Parallel Dynamic Load Balancing for Semiconductor Device Simulations on a Linux Cluster

Yiming Li[*], Shui-Sheng Lin[**], Shao-Ming Yu[***], Jinn-Liang Liu[**], Tien-Sheng Chao[****], and S. M. Sze[*]

[*]Department of Electronics Engineering and Institute of Electronics
[**]Department of Applied Mathematics; [***]Department of Computer and Information Science
[****]National Nano Device Laboratories
National Chiao Tung University, 1001 Ta Hsueh Rd., Hsinchu 300, Taiwan, ymli.ee87g@nctu.edu.tw

ABSTRACT

A parallel dynamic load balancing for 2-D and 3-D semiconductor device simulations is proposed. The hydrodynamic and drift diffusion models are discretized and solved with finite volume (box) and monotone iterative methods. Dynamic load balancing for parallel domain decomposition as well as parallel I-V point simulations are demonstrated to be efficient methods for multidimensional device simulations. Compared with the measured data, numerical results and benchmarks on a realistic N-MOSFET device are presented to show the accuracy and efficiency of the method. The code has been successfully implemented on a Linux-cluster with message passing interface (MPI) library.

Keywords: Parallel semiconductor device simulations, Load balancing, Domain decomposition, I-V point calculations

1 INTRODUCTION

The parallel simulation of multidimensional VLSI devices has been proven to be an indispensable tool for the analysis and optimal design of various semiconductor devices (See [1-5] and references therein). These computations are received considerable attention and become extremely attractive to provide effective ways in TCAD/CAD applications. Parallelization of numerical simulations with adaptive meshing is a complex task, but it could be improved upon by a more efficient simulation methodology and a very flexible and modular software design approaches.

In this paper, a new parallel semiconductor device simulation using dynamic partition is presented. Dynamic load balancing for parallel domain decomposition and parallel I-V point simulations are demonstrated to be efficient methods for multidimensional device simulations. The semiconductor device model are discretized and solved with finite volume (box) and monotone iterative methods [6-9]. The new approach for solving a set of fundamental semiconductor device equations, such as drift diffusion (DD) and hydrodynamic (HD) models [3, 4, 10-12] has been successfully developed and implemented on a Linux cluster. The computation produces load imbalances among processors thereby creating the need for repartitioning and rebalancing of the workload, so a physical based parallel adaptation and dynamic load balancing algorithm are also described. When a refined tree structure is created, the number of processor for next computing will be dynamically assigned and allocated following the total number of vertices firstly. Then a geometric dynamic graph partitioning method in x-, y- or z-direction is applied to partition the number of vertices to each processor. The parallel methods developed and implemented in this paper show these approaches are feasible parallel computing alternatives that can be used to provide fast characterization of multidimensional semiconductor devices.

In the next section, the studied semiconductor device model is briefly outlined. Section 3 is devoted to a discussion on the dynamic load balancing for parallel domain decomposition and parallel I-V point calculations methods. In Section 4, simulation results and measurement data for a submicron N-MOSFET device are presented to demonstrate the accuracy and efficiency of the method. Results included the achieved speedup and related performances are also given in this section. The main conclusions of this work are given in the last section.

2 SEMICONDUCTOR DEVICE MODEL

In this section, we briefly review a hydrodynamic model [11] that consists of the Poisson equation, the carrier current continuity equation, and the carrier energy conservation equation. The electron and hole hydrodynamic equations can be derived from the first three moments of the Boltzmann Transport Equation (BTE) [11]. The zeroth, first, and second moments of the BTE correspond to the particle continuity, conservation of momentum, and energy, respectively. In the steady-state, for studying the behavior of a submicron MOSFET device, as shown in Figure 1, we present the conventional HD model for electrons as follows:

$$\Delta\phi = \frac{q}{\varepsilon_S}(n - p + D), \tag{1}$$

$$\frac{1}{q}\nabla \cdot J_n = G - R, \tag{2}$$

$$\nabla \cdot S_n = J_n \cdot (-\nabla\phi) - n\left(\frac{\omega_n - \omega_0}{\tau_{nw}(T_n)}\right), \tag{3}$$

