High speed extraction of process model parameters for 70nm technology using a distributed genetic algorithm

M. Murakawa*, Y. Oda**, H. Amakawa**, S. Baba**, T. Higuchi*, and K.Nishi**

* MIRAI-ASRC, AIST, 1–1–1 Umezono, Tsukuba, Ibaraki, Japan, m.murakawa@aist.go.jp
** Selete, 16–1 Onogawa, Tsukuba, Ibaraki, Japan

ABSTRACT

In this paper, we present, for the first time, a GA (Genetic Algorithm) application for process model calibration. We propose a distributed GA-based calibration technique combined with the traditional local optimization algorithm, which reduces calibration time considerably. Experimental results demonstrate that the calibration of 144 parameters can be completed with a few minutes, although this would typically take a human expert a few days. GA can, therefore, be a practical and robust tool for process/device calibration.

Keywords: parameter extraction, model calibration, genetic algorithm, parallel computation, B implantation, dual-pearson profile

1 INTRODUCTION

Although Technology CAD (TCAD) is already well-established as an indispensable tool for minimizing the development time and costs involved in marketing new LSI processes and devices, maximum TCAD utilization is heavily dependant on the calibration of relevant model parameters[1]. With the increasing complexity of TCAD models, however, calibration becomes more and more labor and time-intensive. Moreover, the presence of local minimums makes the calibration process much more difficult. Calibration using traditional gradient-based methods, such as the Levenberg-Marquardt (LM) algorithm[2], sometimes fails, and often yields inadequate results.

Recently, genetic algorithm (GA) has emerged as an efficient search method capable of finding optimal global solutions to complicated problems[3][4]. However, a potential drawback with GA is the considerable computation time required to a find a solution – the lack of research papers reporting the application of GA to process simulation would seem to indicate that nobody has seen this to be a feasible proposition.

In this paper, we present an application of GA to process model calibration. We propose a distributed GA-based calibration technique combined with the LM algorithm, which reduces calibration time considerably. Experimental results demonstrate that the calibration of 144 parameters can be completed with a few minutes, although this would typically take a human expert a few days. GA can, therefore, be a practical and robust tool for process/device calibration.

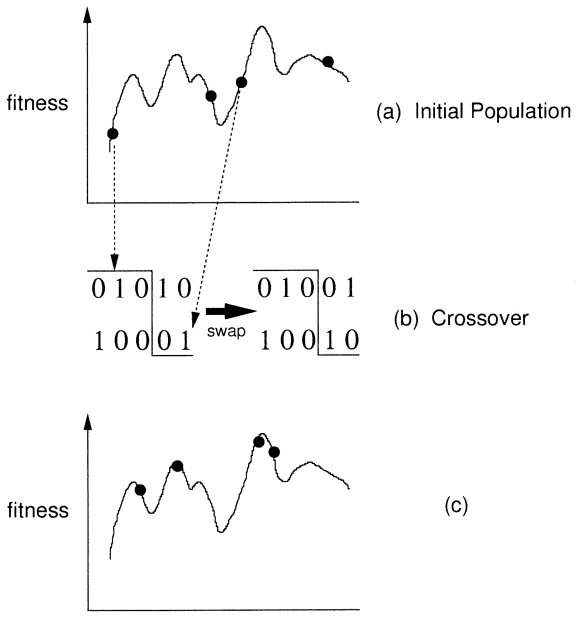

Figure 1: Schema of genetic algorithm

2 GENETIC ALGORITHM

2.1 Basic GA

GAs are robust optimization algorithms that are loosely based on population genetics. In GA optimization, a set of candidate solutions, represented as binary bit strings, is prepared. Each individual candidate is called a *chromosome* and the set of candidates is called a *population.* The problem to be solved is defined in terms of an evaluation function, called the *fitness function*, which is used to evaluate the chromosomes. A chromosome evaluated as having a high fitness value is likely to be a good solution of the problem. In the case of parameter extraction, model parameters are treated as GA chromosomes and the fitness function is defined in terms of the quality of fit for measured data. By repeating GA operations, such as *selection, crossover and mutation* to the population, a chromosome with high fitness value emerges.

