INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 21, 869–875 (1995)

THE PARALLEL PROCESSING TECHNIQUE APPLIED TO THE CELLULAR AUTOMATON METHOD IN FLUID DYNAMICS SIMULATIONS

YUKIHIKO INOUE

Mitsubishi Heavy Industries, Ltd., Advanced Technology Research Center, Yokohama, Japan

KAZUYUKI KATSURAGI

Mitsubishi Heavy Industries, Ltd., Electronics Division, Kobe, Japan

AND

OSAMU UKAI AND TAKESHI ADACHI

Mitsubishi Heavy Industries, Ltd., Advanced Technology Research Center, Yokohama, Japan

SUMMARY

This paper describes an efficient parallel algorithm of the cellular automaton (CA) method for microscopic fluid dynamics simulations. The CA method is parallelized with so-called multispin coding and with one-dimensional domain decomposition. The parallel CA method has a constant computational load balance and small data transfer between only nearby domains. We have applied the parallel CA method to a large-scale Poiseuille flow simulation and an immiscible two-phase flow simulation on a Fujitsu AP1000 with 64 processors.

KEY WORDS: parallel processing; cellular automaton; domain decomposition; two-phase flow

INTRODUCTION

In fluid dynamics a particle simulation method called the cellular automaton (CA) method has been studied since a fundamental model for incompressible Navier–Stokes fluid was introduced by Frisch, Hasslacher and Pomeau (the FHP model).[1,2]

The CA method has some advantages over traditional numerical methods such as the finite difference method. First, the microscopic approach of the CA method enables us to simulate more complicated flows (e.g. immiscible two-phase flow,[3] chemical reactive flow[4]). Second, there is no round-off error because the calculation of the CA method is represented by bit operations completely. Moreover, the CA method is suitable for parallel processing because of its discrete structure.[5,6] Parallel processing is necessary for the practical use of the CA method because it requires very large computer resources. It is important to develop an efficient parallel algorithm and to remove difficulties for parallelization in the applications.

We have developed the parallel CA method with so-called multispin coding[7] and with one-dimensional domain decomposition. In this paper the high efficiency of the parallel CA method is shown. As applications of the parallel CA method, a large-scale Poiseuille flow simulation and an

CCC 0271–2091/95/100869–07

Received 4 June 1994
Revised 7 November 1994

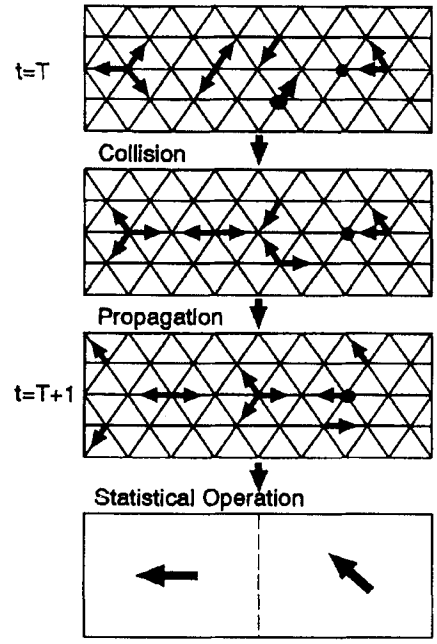

Figure 1. Particle motion in CA method

immiscible two-phase flow simulation are presented. These simulations have been implemented on a Fujitsu AP1000 with 64 processors (peak performance 960 MIPS).

MATHEMATICAL MODEL

The CA method is a completely discrete dynamic system. In the FHP model the motion of seven kinds of particles is defined on a regular triangular lattice[1] (see Figure 1). There are collision processes and propagation processes in the unit time step. Particles change or maintain their velocities while conserving mass and momentum at each node by collision, then move to a neighbouring node in the direction of their velocities by propagation. Macroscopic information of flow is obtained through a statistical interpretation of particle data.