$$J_n = -q\mu_n n\nabla\phi + qD_n\nabla n + \mu_n k_B n\nabla T_n, \quad (4)$$

$$S_n = \frac{-J_n}{q}\omega_n + \frac{-J_n}{q}k_B T_n + Q_n, \quad (5)$$

where (1) is the Poisson equation, (2) is the electron current continuity equation, and (3) is the electron energy conservation equation. The $D=N_D^+-N_A^-$ is the net doping concentration, the generation and recombination process G-R is due to the collision terms. The G is the avalanche generation term and the recombination term R is the sum of Schockley-Read-Hall and Auger recombination [10]. The J_n and S_n, in equations (4) and (5), are the electron current density and energy flux, respectively [13]. The average electron energy $\omega_n=3/2(k_BT_n)+1/2(m^*_nv^2_n)$ and the heat flux Q_n in (5) followed the Fourier law is

$$Q_n = -\kappa_n\nabla T_n, \quad (6)$$

where the electron thermal conductivity κ_n is given by the Wiedemann-Franz law as

$$\kappa_n = 2T_n\frac{n\mu_n k_B^2}{q}. \quad (7)$$

Here, T_n is the electron temperature and μ_n is the electron mobility.

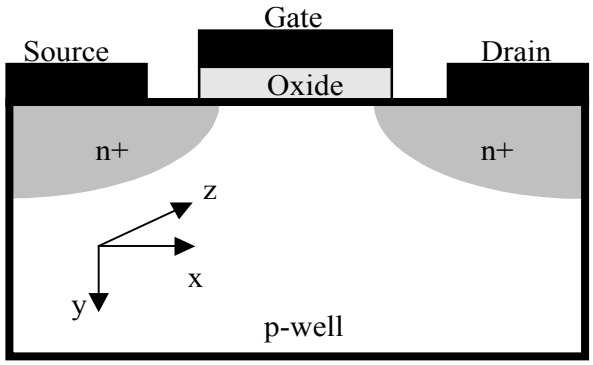

Figure 1: Schematic diagram of a MOSFET.

The DD model is the first generation numerical model for semiconductor device simulation. It assumes local isothermal conditions and is widely employed in semiconductor device design. The DD model consists only of the Poisson equation (1) and electron current continuity equation (2) with reduced current equation (4). The equations (1)-(3) are decoupled with Gummel's decoupling method [5]. Next, 2-D HD and DD models are discretized with finite volume (box) method on an unstructured mesh and 3-D DD is discretized with finite volume (box) method on a structured mesh. The basic HD and DD models for holes are similar to equations (1)-(5).

3 PARALLEL DYNAMIC LOAD BALANCING

A parallel dynamic load balancing for domain decomposition as well as I-V point calculations are discussed here. The discretization of HD and DD models leads to a system of nonlinear algebraic equations and directly solved with monotone iterative algorithm:

$$(D+\lambda I)Z^{(n+1)} = (L+U)^{(n)} - F(Z^{(n)}) + \lambda Z^{(n)}, \quad (8)$$

where Z is the unknown vector, F is the nonlinear vector form, and D, L, U, and I are diagonal, lower triangular, upper triangular, and identity matrices, respectively. And the iteration parameter λ depends on the device structure, doping concentration, bias condition, and nonlinear property of each decoupled equation [9]. Note that the algorithm is one of Jacobi types and hence is highly parallel.

The domain decomposition has recently received great popularity for solving system of linear equations on parallel and distributed computers. It has been applied to the semiconductor device simulation [1, 3]. The simulation domain is partitioned into several disjoint sub-domains, and interface iteration and data exchange algorithms should be used to find the complete solution. In this work, a parallel load balancing for not only domain decomposition but also I-V point calculations is proposed and successfully implemented on a Linux-cluster with MPI library.

