INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids **29**: 251–264 (1999)

COMPARISON OF CELLULAR AUTOMATA AND FINITE VOLUME TECHNIQUES FOR SIMULATION OF INCOMPRESSIBLE FLOWS IN COMPLEX GEOMETRIES

J. BERNSDORF[a], F. DURST[a] AND M. SCHÄFER[b,*]

[a] *Institute of Fluid Mechanics, University of Erlangen-Nürnberg, Cauerstr. 4, D-91058 Erlangen, Germany*
[b] *Department of Numerical Methods in Mechanical Engineering, Technical University of Darmstadt, Petersenstr. 30, D-64287 Darmstadt, Germany*

SUMMARY

A cellular automata method for the prediction of incompressible fluid flows is presented and its practical relevance is investigated by comparing it with a standard finite volume solver. The cellular automata approach is based on an advanced lattice Boltzmann technique for a discrete microscopic description of the fluid flow. The finite volume solution procedure solves the macroscopic Navier–Stokes equations by means of an advanced multigrid pressure-correction technique on block-structured grids. Advantages and disadvantages of the cellular automata technique are discussed and quantitative comparative results related to accuracy, convergence properties and computing times are given for test cases with varying geometrical complexity. As a practical application of the method, results for the numerical simulation of the flow in a porous sedimentary layer are given. Copyright © 1999 John Wiley & Sons, Ltd.

KEY WORDS: incompressible flows; complex geometries; finite volume methods; lattice Boltzmann methods

1. INTRODUCTION

Cellular automata methods represent an alternative approach to more conventional finite volume or finite element techniques for the numerical prediction of fluid flows. While the standard techniques are based on the numerical solution of the macroscopic balance equations of mass and momentum, derived in the framework of continuum mechanics, the cellular automata approach is based on a microscopic description of the fluid flow by a discrete Boltzmann equation.

In recent years, considerable progress has been made with respect to a more general applicability of these kinds of methods for practically relevant flow problems [1]. It is often argued that there are advantages in using such an approach when dealing with flows of highly complex geometries, for instance as occurring in porous media. The major objective of this paper is to evaluate an advanced cellular automata technique quantitatively with respect to this issue. For this, the method is compared with a finite volume multigrid flow solver, which is known to be efficient from the numerical point of view (e.g. Durst and Schäfer [2]). In particular, the comparison covers aspects of numerical accuracy and computational efficiency, which are crucial for practical applications when considering flow geometries of increasing

* Correspondence to: Department of Numerical Methods in Mechanical Engineering, Technical University of Darmstadt, Petersenstr. 30, D-64287 Darmstadt, Germany.

CCC 0271–2091/99/030251–14$17.50

Received November 1996
Revised May 1997

complexity. Comparisons between lattice Boltzmann automata and finite difference methods for simple geometries already give an indication of the efficiency of the Boltzmann automata technique [3,4].

To illustrate the high geometrical flexibility of the cellular automata method, its application to the numerical simulation of a porous sedimentary layer is considered as a more practically relevant test case.

Concerning the flow problems considered, this paper is restricted to two-dimensional laminar steady flows of an incompressible Newtonian fluid at low Reynolds numbers. In the setting of continuum mechanics, such problems are governed by the well-known incompressible Navier–Stokes equations, given by

$$\frac{\partial(\rho \boldsymbol{v}_j)}{\partial x_j} = 0, \tag{1}$$

$$\frac{\partial}{\partial x_j}\left(\rho \boldsymbol{v}_j \boldsymbol{v}_i - \mu \frac{\partial \boldsymbol{v}_i}{\partial x_j}\right) = -\frac{\partial p}{\partial x_i}, \tag{2}$$

where $\boldsymbol{v} = (v_1, v_2)$ is the velocity vector with respect to the Cartesian co-ordinates (x_1, x_2), ρ is the fluid density, μ is the dynamic viscosity and p is the pressure.

It can be shown theoretically, that for the kind of flow problems considered, the assumptions are valid, which are necessary to ensure that the cellular automata technique employed yields, after some suitable scaling process, macroscopic values for velocity and pressure, also fulfilling Equations (1) and (2) [5–7]. The comparative results presented in Section 3 give a clear confirmation of this.

