SIMULATION OF SEMICONDUCTOR DEVICES AND PROCESSES Vol. 5
Edited by S. Selberherr, H. Stippel, E. Strasser - September 1993

PARDESIM — A Parallel Device Simulator on a Transputer Based MIMD-Machine

O. Kalz and D. Schröder

Technische Elektronik, TU Hamburg-Harburg
Eißendorfer Straße 38, D-21073 Hamburg, GERMANY

Abstract

In this paper we present a parallel 3D device simulator on a MIMD[1]-machine. We have investigated the efficiency of a parallel implementation, especially for the fine-grained parallelized iterative linear solvers like ILU-BCG[2]. As a result we have found that an efficient employment of some 10^1 processors is possible on a MIMD-computer with the same convergence behaviour as on an adequate large sequential computer.

1. Introduction

3D-semiconductor device simulation belongs to the field of numbercrunching data processing. This means that high computing performance and a large memory are necessary. Thus, vectorization and parallelism came early into the scope of interest for satisfying the increasing requests to solve the classical semiconductor equations on a discretization grid of 10 000... 250 000 points.

First at all, efficient vector algorithms for the most CPU consuming iterative or direct linear solvers, like ILU-CG, ILU-BCG, or CGS, have been developed and applied [1]. But future requirements can be satisfied only, or at least in a much cheaper way, by massively parallel computers [2]. Traditional shared memory machines can neither solve the memory problem nor allow a fully scalable program, which means an efficiency nearly independent of the number of processors.

Unfortunately, distributed memory algorithms are harder to develop especially under usage of fine grain parallelism, because one has to organize the data exchange between the nodes with regard to the limited communication bandwidth and a noticeable communication setup time.

In consideration of this background we will present the parallelization strategy of the parallel device simulator PARDESIM, which is under development[3] on a SuperCluster SC-128 multiprocessor of Parsytec. Some parameters of the transputer based parallel computer are listed in Table 1.

[1]Multiple Instruction Multiple Data
[2]Incomplete LU Decomposition and Biconjugate Gradient Method
[3]This work was sponsored by the "Deutsche Forschungsgemeinschaft"

2. Implementation of the parallel algorithm

In order to concentrate on parallel algorithms, we restricted hitherto the simulator to a steady-state solver of the traditional semiconductor equation system with two carrier equations on a rectangular tensor mesh in 2 or 3 dimensions [3]. While the discrete nonlinear system is solved successively within the *Gummel* iteration [4], we chose preconditioned gradient methods like ILU-CG and ILU-BCG for the linear systems [5].

To use the advantage of distributed memory computers, we have to implement a domain decomposition algorithm that runs on some 10^1 processors with a high speedup S_p and a sufficient efficiency E_p, defined as follows:

$$S_p = \frac{t_{serial}}{t_{parallel}} \quad \text{on p processors} \quad (1) \qquad\qquad E_p = \frac{S_p}{p} \quad (2)$$

We have measured that the iterative solution of the linear systems takes about 80% of total CPU time, which in turn divides into 40% for the ILU preconditioner, 40% for the matrix vector multiplications, and 20% for local vector operations without communication effort. The hardest problem is to make the incomplete LU decomposition and forward and backward substitution parallel for sparse band matrices, because the algorithm seems to be inherently sequential. But the special matrix block structure (e.g. Fig. 1) caused by the discretization scheme gives us the chance for a time-displaced ILU decomposition (by one row) of each block in parallel. For that purpose the processors are arranged in a ring topology and the matrix blocks are mapped on them in a natural ordering (Fig. 2a).

