Number of different populations produced throughout the process.
- *Fitness base*. In order to properly compare fitness results for different experiments, there are some normalization functions that compensate exaggerated peaks in the fitness results.
- *Elite population size*. Number of the fittest experiments kept for the next generation.
- *Mutant percentage*. Proportion of mutants in relation to the total population in each generation.
- *Mutants from elite*. Indication whether mutants are generated from the elite population or just any other experiment in the generation.
- *Mutation percentage*. The amount of mutation should be applied to a gene (parameter value) in a chromosome.

Aside from this, the optimization criterion for each response, as well as its relative importance or weight can be set for the optimization execution.

Although each of these parameters has a default value, which allows a new model to be immediately tried out, without needing to set any of them, it's possible to set some or all of these parameters to drive the algorithm through the actual model.

Another particular feature of this implementation is the generation of a "generations file", which is basically a table that registers all the chromosomes configuration and their resulting responses and goal value. This table can be examined with numerous analysis tools to visually review the iterative process and the resulting evolution of the model towards a better or fitter set of chromosomes.

In this case, a parallel coordinate plot [8] displays simultaneously all chromosomes, becoming a key element to understand different aspects of the model, by visualizing and analyzing each generation separately.

III. Using the GA for visual exploration

Running the model for a couple of initial generations (between 4 and 10), with the default configuration, delivers information about some of the specific characteristics of the model, each generation offering some insights about some of its aspects.

A. *First generations*

Analyzing the first generations, using a parallel coordinate plot, allows to realize, as shown in the next image, how well covered is the domain space with the chosen DoE, or not.

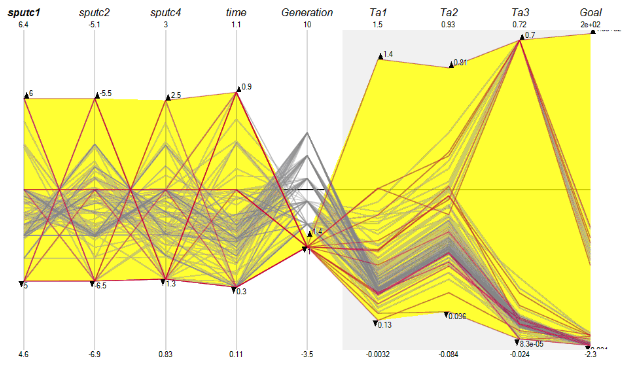

Figure 2. First generation used in the GA execution after evaluating the fitness or goal function. On the left, each parameter value and its range. On the right, with a gray background, each of the simulation-obtained measurements and corresponding goal function result. In the middle, generation number serves as a reference to understand the evolution.

Parameter values and ranges indicate whether the domain has been covered sufficiently. Also resulting diverse measurements evidence a rather wide variety of simulation-obtained shapes for the Nitride layer.

Each chromosome's fitness is displayed in the Goal axis (note that less Goal value equals better fitness for this goal minimization model), and it shows that this initial population produced several fitted solutions, as well as other not so fitted ones. One particular solution shows the worst fitness as displayed by the red line in the upper edge of the yellow section at the right. The significant difference of this solution with the others might suggest that this belongs to a low potential area.

B. *Intermediate generations*

Analyzing an intermediate generation allows understanding the speed of evolution and the size of the genetic pool.

As shown in the next image, the population converged in just a couple of generations to a set of very fit experiments belonging to a reduced portion of the whole domain space, but the variation on some parameter values suggest that these chromosomes are still diverse enough not just to cover a small high-potential area, but rather a big one, or several small ones.

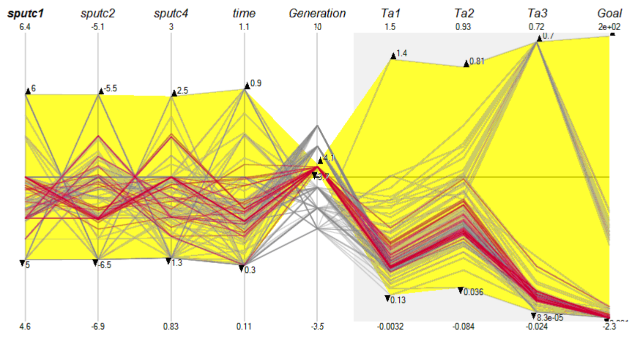

Figure 3. Intermediate generation plot.

Also, the variation in individual responses values suggests that the set of fitter solutions found so far, cover a rather wide variety of solution instead of converging to a reduced area.

C. *Last exploratory generation*

Finally, when analyzing the last exploratory generation resulting plot, other different aspects of the algorithm performance can be discovered. Considering that members of this last generation are evaluated as the overall best-found chromosome set, their gene values ranges suggest the optimal parameter values ranges for this deposition model.

At the same time, the variability in individual responses values for these similarly goal-function evaluated solutions, suggests that one or more of these responses' criterion has not been completely satisfied, evidencing a compromise when targeting a multi-response model.