As shown in Figure 2, based on the 2-D or 3-D device structure and bias condition the simulation domain is dynamically partitioned into m disjoint sub-domains. When a refined tree structure is created, the number of processor for next computing will be dynamically assigned and allocated following the total number of nodes firstly. A geometric dynamic graph partitioning method in x-, y-, or z-direction is then applied to partition the total number of nodes and assign those partitioned nodes to each processor. The partitioned sub-domain is solved with equation (8).

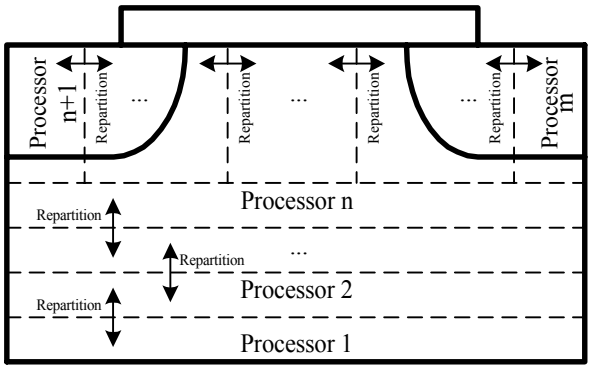

Figure 2: An illustration of parallel dynamic partition of domain decomposition for a 2-D MOSFET.

A detail computational procedure for parallel domain decomposition is as follows: (i) Initialize the MPI environment and configuration parameters. (ii) Based on unstructured 1-irregular or structure meshing rule, a tree structure and mesh are created. (iii) Count number of meshes and applies a dynamic partition algorithm to determinate how many processors are required in this simulation. All nodes are numbered, besides that the boundary and critical points are identified. (iv) All assigned jobs are solved with equation (8). The computed data communicates by the MPI protocol. (v) Do convergence test for all elements and run the adaptive refinement for those needed elements. (vi) Repeats steps (iii)-(v) until the error of all elements is less than a specified error bound. (vii) Host processor collects all computed results and stops the MPI environment.

The load balancing dynamic partition algorithm used in step (iii) is outlined as follows: (a) Count the number of total nodes. (b) Find out the optimal number of processors based on the node numbers and an empirical formula. (c) Calculate how many nodes should be assigned to each processor by dividing total nodes with the optimal number of processors. (d) Along x, y, or z direction in 2-D or 3-D device domain, search (from left to right, bottom to top, and front to back) and assign nodes to these processors sequentially. Repeats this step until all nodes have been assigned. (e) In the neighborhood of p-n junction, if needed, one may change search path for obtaining a better load-balancing performance.

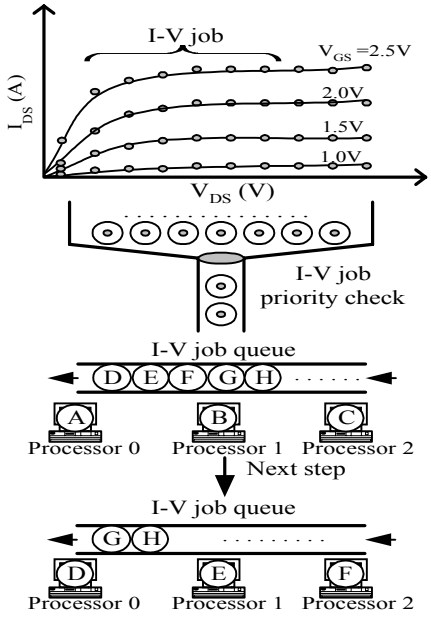

Figure 3: An illustration of parallel dynamic I-V job partition and assignment for device terminal characteristics calculations.

The parallel load balancing for I-V point calculations, as shown in Figure 3, is also proposed and implemented on a Linux-cluster with MPI library. The processes of parallel I-V characteristic point calculation are stated here. They are: (I) Initialize the MPI environment and configuration parameters. (II) Server creates required processor; each processor has its own client. (III) Server sends out all scheduled I-V points to processors, each processor communicates with client. (IV) Client calculates assigned I-V points with equation (8) independently. (V) If the job is done, client sends the data back to server and call for next computation. (VI) Repeats (III)-(V) until all jobs are done. (VII) Stops the MPI environment. Figure 3 shows that the I-V points are computed independently and it's different from the widely used continuous method in the device simulation.