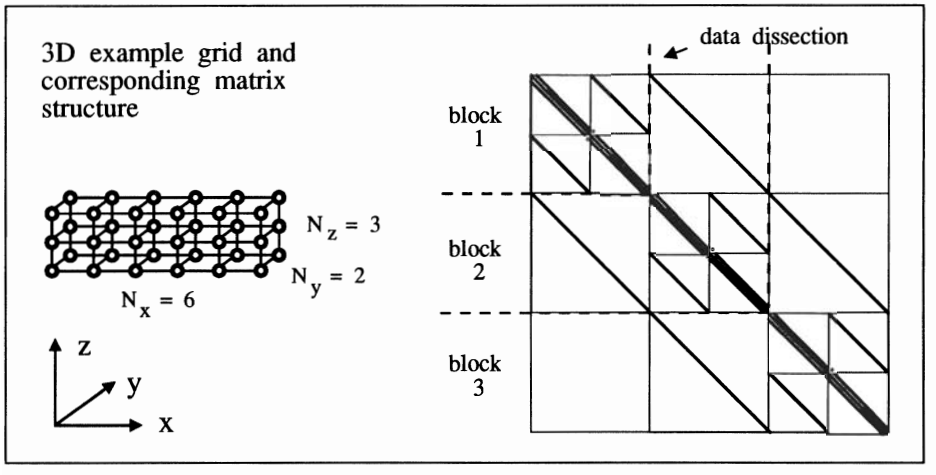

Figure 1: Nonzero matrix structure for an example grid

For the *incomplete* decomposition of the n-th row of one block only the results up to the (n-1)-th row of the *preceding* block and local block data are needed. The inclusion of the block coupling outer diagonals saves a good convergence behaviour but limits the degree of parallelism. As a theoretical maximum speedup resp. efficiency for ILU decomposition on N_B available processors without communication costs we find

$$S_{N_B,ILU} = \frac{N_B \cdot B_{Dim}}{B_{Dim} + N_B - 1} \quad (3) \qquad E_{N_B,ILU} = \frac{B_{Dim}}{B_{Dim} + N_B - 1} \quad (4)$$

where N_B is the number of blocks (N_z of a 3D-grid) and B_{Dim} is the block dimension ($N_x \cdot N_y$ of a 3D-grid).

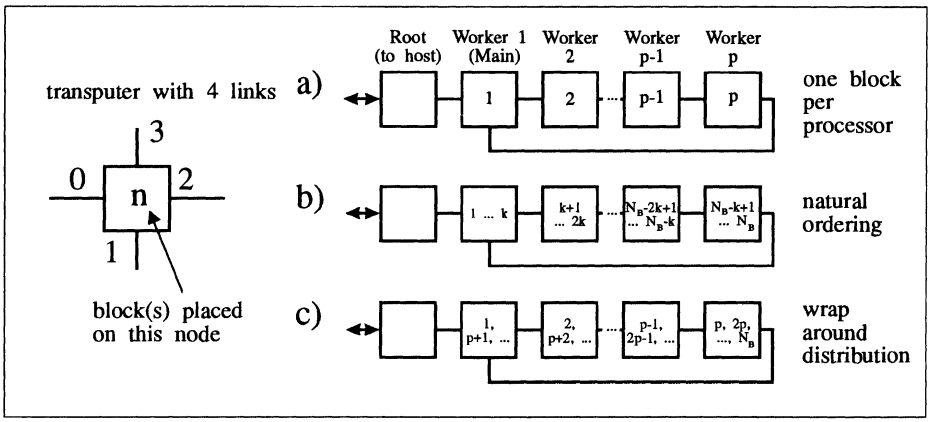

Figure 2: Topology of processor network and block mapping example

If there are more blocks than processors we use a mapping scheme that places neighbouring block-processes on the same processor (Fig. 2b), instead of a wrap around distribution (Fig. 2c) as usual for dense matrices. This profits from a partly drastic reduction of interprocessor communication effort and an increasing job size per processor. In this case, the estimations (3), (4) for $p = N_B$ have to be replaced by the following formulas for $p \leq N_B$, with k the number of blocks per processor.

$$S_{p,ILU} = \frac{B_{Dim} \cdot p}{B_{Dim} + p - 1} \quad (5) \qquad E_{p,ILU} = \frac{B_{Dim}}{B_{Dim} + p - 1} \quad (6)$$

From (6) one can see that the mapping of neighbouring blocks on the same processor does not decrease the ILU efficiency compared with the case of one block per processor (4). Nevertheless the maximum theoretical speedup can only be reached for a large block dimension B_{Dim} and is limited by N_B, i.e. the maximum number of grid coordinates in one space direction.