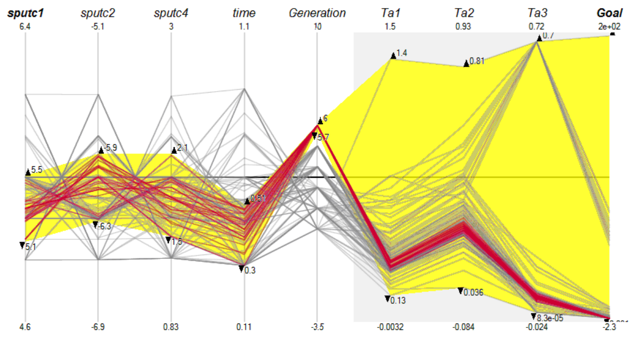

Figure 4. Last generation produced by the initial GA execution.

In the previous image, the right side of the plot shows that all experiments in this generation, displayed in red, are considered to be almost equally fit, according to the goal function value. As a confirmation, individual measurements values vary only a fraction, therefore suggesting that the whole population contains chromosomes that are similarly good for the desired result, and that the three responses criteria are equally solved in this case. But the left side of the plot evidences a visually significant variation in the parameters values, once again, suggesting that these chromosomes belong to different areas of the domain.

IV. ANALYSIS-DRIVEN ADJUSTMENTS

After analyzing the initial exploratory run of the algorithm, several conclusions and decisions can be made that finally translate into changes to the algorithm behavior parameters and try a new adjusted run, potentially leading to better results. These conclusions might even lead to try a different local search optimization algorithm in order to refine the best solution.

Having reviewed the results of the exploratory runs, analysis-driven adjustments can be made.

- Coverage quality in the first generation might lead to decide whether a more complete DoE is needed or if a simpler and less-costly one is enough in a definitive and longer optimization run of the algorithm.
- Experiments that result in a bad (high) goal value, discovered in early generations might suggest redefining the parameter value ranges to leave these experiments out from the beginning, reducing the total domain space, although not necessarily reducing the number of simulations needed.
- In any generation, analyzing the variance on goal function results might lead to decide a different type of fitness base in order to properly compare highly different response-resulting experiments. On the other hand, an excessively flat plot of the goal function results is a good candidate for a more sensitive fitness base, to allow a clearer comparison between similar experiments.
- If an intermediate generation is too close to the optimum, it might be the indication that the algorithm drives the population too aggressively towards convergence, and left out areas that potentially could produce better results. This could be adjusted by reducing the amount of individual mutation allowed, or even reducing the weight influence of the fittest parent in crossovers.
- If one or a few experiments survive many generations and eventually lead to regular results, might be an indication that the elite population size is too large. Decreasing the elite set size to only one experiment might force the algorithm to explore other initially low-potential areas that might lead to narrow good-potential sections.
- On the other hand if some very good solutions are becoming extinct in intermediate generations, it is an indication of the existence of more than one good potential area, and a clear recommendation to increase the size of the elite population.
- If the convergence to the optimum in the last generation is not evident enough, it might a clear indication that more generations are needed, or even that a gradient-based algorithm is needed for the final approach to the optimum solution.
- A slow convergence with too many low-fitness chromosomes is an indication that the area in which the optimum candidate is located might be too narrow. An indication to adjust the algorithm to converge more directly. For example, reduce the percentage of mutants in the whole population, and/or that mutant population should be produced from elite population only, and use a larger elite population size.
- Analyzing the last exploratory generation's individual responses helps validating if defined weights are right. If all the near-optimum solutions present a high variability in one or more individual responses values, it is an indication that the balance between individual targets for responses is compromised. Solving this means assigning the particular response a higher weight in comparison to the others, in order to properly understand its real influence in the model and to balance the goal function evaluation.

V. CONCLUSION

As the technology evolves, the underlying TCAD process and device simulation models undergo enhancements and fine-tuning. Using a visual exploratory optimization approach, which tunes the algorithm to match the changes in the TCAD models and parameters, has proven to be an ideal technique to maximize the results using an optimization algorithm. These optimization algorithms have helped to better understand the intrinsic aspects of the TCAD models.

VI. ACKNOWLEDGMENT

The latest implementation of the Genetic Algorithm in SWB Optimizer was implemented by Nikolai Tchernitchin, Synopsys Chile. The reference Topography example was designed by Raphael Hude, Synopsys Switzerland. Lars Bomholt, Synopsys Switzerland, encouraged the optimization visualization concepts.

REFERENCES

[1] Synopsys Inc, Sentaurus Workbench Optimizer User Guide, MV, California, USA, 2007.

[2] Synopsys Inc, Sentaurus Topography User Guide, MV, California, USA, 2007.

[3] Zbigniew Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, Berlin, Germany, 1999.

[4] Holland, J.H, “Adaptation in natural and artificial systems”, University of Michigan Press, Ann Arbor, 1975.

[5] Grefenstette, J.J, “Incorporating problem specific knowledge into Genetic Algorithms”, in [6] p.42-60

[6] Davis, L. (editor), “Genetic Algorithms and Simulated Annealing”, Morgan Kaufmann Publishers, San Mateo, Ca 1987.

[7] Montgomery, D., Design and Analysis of Experiments, John Wiley & Sons, 2005.

[8] Inselberg, A.. “The plane with parallel coordinates”, Visual Comput. 1, 69–97, 1985.