The features of the finite volume method employed for comparison sake are described in detail in [2], so only a brief summary is given here. The method uses block-structured grids with a collocated arrangement of variables and second-order discretization is used for all terms (central differences, linear interpolation). After solving the momentum equations for an intermediate velocity field, the continuity equation is used to obtain a pressure-correction equation following the spirit of the SIMPLE algorithm of Patankar and Spalding [8]. The equations for the Cartesian velocity components and the pressure-correction are solved successively, and outer iterations are performed to take into account the non-linearity and the coupling of variables.

Further, a multigrid method is employed that keeps the number of outer iterations nearly independent of the number of control volumes. The method is based on the full approximation scheme with V-cycles and it is implemented with a nested iteration technique (full multigrid method). Details of the multigrid scheme, which has been proven to work efficiently for a variety of applications [9–11], are given in [12].

2. CELLULAR AUTOMATA METHOD

The basic idea of cellular automata methods is the numerical simulation of simplified molecular dynamics of the fluid. This is done by evaluating a time- and space-discrete Boltzmann equation, the so-called lattice Boltzmann equation [13]. Macroscopic values like pressure and velocity can be obtained from the automata fluid density distributions, which, if some assumptions are fulfilled, have a behaviour similar to that governed by the Navier–Stokes equations [5–7].

Before describing the lattice Boltzmann automata (LBA) used for the comparison here, the so-called Frisch, Hasslacher and Pomeau (FHP) automata is introduced, allowing for an easier understanding of the basic principles of the underlying techniques.

2.1. FHP automata

The FHP automata was first proposed by Frisch *et al.* in 1987 [13]. The basic principle of this approach is that binary 'particles' with unit mass are propagated on a hexagonal Bravais lattice (see Figure 1) in discrete time steps with unit velocity. It should be noted that a triangular or quadrilateral lattice, which are commonly used in connection with finite volume or finite element techniques, would cause an anisotropic flow.

A particle position and a (discrete) velocity vector is indicated by its lattice co-ordinates and its position at the node. Every node splits into six cells, as indicated in Figure 2, showing as an example a lattice node with particles at cells 1, 2 and 5. An exclusion principle is imposed that does not allow the location of more than one particle per cell at the same time. The six possible velocity vectors $\vec{c}_j$ are:

$$\vec{c}_j = [\cos(2j\pi/6), \sin(2j\pi/6)], \quad j = 1, \ldots, 6. \tag{3}$$

A flow diagram of the overall algorithm is indicated in Figure 3.

The propagation step consists of moving all particles to the next node in the direction of its velocity vector, with the restriction that propagation is only possible between the six nearest neighbours along the lattice connection lines (see Figure 4).

Depending on the particular model employed (FHP I, II or III), more or less complicated collision rules may change the velocity vectors of those particles entering a collision state after the propagation step. A minimum set of collision rules to reproduce realistic flow are the head-on and the three particle collisions (see Figure 5) that are used in FHP I. It should be noted that particle mass and momentum are conserved during collisions.

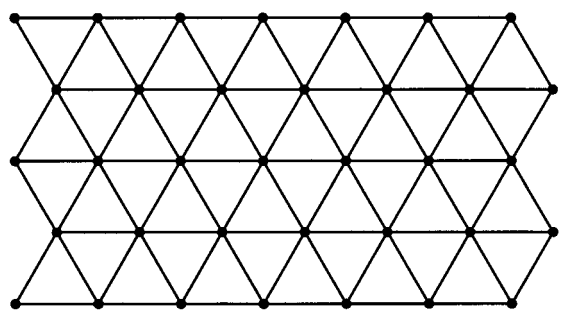

Figure 1. Hexagonal Bravais lattice.

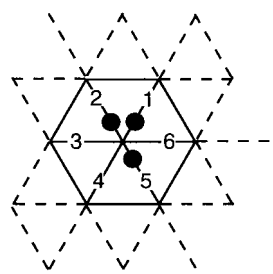

Figure 2. Lattice node with particles at cells 1, 2 and 5.

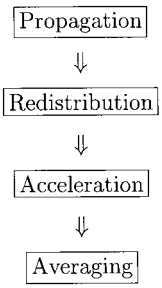

Figure 3. Flow diagram of the cellular automata algorithm.

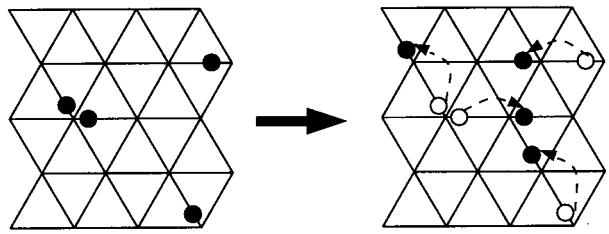

Figure 4. Propagation: particles moving to new positions.