3. Numerical results

As a result we have found that the described parallelization strategy leads to a sufficient efficiency if the job size per transputer is large enough (Fig. 3). In Fig. 4 the measured efficiency of the critical forward backward substitution on 4 processors is compared with the theoretical maximum. Especially for $N_B = p = 4$, i.e. one block per processor, the estimation (6) is too optimistic because already few arithmetic operations require one communication step in this case. Finally Fig. 5 shows that a scalability of the complete linear solver on a different number of processors can be observed if the matrix size, i.e. the number of grid points, increases proportional to the number of used processors. Otherwise the efficiency decreases rapidly if the matrix size remains constant.

As a next implementation step the distributed matrix assembly and the nonlinear iteration scheme has to be implemented in parallel in order to achieve the aspired speedup for the complete simulator.

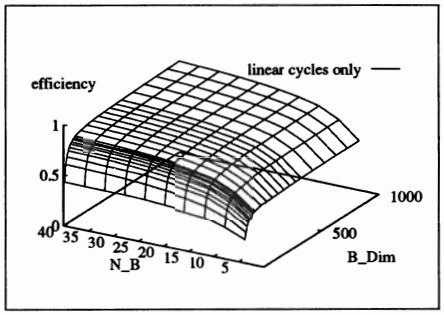

Figure 3: Efficiency $E_{4,ILU-BCG}$ of linear solver on 4 processors

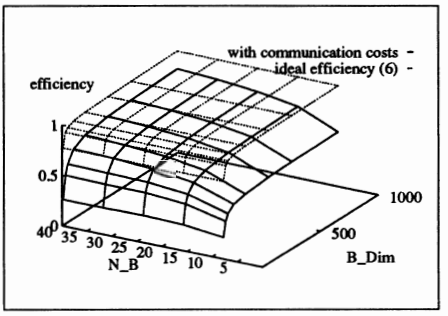

Figure 4: Efficiency $E_{4,ILU}$ of forward backward substitution

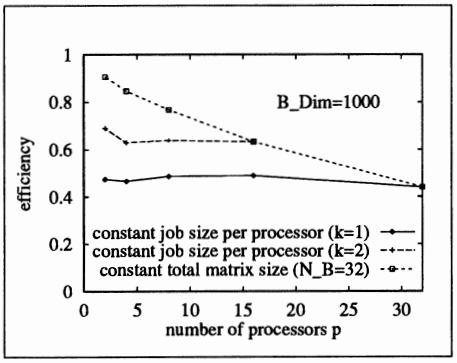

Figure 5: Efficiency $E_{p,ILU-BCG}$ of a variable number of processors

topology	free configurable of maximum 128 processors
CPU	Transputer T805 (INMOS) 32 bit RISC with FPU
clock	30 MHz
peak performance	about 2 MFLOPS per processor
memory	4 MByte + internal memory
communication	4 links per transputer (each 20 Mbit/s)
compiler	Par.C (C with parallel extensions)

Table 1: Parameters of multi processor system SuperCluster SC-128

References

[1] Heiser, G.; Pommerell, C.; Weis, J.; Fichtner, W.: Three-Dimensional Numerical Semiconductor Device Simulation: Algorithms, Architectures, Results, IEEE CAD-10 (1991), No. 10, p. 1218-1231

[2] Webber, D.; Tomacruz, E.; Guerrieri, R; Toyabe, T.; Sangiovanni-Vincentelli, A: A Massively Parallel Algorithm for Three-Dimensional Device Simulation, IEEE CAD-10 (1991), No. 9, p. 1201-1210

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

[4] Gummel, H.K.: A Self-Consistent Iterative Scheme for One-Dimensional Steady State Transistor Calculations, IEEE ED-11 (1964), p. 455-465

[5] van der Vorst, H.K.; Dekker, K.: Conjugate gradient methods and preconditioning, Journal of Computational and Applied Mathematics 24 (1988), North Holland

[6] Wu, K.-C.; Chin, G.R.; Dutton, R.W.: A STRIDE Towards Practical 3-D Device Simulation - Numerical and Visualisation Considerations, IEEE Trans. CAD 10 (1991), No. 9, 1132-1140