(9B-1)

PARALLEL COMPUTATION METHOD IN DEVICE SIMULATION

Kazuhiro MOTEGI[*] *Shigeyoshi WATANABE*[†]

Gunma University

1. Introduction

The fast and reliable computation method of linear equations is desired to build a device simulator because of its convergence in the computation. For the limitation of computation time the direct solution method of linear equations is hard to be applied to the device simulation.

Recently, multiprocessor parallel computer were manufactured and begun to be used actually in several fields. But, there happens a lot of problems, e.g., in an algorithm of a numerical analysis and in an organization of a hardware. One of the most important problems to be solved is to implement the algorithm of a parallel computation of a sparse matrix.

Multi Step Diakoptics (MSD)[1][2] extended from Kron's Diakoptics by one of authors is a parallel computation technique that solves linear equations with the direct method. MSD is not a partition method of a matrix but the parallel computation method partitioned a linear problem . The algorithm of MSD is applicable to a circuit simulation and a device simulation.

This paper describes the algorithm of MSD for the device simulator and shows computational examples executed on a real MIMD(Multiple Instructions stream Multiple Data stream) type parallel computer named a Cellar Array Processor(CAP). The results show the efficiency of MSD in the device simulation.

2. Basic Semiconductor Equations

We use Selberherr's device models and make the basic semiconductor equations - Poisson equation and carrier(electron and hole) equations[3]. These are partial differential equations that have unknown variables - ψ(internal potential), n(electron density) and p(hole density):

$$\lambda^2 \nabla\psi(n - p - c) = 0, \quad (1)$$

$$\nabla \cdot (D_n \nabla n - \mu_n n \nabla\psi) - GR(\psi, n, p) = 0, \quad (2)$$

$$\nabla \cdot (D_p \nabla p + \mu_p p \nabla\psi) - GR(\psi, n, p) = 0, \quad (3)$$

where c is a net impurely density, $D_n(D_p)$ is an electron(hole) diffusion coefficients, $\mu_n(\mu_p)$ is an electron(hole) mobilities, GR is a carrier generation rate and λ is a normalization factor.

The decouple method is used to solve these equations. Each equation is linearized by Newton-Raphson method, is discretized by the finite difference method and are transformed to the system of the linear equations. Then, we obtain:

$$A\delta\phi = b, \quad (4)$$

where A is a sparse and a band matrix, $\delta\phi$ is a solution vector ($\phi = \psi$, n or p) and b is a right hand vector. Here, we define an internal iteration as an iteration to transform the nonlinear equation to the linear equations, and an external iteration as an iteration to compute the system of linear equations.

[*]Student of Doctor Course, Faculty of Engineering, Gunma University, Kiryu-shi, 376, Japan

[†]Associate Professer, Faculty of Engineering, Gunma University, Kiryu-shi, 376, Japan

3. Diakoptics

As described in reference[2], Diakoptics is the branch tearing method of a grid structure. The grid structure is made by the five points discretization of partial differential equations(1), (2) and (3). The structure is partitioned into several subdomains by tearing branches connected between subdomains. The connected relation between subdomains can be expressed by an incident matrix of C. The element $c_{i,j}$ of C shows the incident relation between a grid and a branch. For example, $c_{i,j} = 1$ if i grid is incident to j branch.

The matrix A of (4) is equivalently expressed by the form of

$$A = Ao + CZ^{-1}C^t, \quad (5)$$

where, Ao is a block diagonal matrix and Z is a diagonal matrix. Each block of Ao is made by the difference equation of the subdomain. In the case, the outer grids of the subdomain are treated as the boundary condition of the differential equations. The values of elements of Z are determined by the off-diagonal elements of A (see reference[2]).

Diakoptics is applied to solve (4) by the equivalent direct method of

$$\delta\phi = Ao^{-1}b - Ao^{-1}C(Z + C^t A_o^{-1} C)^{-1} C^t Ao^{-1} b \quad (6)$$

The first term on right-hand side expressions in (6) can be solved in parallel because of Ao. The most difficult problem of Diakoptics is caused by the second term, i.e., the solution of the matrix of

$$W = Z + C^t A_o^{-1} C \quad (7)$$

The MSD is a solution method of the problem.

4. M. S. D.[4]

The algorithm of MSD is composed of three phases - the first phase is the calculation of subdomain problems in parallel, the second phase is the data communication between incident clusters, and the third phase is the modification of the solutions of subdomains.

Let us explain in detail. Each processor in the parallel computer executes the task of a subdomains. The task makes three linear equations derived from (1),(2) and (3), the incident matrix $C_{p,r}$ and the diagonal matrix Z_r. Where p is the label of the subdomain and r is the label of the group of branches incident to p subdomain. We combine branches that connect two subdomains to a group and call it a connector.

Let us denote R_p is a group composed by connectors incident to p subdomain.

The schedule of parallel computation is done by selection of connectors step by step. At every step, at most one connector in R_p is selected in any p and the selected connector is removed from R_p. In the first step, two subdomains are connected and are made a cluster. The cluster is a group of subdomains. The cluster including p subdomain is denoted by S_p. In the following steps, two subdomains in the same cluster or in the different

clusters are connected. Finally, all subdomains are connected (the original grid structure is restored). The rule of selection is

1. in order to reduce the total number of steps, more than one connector may be selected in every step.