To get some directed flow (and a pressure gradient), the velocity vectors of a certain amount of randomly chosen particles have to be changed in every iteration. A flow in the (positive) x-direction, for example, is achieved by moving particles from cell 3 to cell 6 at the corresponding nodes, as illustrated in Figure 6. The resulting local disturbance is propagated through the lattice, and after a number of iterations, an equilibrium between acceleration and viscous forces imposed by collisions and boundary effects (see below) evolves.

In order to obtain macroscopic values, e.g. pressure and velocity, from the particle distributions, appropriate time and space averaging procedures have to be carried out. Denoted by $n_i(\vec{r}_*, t_*) \in (0, 1)$, the binary particle density at cell $i \in (1, 2, \ldots, 6)$ of lattice node $\vec{r}_*$ at time t_*, for example, the mean velocity can be obtained by:

$$\langle u \rangle = \frac{1}{M} \sum_{t_*=1}^{M} \frac{\sum_i \vec{c}_i \sum_{\vec{r}_*} n_i(\vec{r}_*, t_*)}{\sum_i \sum_{\vec{r}_*} n_i(\vec{r}_*, t_*)}, \tag{4}$$

where M is the number of iterations taken into account for averaging.

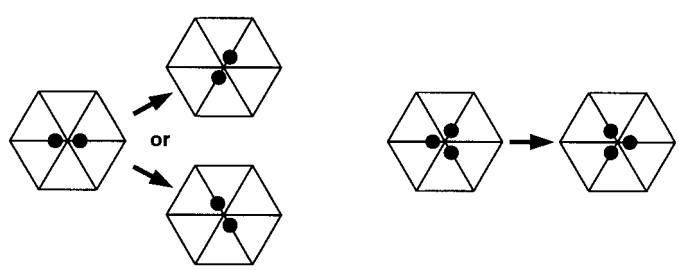

Figure 5. Head-on and three particle collisions.

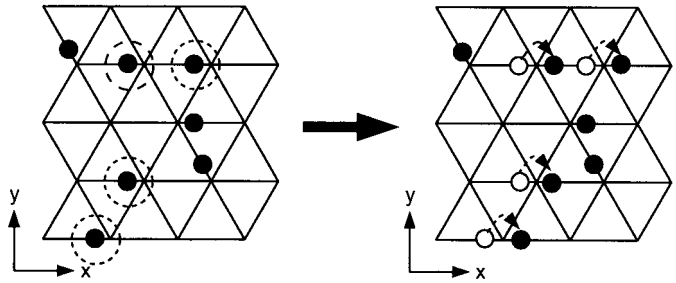

Figure 6. Acceleration: randomly chosen particles (circle) change velocity vector.

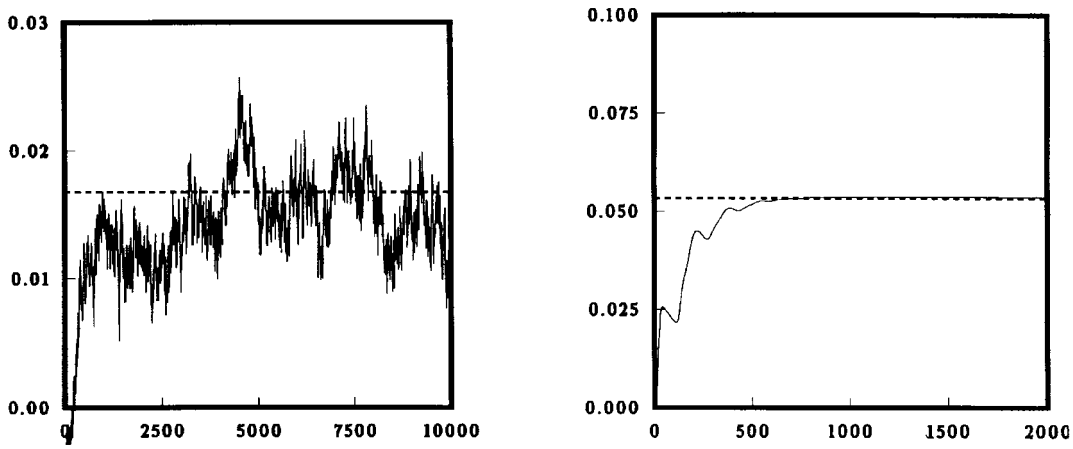

Figure 7. Mean flow rate (y-axis) vs. iteration number (x-axis) for FHP (left) and LBA (right). The dashed lines indicate the theoretical value.