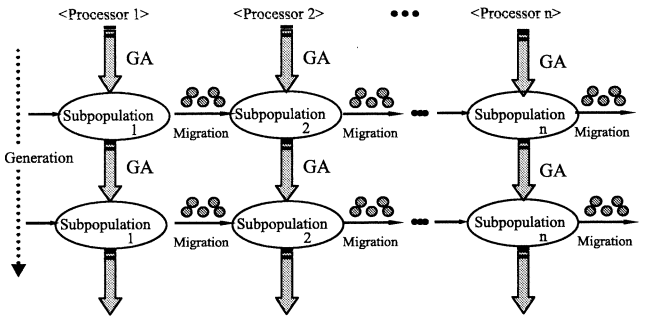

Figure 2: Schema of the distributed genetic algorithm

2.2 Distributed GA

Although GAs have much potential as a parameter extraction method, one drawback to the use of conventional GA for this is the considerable computational cost involved. For example, in the calibration of the B implantation model, it is necessary to optimize as many as 144 parameters simultaneously. In the parameter-extraction experiments, reported below, conventional GA required a few hours to optimize the parameters. To overcome this difficulty, we propose a distributed GA-based calibration technique. In the distributed GA[5][6], the following steps are performed.

1. Initialization: The total population is divided into several subpopulations (islands).
2. GA: A simple GA is executed for each sub-population for several iterations.
3. Migration: After several iterations, some chromosomes are selected and moved to one of the other islands. After this migration step, the process returns to step 2.

This algorithm can speed up GA computation significantly because it can easily be implemented on coarse-grain parallel computers, such as PC cluster systems or multi-processor workstations. Moreover, this algorithm can avoid local minimums at lower computational costs than the single population model[7]. As the range of genetic interaction (crossover) is limited to sub-population members, the diversity of the chromosomes is maintained to avoid premature convergence.

3 CALIBRATION OF IMPLANTATION MODEL PARAMETERS

We have applied this GA method to the calibration of the B implantation model towards the development of a decanano CMOS era. B profile is represented as a dual-Pearson profile[8] with 9 parameters for each profile (Figure 3). We prepared 16 profiles for different implantation energies (2keV–1000keV). B profiles (as implanted) are taken from a well-calibrated Monte Carlo implantation model, covering the low energies used in decanano Tr. processes.

$$f_{pearson}(x) = rf_1(x) + (1-r)f_2(x) \quad \text{where} \int f(x)dx = dose$$

$$\frac{df_i(s)}{ds} = \frac{(s-a_i)f(s)}{b_{0i} + b_{1i}s + b_{2i}s^2} \qquad s = x - R_{pi} \qquad \text{(i=1,2)}$$

$$\begin{cases} a_i = -\gamma_i \sigma_{pi}(\beta_i + 3)/A_i \\ b_{0i} = -\sigma_{pi}^2(4\beta_i - 3\gamma_i^2)/A_i \\ b_{1i} = a_i \\ b_{2i} = -(2\beta_i - 3\gamma_i^2 - 6)/A_i \end{cases} \quad \text{where} \quad \begin{array}{l} A_i = 10\beta_i - 12\gamma_i^2 - 18 \\ b_{1i}^2 - 4b_{0i}b_{2i} < 0, \ b_{2i} < 0 \end{array} \quad \text{(i=1,2)}$$

Model parameters: $r, R_{p1}, R_{p2}, \sigma_{p1}, \sigma_{p2}, \gamma_1, \gamma_2, \beta_1, \beta_2$

Figure 3: Dual-Pearson profile: Each profile has 9 parameters to be optimized.

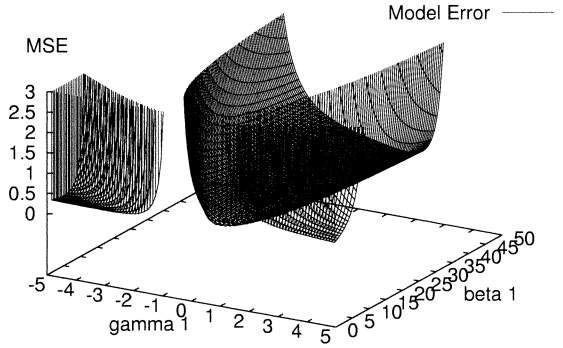

Figure 4: Model error versus values of dual-Pearson profile parameters γ_1 and β_1: This plot was obtained by varying the values of γ_1 and β_1, while holding all the other parameters constant. Although there are many local minimums in the nine-dimensional parameter space, few can be seen in this landscape because this plot is two-dimensional.

Figure 4 shows an example of the local minimum in this task. Clearly, there are some local minimums in the error surface, which would mislead traditional gradient-based techniques. In addition, this calibration is more complex because parameters must form a smooth curve across the range of implantation energies. In Figure 5 model parameters extracted by the LM algorithm are plotted versus energy. These do not form a smooth curve because the LM algorithm extracts the parameters separately for different energies. In order to achieve a smooth curve, we propose the GA method to calibrate all the parameters for all profiles simultaneously. Figure