2. at every step, only one connector is connected in one cluster.

The first phase is a subdomain problem- solving phase and linear equations

$$A_p\phi_p = b_p,\ A_pU_{p,r} = C_{p,r}\ r\ in\ R_p \qquad (8)$$

are solved by p processor that computes the task of p subdomain. The second phase is a communication phase. According to the schedule, p processor transfers the data of ϕ_p and $U_{p,r}$, r in R_p to all of the processors that compute tasks of subdomains in S_p. The third phase is a modification phase. When the p processor receives the data at every step, the following calculations are done in the this phase.

$$\phi_p = \phi_p - U_{p,k}W_k^{-1}\sum_f C_{f,k}^t\phi_f \qquad (9)$$

$$U_{p,r} = U_{p,r} - U_{p,k}W_k^{-1}\sum_f C_{f,k}^t U_{f,r}\ r\ in\ R_p \qquad (10)$$

where, k is a connector connected to the p subdomain and f is two domains incident to k connector. The phase 2 and 3 are repeated until final step of the schedule.

5. Results of Computation[5]

Let us denote the computation time of each phase is T_{sub}, T_{com} and T_{mod}, respectively. The total computation time is defined as the sum of T_{sub}, T_{com} and T_{mod}.

At first, we use a quasi-parallel simulator CSS(CAP Software Simulator). CSS can work on a workstation and simulate the parallel computation by assigning all tasks of the parallel computation into processes of the workstation. Therefore, it is possible to measure the computation time of any task on CSS. Figure 1 shows the relation between the number of the discretized grids and the total computation time. The parameter $2^m \times 2^n$ in Figure 1 denotes the number of the subdomains that are made by tearing in 2^m parts on X-axis and in 2^n parts on Y-axis. The figure shows that the effectiveness of MSD is increased in accordance with increasing the number of grids.

Next, we show the result of parallel computation executed on CAP[6]. The relation between the number of processors and grids is shown in Figure 2. The CAP is organized by a workstation constructed 32bits CPU, called host, and 64(= 8×8) processors constructed 16bits CPU, called cell, that are arranged in a grid shape. All cells are connected to the host by a common bus. The common bus is the data communication bus connecting all cells. Because CAP has a limitation of the memory size, the number of grids in any subdomain isn't able to be out of 64 grids in a diode model.

The results of Figure 2 doesn't show the efficiency in case of small size problems. Because of a few discretized grids, the sum of the second phase time T_{com} and the third phase time T_{mod} is greater than the first phase time T_{sub} in the algorithm of MSD. However, if total number of grids is over 2^9, the result on CSS shows the efficiency of MSD. Therefore, we conclude that the efficiency of MSD increases if the number of the grids is over 2^9[5].

6. Acknowledgment

We would be grateful for Dr. Mitsuo Ishii, a manager in a Fujitsu laboratories Ltd., to permit the use of parallel computer CAP and for everyone to instruct the utilization of CAP.

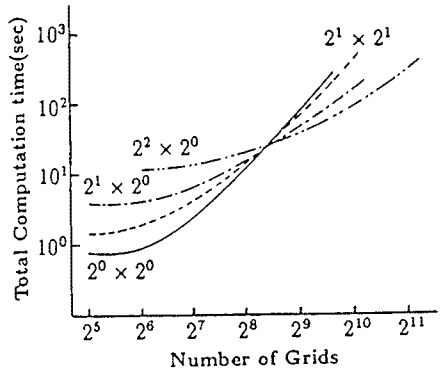

Fig. 1 Computation Time on CSS.

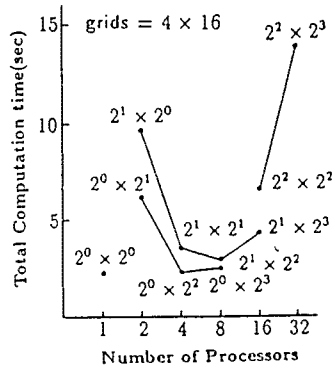

Fig. 2 Computation Time on CAP.

References

[1] S. Watanabe, T. Fukao: "Multi-Step Diakoptics and a Parallel Computation Method for Large Scale Circuit Network Problems", *Journal of IEICE of Japan(A)*, 70-A-2, pp.220-227(1987)

[2] T. Fukao, E. Teratsuji: "An Expansion of Multi-Step Diakoptics and an Application to Singular Partition Problems of a Circuit Network", *Journal of IEE of Japan(C)*, 102-C-2, pp.129-136(1982)

[3] S. Selberherr: "Analysis and Simulation of Semiconductor Device", *Springer-Verlag*(1984)

[4] S. Watanabe, T. Miyazaki, K. Motegi, T. Ode: "A Parallel Computation Method to Solve Poisson Equation with a Multi-Step Diakoptics", *Journal of the Japan Society for Simulation Technology* , Vol.8, 4, pp.49-57(1989)

[5] K. Motegi, S. Watanabe: "An Example of Parallel Calculations in Device Simulation", *the Japan Society for Simulation Technology, Proceeding of the 11-th Annual Symposium of the Institute of Computation, Electronics and Electrical Engineerings*, pp.223-228(1990)

[6] M. Ikeda, K. Horie, M. Ishii: "Cell OS for Cellular Array Processor CAP", *Technical Report of IEICE of Japan(A)* , CAS86-209, pp.43-50(1987)