2.2. Lattice Boltzmann automata (LBA)

Due to the discrete nature of the binary FHP automata, stochastic noise makes it necessary to average over long-time and large areas. This problem can be avoided by using particle densities as ensemble averages of the Boolean values. The particle collision rules are replaced by matrix multiplications that produce the new particle distributions after the collisions:

$$N_i(\vec{r}_* + \vec{c}_i, t_* + 1) = N_i(\vec{r}_*, t) + \Delta_i(N), \tag{5}$$

The densities N_i are the ensemble averages of the binary particle densities n_i, where the exact definition of the collision operator $\Delta_i(N)$ depends on the details of the model used (for the two

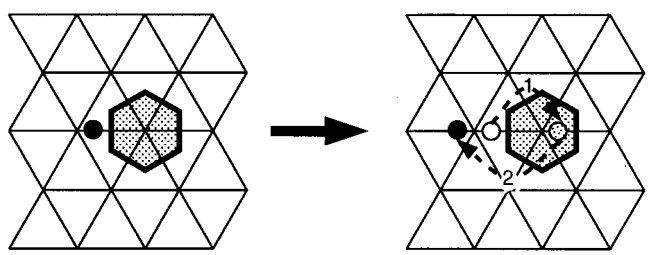

Figure 8. Treatment of an obstacle boundary: a particle that would be moved to an occupied node is bounced back.

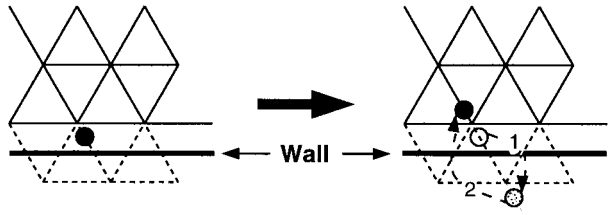

Figure 9. Treatment of a wall boundary: a particle that would leave the lattice is bounced back.

most common see [7,14]). For the examinations conducted, a collision operator was used as defined in the enhanced collision model [7]:

$$\Delta_i(N) = a_{ij}(N_j - N_j^{\mathrm{eq}}), \tag{6}$$

where N_j^{eq} denotes the equilibrium population and a_{ij} is the coefficient of the collison matrix determining the scattering rate between the directions i and j. The value of a_{ij} depends on the angle between the directions $\vec{c}_i$ and $\vec{c}_j$, and altogether the coefficients have to fulfil the conservation of mass and momentum:

$$\sum_{i=1}^{b} a_{ij} = 0, \qquad \sum_{i=1}^{b} \vec{c}_i a_{ij} = \vec{0}, \quad j = 1, \ldots, b. \tag{7}$$

The advantage of this approach is that the computations can be performed on smaller grids with less iterations, since after some time, the density distribution does not change, and the final values can be directly obtained without any time and space averaging processes. This effect is illustrated by a comparison of the FHP and the LBA for the simulation of a plane Poiseuille flow, which is shown in Figure 7. Indicated are the mean flow rate versus the iteration number for both cases. In most practical applications, this advantage compensates the less efficient and slower operations (from the computational point of view) with real numbers (particle densities) necessary for the LBA method compared with the more efficient logical operations on binary particles with an FHP approach.

2.3. Realization of boundary conditions

In both cases, FHP as well as LBA, obstacles can be handled easily by just marking lattice nodes located inside an obstacle (occupied nodes) and employing the additional rule that particles or particle densities, which would be moved to occupied nodes, are simply bounced back (see Figure 8).

This simple procedure allows for an easy implementation of arbitrary complex structures (e.g. the flow simulation through a porous sedimentary layer, as presented in Figure 18) or to change the obstacle structure during the computation, which is necessary for problems with

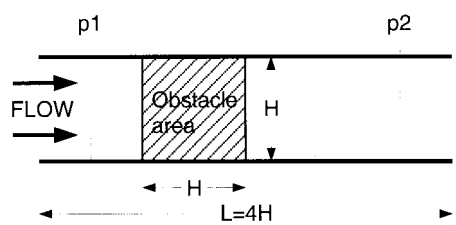

Figure 10. Channel dimensions and locations p1 and p2 for evaluating the pressure difference.