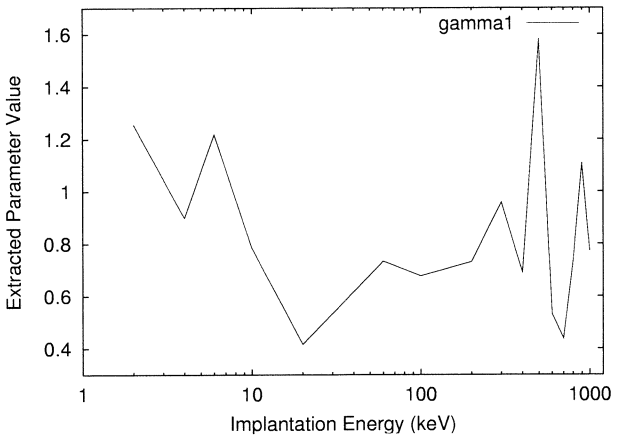

Figure 5: Model parameters extracted by the LM algorithm versus implantation energy

6 shows the flowchart for the fitness evaluation in our GA calibration. Using this fitness not only evaluates the smoothness of the curve over a range of energies but also reduces computational time significantly, because the slow local optimization with GA can be compensated for by combining this with the LM algorithm. A fitness value is calculated as follows:

1. The GA chromosome represents initial values (start points) in the LM algorithm for all profiles.

2. The LM algorithm independently calibrates 9 parameters for each profiles.

3. The calibrated parameters are modified to fit to the smooth function $g_i(E)$:

$$g_i(E; m_{1i}, m_{2i}, m_{3i}) = m_{1i}E^{m_{2i}} + m_{3i} \quad (i = 1, \cdots, 9) \tag{1}$$

where E is the implantation energy, m_{1i}, m_{2i}, m_{3i} are fitting parameters, and i is an index number for the nine parameters. This function is determined independently for each parameter by a conventional fitting method.

4. The mean squared error (MSE) for each profile is calculated using the modified parameters by Equation 1.

5. The fitness value is set to the summation of the MSE for each profile.

As this fitness function has so many local minimums, it is difficult for traditional gradient-based methods to optimize.