This simple mathematical model can be represented by so-called multispin coding.[7] In multispin coding the existence of a particle at each node is represented by one bit called a cell. Cells are packed into the machine word and calculated simultaneously by bit operations on words (collision by Boolean operations, propagation by bit shifts and substitutions). Therefore the calculation is very efficient, so that a large number of particles can be dealt with.

PARALLELIZATION

The calculation of the CA method is very local; in the FHP model the collision process occurs independently at each node and propagation processes act on neighbouring nodes. Moreover, cell data can be constructed in arrays corresponding to the lattice. Thus the CA method has a fine grain structure in space and is therefore parallelized with the domain decomposition technique. This type of parallelization is called single-instruction stream/multiple-data stream (SIMD).

The computational load of multispin coding does not depend on the number of particles but on the number of nodes, because node states are updated by the fixed bit operations on each cell. Therefore the computational load balance can be maintained by decomposing the domain according to each processor's ability only once. It is not necessary to consider dynamical computational load balancing. In this parallel algorithm, data transfer between nearby domains is necessary because of the interactions between neighbouring nodes. In propagation processes, cell data at the boundary of decomposed domains are transferred.

In order to decrease the communication time for data transfer, we have considered domain decomposition patterns suitable for the CA method. There are two typical domain decomposition patterns: one-dimensional and two-dimensional (see Figure 2). Generally the parallel processing with domain decomposition technique adopts two-dimensional decomposition because the boundary data are inversely proportional to the number of decomposed domains. However, our parallel CA method adopts one-dimensional decomposition because of the following advantages.

1. *Simple network.* Corresponding to the triangular lattice of the CA method, the network for two-dimensional decomposition has six neighours. On the other hand, there are only two neighbours for one-dimensional decomposition. Therefore only a few data sets are transferred at each time step in one-dimensional decomposition. Moreover, the simplicity enables us to develop the software and hardware easily.
2. *Suitability for data structure.* Cell data can be decomposed in parallel with the machine word line one-dimensionally. Then all the boundary data can be transferred on words simultaneously.

We have compared two kinds of domain decomposition patterns in a 325,680-node calculation of the FHP model. Let T_n be the elapsed time with n processors. We define the speed-up factor $S_n = T_1/T_n$. The relation between the number of processors and S_n for two kinds of decomposition patterns is shown in Figure 3. We can see that one-dimensional domain decomposition is more effective than two-dimensional and its speed-up is approximately linear.

SIMULATIONS

The following simulations by the parallel CA method present difficulties for parallelization in terms of computational load balance and data transfer.

Large-scale Poiseuille flow

A Poiseuille flow with Reynolds number 2155 has been simulated using the FHP model. The Reynolds number of the CA flow is estimated from the macrodynamical equations derived from the microscopic conservation equations.[1]

In this simulation there is a difficulty for parallelization. The extra force calculation required to keep the mean velocity constant causes imbalance of the computational load. We therefore assigned the extra force calculation to only one processor for the inlet, and in order to maintain the computational load balance, we decreased the domain size of the inlet (see Figure 4). This type of parallelization is called multiple-instruction stream/multiple-data stream (MIMD).

We confirmed that there is no idling time for this implementation caused by the extra force calculation using the AP1000's performance monitor. Figure 5 shows the relation between time and the number of operating processors during 10 time steps. We can see that most of the processors were always in calculation time (black area) but in very short communication time (white area). This calculation (5,171,200 nodes, 1,500,000 time steps) took 59 h on the AP1000.

Figure 6 shows the velocity distributions after 0, 300,000, 600,000 and 1,500,000 steps. The simulation results are in good agreement with the theoretical curve.