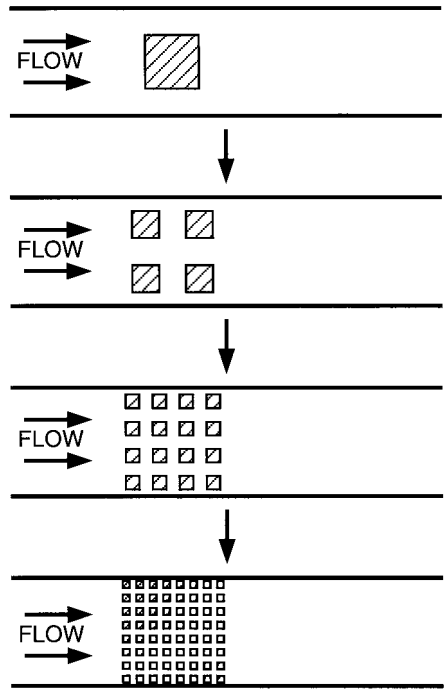

Figure 11. Obstacle structure with increasing complexity.

time varying flow geometry. Of course, due to the lattice structure, obstacle geometries can only be approximated by hexagons. In addition, one has to take care that the smallest structure elements are large enough to avoid the so-called finite size effect, which may lead to non-physical results [15,16] if the number of unoccupied cells between the obstacles is too small.

The same 'bounce back' principle as described above is applied to fixed walls at the lattice boundaries (for this and other possible boundary conditions see [17]). All particles tending to leave the lattice are reflected towards their incoming direction (see Figure 9), resulting in a zero mean velocity at the boundary (no-slip boundary condition).

It is well-known that the simple 'bounce back' boundary conditions employed are only first-order-accurate, and several types for second-order boundary conditions have been suggested [3,18–21]. The major drawback of all of these second-order boundary treatments is the large amount of additional computational work that has to be done in comparison with the first-order bounce back condition. On the other hand, it has been shown [18] that under appropriate conditions (not too high inner viscosity of the LBA on too small lattices), the bounce back condition leads to sufficiently good results. In the examination here, the lattice resolution was increased for every sample of obstacles several times, until no further change could be seen in the measured flow properties. This way, the error in the computations originating from boundary conditions was ensured to be negligible.

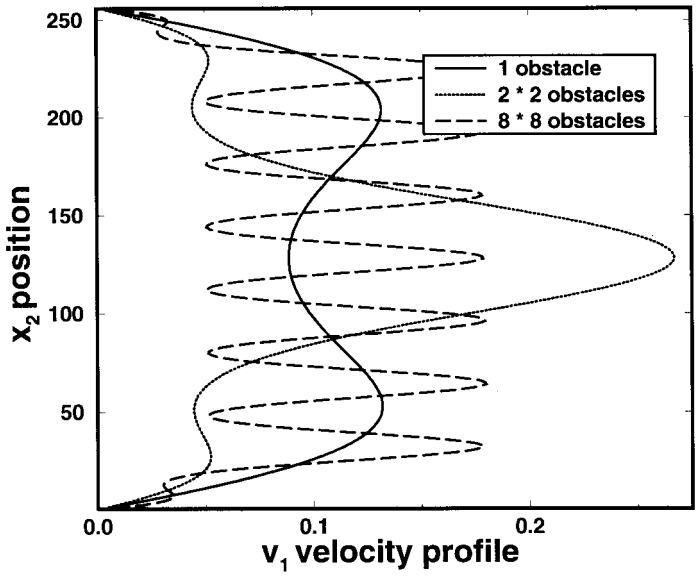

Figure 12. Profile of velocity magnitude in a cross-section closely behind the obstacle area for different obstacle numbers.