4 RESULTS AND DISCUSSIONS

Figure 4 demonstrates the simulated electrostatic potential in 3-D simulation. The simulated electrostatic potential with 3-D DD model is plot for L_{gate}=0.35μm N-MOSFET at V_{DS}=2.0V and V_{GS}=2.0V. The left color bar is for x-y plane and the other is for y-z plane. The 3-D DD model simulation used finite volume discretization with structure tetrahedral mesh. As shown in Figure 5, comparison of measured and calculated I-V curves for the same N-MOSFET presents the accuracy between a 2-D HD simulation and measurement. For the 2-D HD simulation, the finite volume discretization with 1-irregular mesh is applied in this work. All of the HD simulated I-V points in Figure 5 are computed independently by the proposed parallel I-V point calculations method.

Tables 1-3 contain the parallel simulation timing results for the proposed algorithms in 3-D DD model simulation. Table 1 gives speedup performance results of dynamic load balancing approach for domain decomposition. A speedup of 7.06 (mesh size=256k) is obtained on an 8 Pentium-III processors Linux-cluster with MPI library. The performance of dynamic load balancing for 3-D domain decomposition approach is also shown in Table 2. Maximum difference is defined as the maximum ratio of the code execution time difference divided by the maximum code execution time [14]. With the same Linux-cluster computing system, Table 3 shows a similar speedup of 7.166 for 3-D parallel I-V point calculations method.

5 CONCLUSIONS

A parallel dynamic load balancing for 2-D and 3-D submicron semiconductor device simulations has been proposed and successfully implemented on a Linux-cluster with MPI library. Dynamic load balancing for parallel domain decomposition and parallel I-V point simulations have been demonstrated to be efficient methods for multidimensional submicron device simulations. Compared with the measured device I-V data, numerical results and benchmarks on a submicron N-MOSFET have been presented to show the accuracy and efficiency of the method.

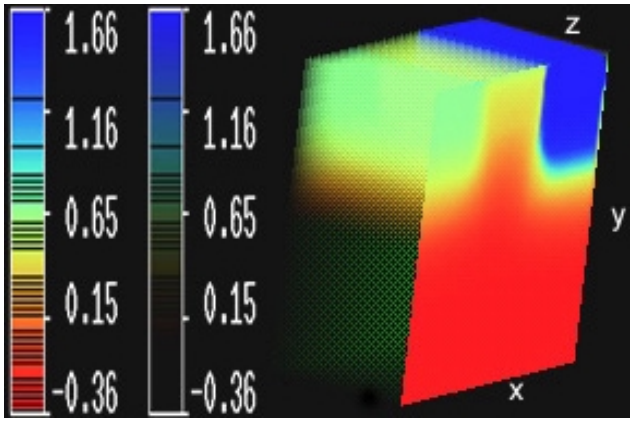

Figure 4: Simulated electrostatic potential for L_{gate}=0.35μm N-MOSFET at V_{DS}=2.0V and V_{GS}=2.0V in 3-D DD simulation. The left color bar shows the scale in x-y plane and the other is for y-z plane.

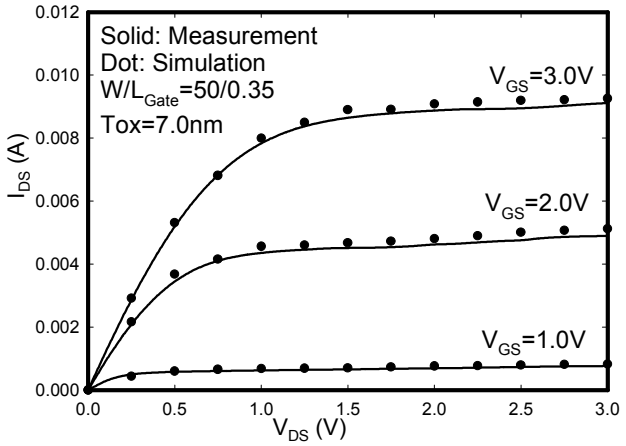

Figure 5: Comparison between 2D HD simulated (Dot) and measured (Solid) drain current versus drain voltage curves for L_{gate}=0.35μm N-MOSFET. All I-V points are calculated with the proposed parallel I-V point calculation method.