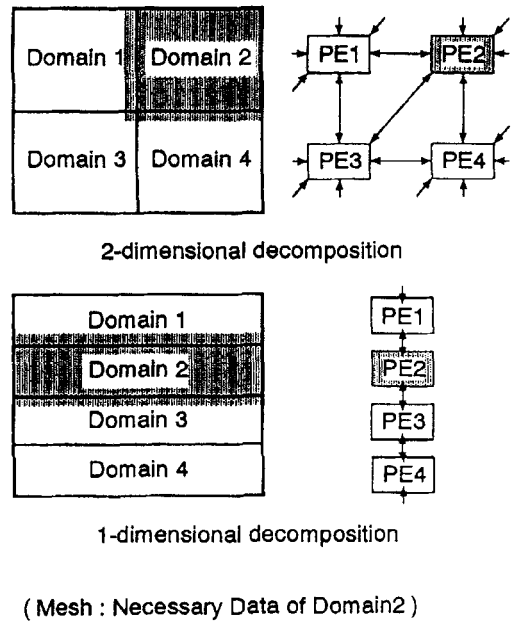

Figure 2. Domain decomposition patterns

Immiscible two-phase flow

The first immiscible model built upon the FHP model was introduced by Rothman and Keller (the RK model).[3] The RK model is represented by two kinds of coloured particles and based on a minimization principle. The collision rule makes the coloured particles separate into two phases at each node. However, the calculation is too complex to be represented by multispin coding and consequently the RK model takes a much longer calculation time than the FHP model.