3. NUMERICAL RESULTS

3.1. Channel flow with obstacles

As a test case for the comparison of the LBA method with the finite volume approach, a flow in a channel of height H and length $L = 4H$, with different numbers of square obstacles regularly placed in the second quarter of the channel (see Figures 10 and 11) was considered. The sizes of the obstacles are chosen in a way that the space occupied by them is the same for all cases, and the distance between the wall and the obstacles nearest to the wall is always half the distance between the obstacles. Thus, when increasing the number of obstacles, this test case represents a flow problem of systematically increasing geometrical complexity. The

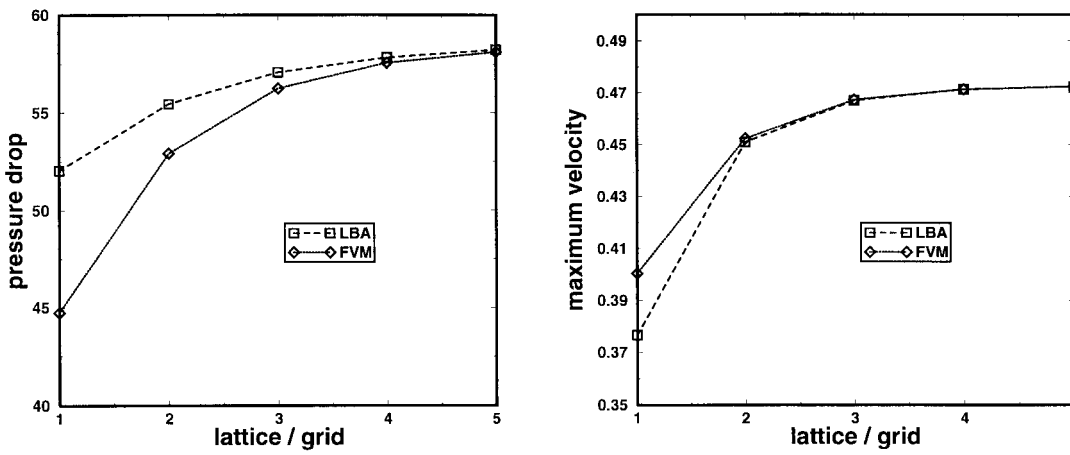

Figure 13. Pressure drop (left) and maximum velocity (right) vs. grid size using LBA and FVM.

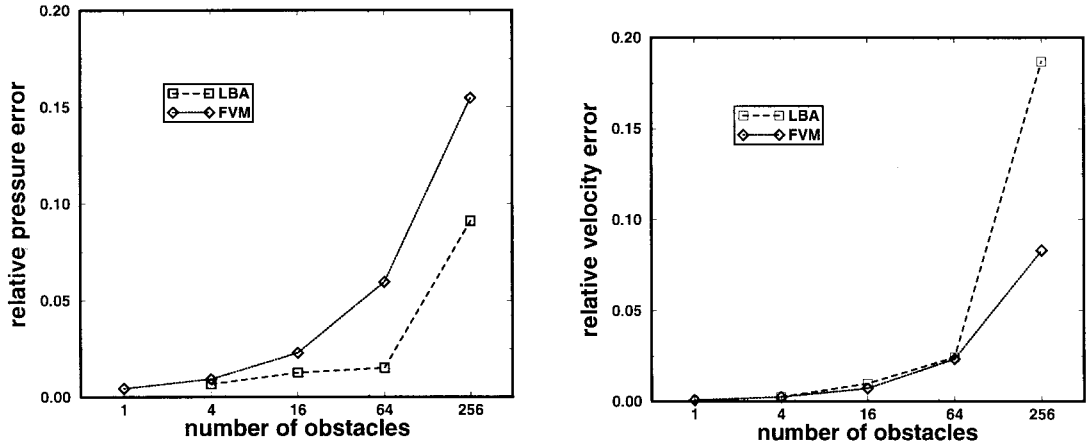

Figure 14. Error in pressure drop (left) and maximum velocity (right) vs. number of obstacles using LBA and FVM.

increasing complexity of the flow pattern is illustrated in Figure 12, showing the profiles of the velocity magnitude in a cross-section close behind the obstacle area for the cases with different obstacle numbers.

Concerning the boundary conditions at the inlet and outlet of the channel, for the finite volume code, a parabolic inflow profile corresponding to a Reynolds number of $Re = 0.1$ (based on the channel height) and a zero gradient outflow condition are chosen; whereas the cellular automata boundary conditions are periodic in the flow direction. A change in the density distribution of the first lattice row at the inlet (decrease in cells 2, 3 and 4; increase in cells 1, 5 and 6, see Figure 2) leads to a directed flow for the cellular automata. The flow profile already appeared to be parabolic some lattice rows away from this first 'acceleration' row. The region downstream of the obstacle area, for the considered Reynolds number, is long enough to avoid any influence of the outlet to the flow profile.