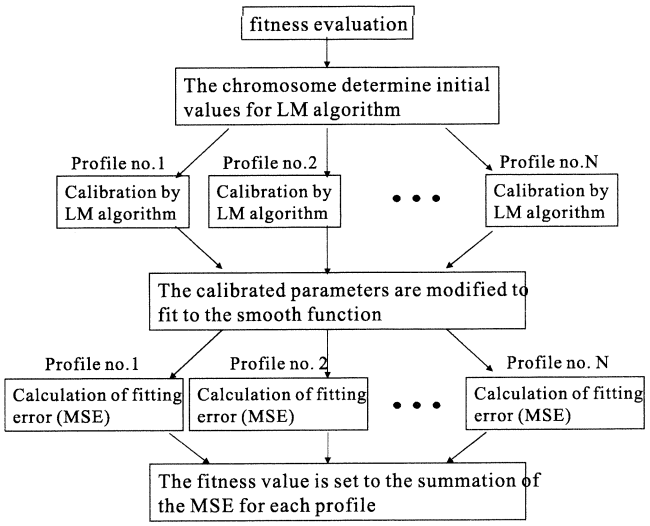

Figure 6: Flowchart of the proposed GA fitness evaluation

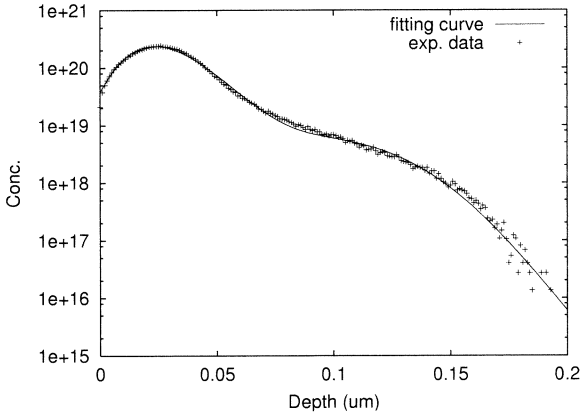

Figure 7: Calibrated profile for an implantation energy of 6keV

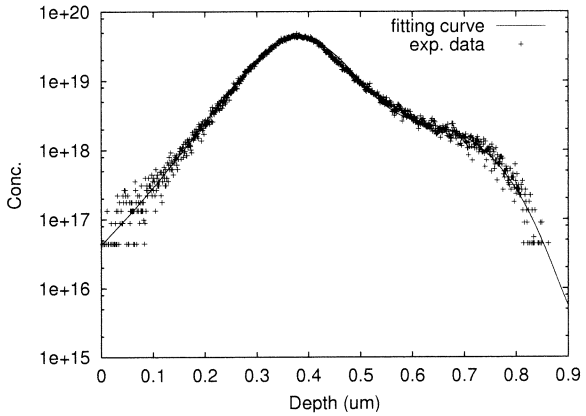

Figure 8: Calibrated profile for an implantation energy of 100keV

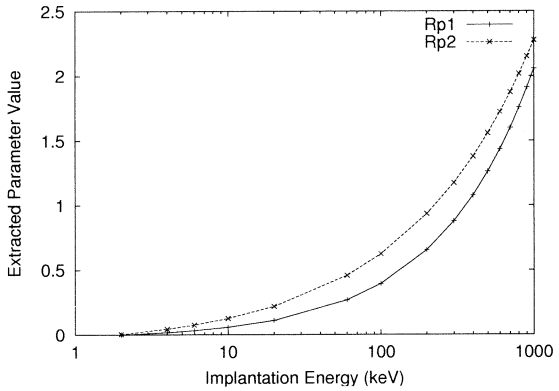

Figure 9: Extracted R_{p1}, R_{p2} versus implantation energy

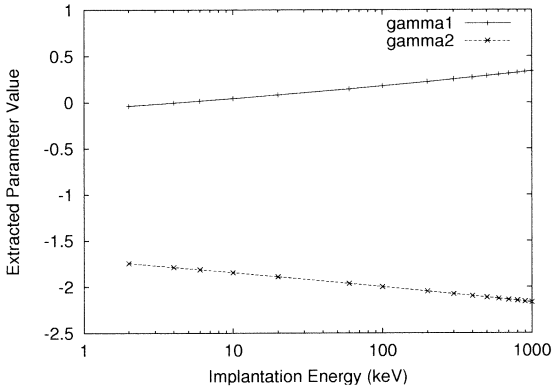

Figure 10: Extracted γ_1, γ_2 versus implantation energy

Optimization Method	LM algorithm	Conventional GA	Proposed GA	Proposed GA with 2 Processors	Proposed GA with 16 Processors
Computation Time (sec)	×	7167	943	494	61 (estimation)

Table 1: Comparison of computation times to calibrate 144 parameters. The experiments were performed on a PC workstation (CPU: Pentium IV 2.2GHz X 2, Memory 512MB, OS: Linux 2.4.5)

4 CALIBRATION RESULTS

In the experiments, parameter values in the chromosome were restricted as follows:

$$\begin{array}{ll} 0.0 \leq R_{pi} \leq 2.0 & 0.0 \leq \sigma_{pi} \leq 2.0 \\ -5.0 \leq \gamma_i \leq 5.0 & 0.0 \leq \beta_i \leq 50.0 \end{array} \quad (2)$$

In the calculation of the MSE, the errors around the profile peak were weighted by 10.0. The population size for the GA was 50. Figures 7 and 8 show calibration results for implantation energies of 6keV and 100keV, respectively. The calibrated dual-Pearson profiles clearly fit well to the data. Figures 9 and 10 show extracted parameters R_p and γ plotted versus implantation energy. The parameters are successfully extracted to form a smooth curve. A comparison of computation times with the LM algorithm, a conventional GA, and our GA is given in Table 1. The computation time of our GA outperformed the conventional GA by 7.60 times. Moreover, our distributed GA could be speeded up by 1.91 times with 2 CPUs and 15.46 times (estimation) with 16 CPUs. The LM algorithm could not calibrate the parameters to form a smooth function without human intervention.

5 CONCLUSION

We have proposed a distributed GA-based calibration technique combined with a traditional local optimization algorithm for process model calibration. By using this method, the calibration time was significantly speeded up, approaching near-optimal calibration with more than 100 parameters automatically. We are currently conducting calibration experiments on a PC cluster system. The proposed method can be applied to many calibration and optimization tasks, which take several weeks to complete with 70nm technology, bringing a drastic change for TCAD applications.

REFERENCES

[1] ITRS, *ITRS roadmap 2001*, p. 432, 2001.

[2] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, p. 542, Cambridge University Press, 1988.

[3] J. H. Holland, *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, 1975.

[4] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley, 1989.

[5] R. Tanese, "Distributed genetic algorithms," in *Proceedings of the Third International Conference on Genetic Algorithms*, 1989, pp. 434–439.

[6] E. C. Paz, "A summary of research on parallel genetic algorithms," IlliGAL Report 97003, University of Illinois, 1997.

[7] V. S. Gordon and D. Whitley, "Serial and parallel genetic algorithms as function optimizers," in *Proceedings of the Fifth International Conference on Genetic Algorithms*, 1993, pp. 177–183.

[8] C. Park, K. M. Klein, and A. F. Tasch, "Efficient modeling parameter extraction for dual pearson approach to simulation of implanted impurity profiles in silicon," *Solid-State Electronics*, vol. 33, no. 6, pp. 645–650, 1990.