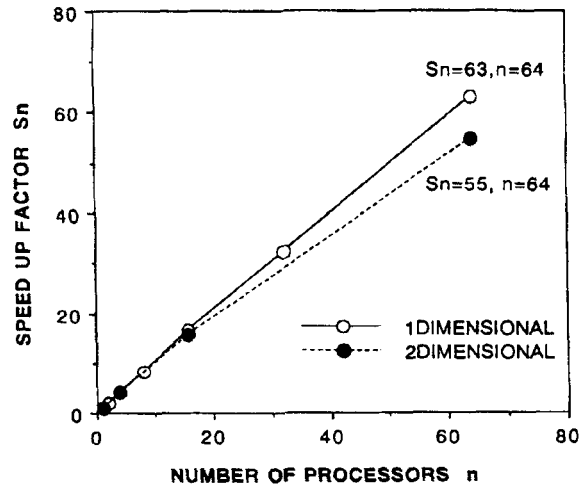

Figure 3. Speed-up of FHP model

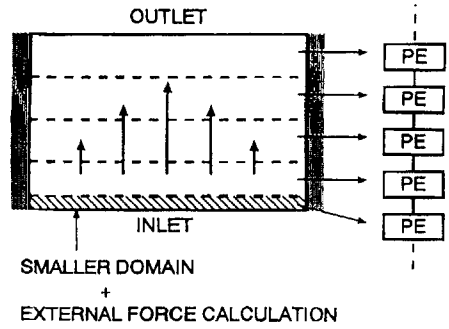

Figure 4. Parallelization of Poiseuille flow

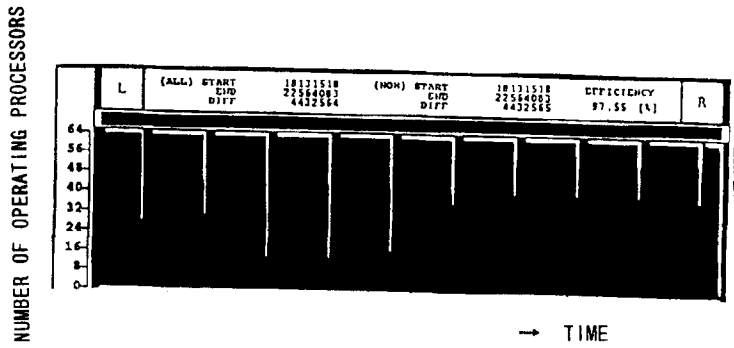

Figure 5. Performance monitoring

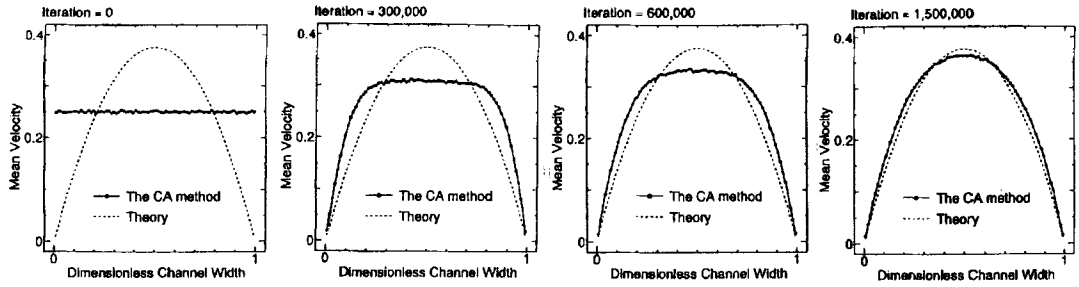

Figure 6. Poiseuille flow by CA method

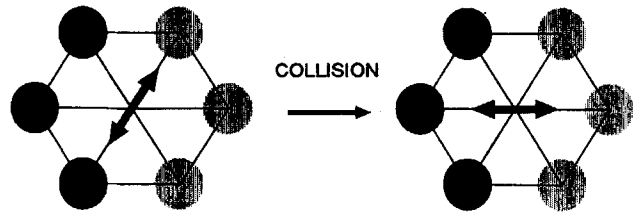

Figure 7. Collision in immiscible model

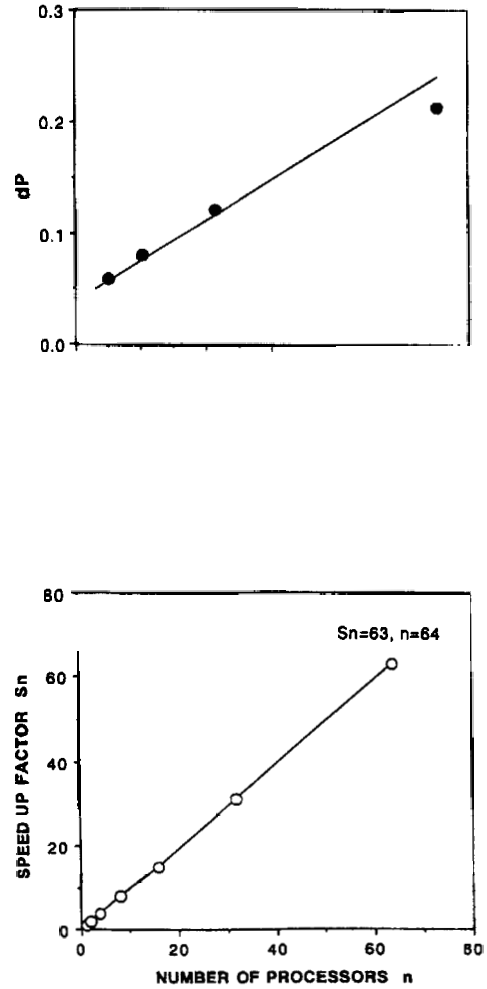

Figure 9. Speed-up of immiscible model

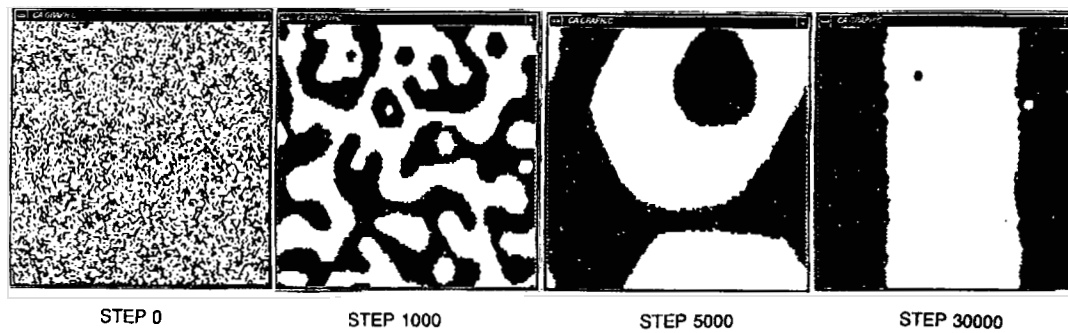

Figure 10. Phase separation by CA method

Therefore we have developed a simplified immiscible model with multispin coding as