The convergence criterion for the iteration process in the finite volume method is that the absolute sum of the residuals for mass and momentum (weighted with the corresponding inlet values) is $< 10^{-4}$. For the LBA method, the program stops when the maximum deviations of the mean flow rate in the last quarter section of the channel differ $< 10^{-4}$ for the last 50 iterations.

As reference quantities for evaluating the accuracy of the computations, the pressure difference between the cross-sections at $x = L/8$ and $x = 7L/8$ of the channel (see Figure 10) and the maximum absolute value of the velocity were computed.

In Figure 13 the reference quantities are indicated for the 2×2 obstacle case for both methods, using lattice/grid sizes of increasing fineness ranging from 64×16 cells (grid 1) to 1024×256 cells (grid 5). For the FVM, an equidistant Cartesian grid is employed.

The results show good agreement in the high resulution limit for both pressure drop and maximum velocity. Actually, the pressure drop on the coarser grids is closer to the fine-grid value for the LBA; whereas the finite volume method gives closer values for the velocity. For the other obstacle configurations, similar results concerning the dependence of the accurracy of the reference quantities on the grid fineness were found.

In Figure 14 the error for the reference quantities for a fixed grid size (256×64 cells) is plotted against the number of obstacles for both methods, indicating the dependence of the accuracy of the methods on the complexity of the geometry. It can be observed that concerning the pressure, the LBA results are slightly better, whereas for the velocity error, the FV program is closer to the final values.

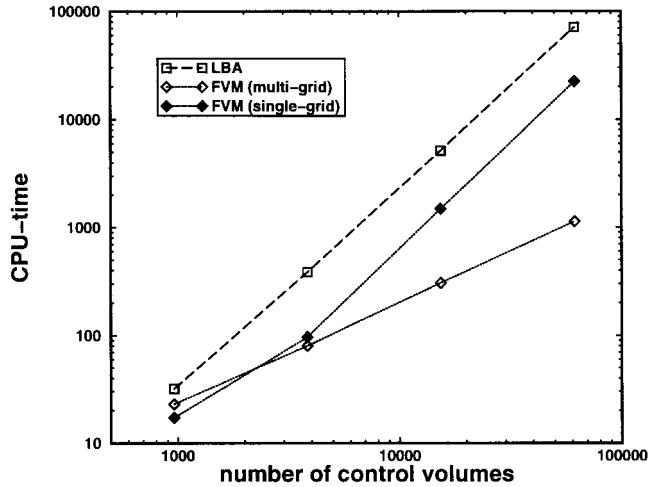

Figure 15. CPU times vs. number of control volumes for LBA and FVM (single-grid and multigrid).

Concerning a comparison of the computational efficiency of the LBA and the FVM, in particular, the following two aspects are of interest: the behaviour of the methods for increasing grid size and for increasing the number of obstacles (i.e. increasing geometrical complexity). In order to isolate the corresponding effects both aspects are studied separately.

In Figure 15 the CPU times for computations with 64×16 to 512×128 cells for the case with one obstacle are shown for both methods, where for the FVM timings are given for multi- and single-grid computations (all computations were carried out on a SUN10 workstation). The LBA and the single-grid FVM show the typical quadratic increase in CPU times with grid size, as expected from these kind of methods. The absolute timings are lower for the single-grid FVM computation. The multigrid FVM shows the usual linear increase in CPU time with grid size, resulting in increasing advantages compared with the other two methods when the grid is refined.

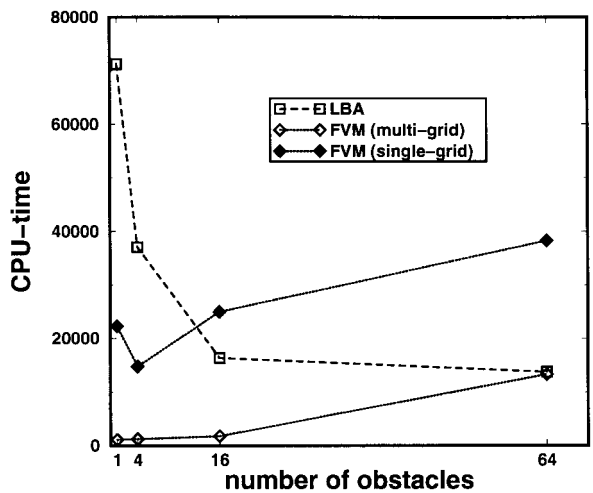

Figure 16. CPU times vs. number of obstacles for LBA and FVM (single-grid and multigrid).