Table 1: Parallel time and speedup performance for 3-D domain decomposition with dynamic partition on an 8 Pentium-III processors Linux-cluster with MPI. The applied voltage for the test N-MOSFET is V_{DS}=2.0V and V_{GS}=2.0V.

Mesh Size	Sequential time (Sec.)	Parallel time (Sec.)	Speedup	Efficiency
64K	2260	340	6.65	83.1%
100K	5200	770	6.75	84.4%
145K	10500	1540	6.82	85.3%
200K	22160	3160	7.01	87.6%
256K	31140	4410	7.06	88.3%

Table 2. Load balancing for 3-D domain decomposition with dynamic partition on the Linux-cluster system. The applied voltage is V_{DS}=2.0V and V_{GS}=2.0V.

Mesh Size	Parallel time (Sec.)								Max. differ.
	CPU #0	CPU #1	CPU #2	CPU #3	CPU #4	CPU #5	CPU #6	CPU #7	
64K	340	340	340	340	340	340	340	340	0.00%
100K	770	760	770	760	770	750	770	770	2.60%
145K	1540	1510	1540	1540	1520	1520	1530	1540	1.95%
200K	3160	3160	3140	3120	3060	3100	3080	3160	3.16%
256K	4410	4260	4410	4310	4380	4400	4380	4400	3.40%

Table 3. Parallel speedup for parallel I-V point simulations on the Linux-cluster. There are total 9 I-v curves in the 3-D DD model simulation. The applied voltage is V_{DS}=0.0, 0.5, …,3.5, 4.0V and V_{GS}=0.0, 0.5, …,3.5, 4.0V. The mesh size is from 25K to 256K and is adaptive generated corresponding to a posterior error estimation.

Number of processors	1	2	4	8
Parallel time (Min.)	3834	2049	1052	535
Speedup	--	1.871	3.644	7.166
Efficiency	--	93.55%	91.10%	89.58%

REFERENCES

[1] A. J. Garcia-Loureiro, et al., J. Modeling Simulation Microsystems, 1, 2, 115, 1999.

[2] Y. Yuan and P. Banerjee, Proc. Int. Conf. IPDPS, 323, 2000.

[3] G. Antonoiu,.et al., Proc. Int. Semicond. Conf., 2, 371, 1998.

[4] N. R. Aluru, et al., IEEE Trans. CAD, 15, 9, 1029, 1996.

[5] J.-L. Liu, et al., Proc. Int. Conf. CSCC-MCP-MCME, 4, 199, 2000.

[6] I. Babuska and W. C. Rheinboldt, SIAM J. Numer. Anal., 15, 736, 1978.

[7] J. R. Whiteman ed., Mathematics of Finite Elements and Applications, Wiley, 1994.

[8] J.-L. Liu, et al., Appl. Num. Math., 21, 439, 1996.

[9] Y. Li, et al., Proc. Int. Conf. VLSI-TSA, 27, 1999.

[10] S. M. Sze, Physics of Semiconductor Devices, 2nd Ed., Wiley-Interscience, 1981.

[11] K. Blotekjer, IEEE Trans. Electron Devices 17, 38, 1970.

[12] M. Ieong and T.-W. Tang, IEEE Trans. ED., 44, , 2242, 1997.

[13] A. Forghieri, et al., IEEE TCAD., 7, 231, 1988.

[14] K. Dowd and C. Severance, High Performance Computing, O' Reilly, Sebastopol, 1998.