1. denotation of the superior colour at each node
2. decision of the local colour potential from the colour of neighbouring nodes
3. change in particle direction according to the colour potential

where all data are represented by bit data and all directions are restricted to the lattice. Figure 7 shows an example of the collision rule of our model. Note that the dynamics of a single phase is the same as in the FHP model and that the colour potential energy changes into kinetic energy in our model.

In a phase separation such as water/oil interaction the surface tension plays an important role. Here the surface tension of our model has been tested, the same as the RK model.[3] A bubble of a fluid is set in another fluid initially. When the system attains equilibrium, the bubble radius R and the pressure difference dP at the interface are measured. Figure 8 shows that the surface tension of our model satisfies Laplace's law: dP is proportional to the inverse of R.

The parallel CA method of our simplified immiscible model has been applied to a phase separation simulation. There is another data transfer to the neighbouring node in the immiscible collision process. The data are concerned with the colour information at each node. However, it can be minimized by the previously discussed simplification of the model. Moreover, there is no idling time because of the constant computational load due to multispin coding. Therefore we achieve parallel efficiency as good as for the FHP model (see Figure 9). This simulation (16,384 nodes, 30,000 time steps) was calculated in 3 min by the AP1000.

Figure 10 shows the non-equilibrium behaviour of our model after 0, 1000, 5000 and 30,000 steps. We can observe that the random mixture of two fluids is eventually separated into two phases completely.

For practical phase separation simulations, more exact microscopic models are necessary. We consider our simple model to be a prototype of such models.

CONCLUSIONS

A parallel algorithm of the CA method has been developed. There is no idling time because of the constant computational load due to multispin coding and the communication time is decreased by one-dimensional domain decomposition.

Our parallel CA method has been applied to two examples: a large-scale Poiseuille flow by MIMD-type parallelization and an immiscible two-phase flow by simplifying the mathematical model. These simulations have been calculated efficiently on a Fujitsu AP1000.

The high efficiency of the parallel CA method shows its feasibility for massively parallel processing of larger-scale and more complex microscopic fluid dynamics simulations.

ACKNOWLEDGEMENTS

The authors would like to express their sincere gratitude to Professor Ryoichi Takahashi and Dr. Yousuke Matsukuma of the Tokyo Institute of Technology for their useful comments and suggestions on the CA method. They would also like to thank Fujitsu Limited for the implementation on an AP1000.

REFERENCES

1. U. Frisch, D. d'Humieres, B. Hasslacher, P. Lallemand, Y. Pomeau and J. Rivet, *Complex Syst.*, **1**, 649–707 (1987).
2. S. Wolfram, *J. Stat. Phys.*, **45**, 471–526 (1986).
3. D. Rothman and J. Keller, *J. Stat. Phys.*, **52**, 1119–1127 (1988).
4. A. Lawniczak, D. Dab, R. Kapral and J. P. Boon, *Physica D*, **47**, 132–158 (1991).
5. F. Hayot, M. Mandel and P. Sadayappan, *J. Comput. Phys.*, **80**, 277–287 (1989).
6. B. Boghosian, W. Taylor IV and D. H. Rothman, in *Proc. 3rd Int. Conf. on Supercomputing*, vol. 2, pp. 34–44, 1988, International Supercomputing Inst., Petersburg, Florida, USA.
7. J. Hardy, O. de Pazzis and Y. Pomeau, *Phys. Rev. A*, **13**, 1949 (1976).