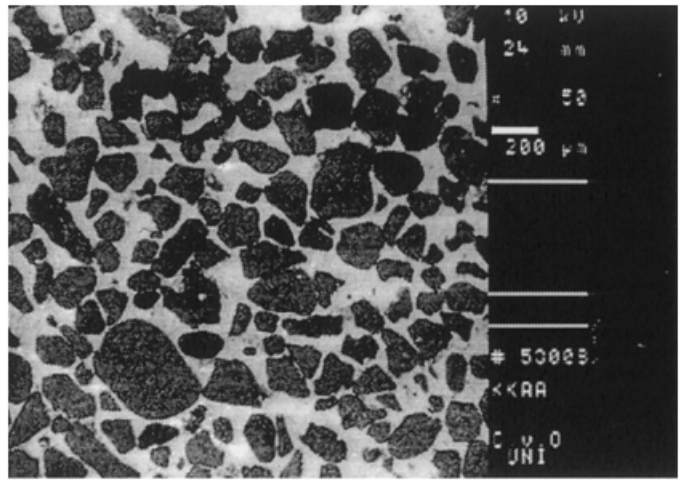

Figure 17. Digitised electron microscope picture of a sedimentary layer, from [23].

The CPU times for computations with 1 to 8×8 obstacles and a fixed lattice/grid with 512×128 cells are shown in Figure 16 for both methods (for the FVM, timings are given for multi- and single-grid computations). For the multigrid case with every increase in the number of obstacles, one grid level less could be used for the computation because the coarsest grid has to be still fine enough to fully resolve the obstacle structure. Comparing the CPU times, an increase for the FVM with increasing number of obstacles is evident. For the LBA, the CPU time decreases with increasing number of obstacles and finally becomes almost independent of the number of obstacles. As a consequence of this general behaviour, the LBA approach is already faster than the single-grid FVM for the 16 obstacle case. The results also indicate a break-even point between the multigrid FVM and the LBA, but it is shifted to larger numbers of obstacles owing to the higher numerical efficiency of the multigrid method.

3.2. Complex geometry flow

For practical applications, especially in connection with real porous structures, the LBA approach allows for a quick and easy numerical simulation of flow properties, and therefore, this method appears to be a good working tool for this kind of problems. To illustrate the capabilities of the method, the fluid flow through a digitised electron microscope picture of a sedimentary layer of the Northern sea shore, as shown in Figure 17, was simulated.

The computation was performed on one processor of a Cray-YMP. Ten thousand iterations were needed for a 500×500 lattice to reach the steady state. This corresponds to 1200 s of total CPU time, where $1.47 \cdot 10^6$ lattice site updates per second can be performed with a performance of 190 MFLOPS (from 330 MFLOPS theoretical peak performance). This indicates the possibility of a straightforward vectorisation of the lattice Boltzmann cellular automata computations, which constitutes another advantage of this method. (The same is true for the parallelisation of the method, but these aspects are not the topic this paper.)

Figure 18 shows the computed velocity vectors for the sedimentary layer, and an enlarged section can be seen in Figure 19. The pictures illustrate that, regardless of the complexity of the pores, the flow features expected are well-captured.

4. CONCLUSIONS

The quantitative behaviour of an advanced cellular automata technique based on a LBA was investigated in comparison with a FVM. The accuracy of the results with respect to velocity and pressure on comparable lattice/grid sizes appeared to be in the same order of magnitude for both methods and the results coincide for fine-grids.

Concerning complex geometries, the CPU times for the LBA first decreased with increasing complexity of the obstacle structure and became almost independent from it for highly complex structures. The FVM appeared to be more efficient for simple geometries, but the CPU times increased for a higher number of obstacles in the channel. There appears to be a break-even point, where, at a certain complexity of the geometry, the lattice Boltzmann method becomes more efficient than the finite volume approach.

Of course the major drawback of all direct numerical flow simulations, to which the lattice Boltzmann method belongs, is the huge amount of computer resources needed to handle increasing lattice sizes for increasing Reynolds numbers. However, in most of the flow problems in highly complex geometries, which are considered to be the main area of application for this method, typical Reynolds numbers are orders of magnitudes smaller than

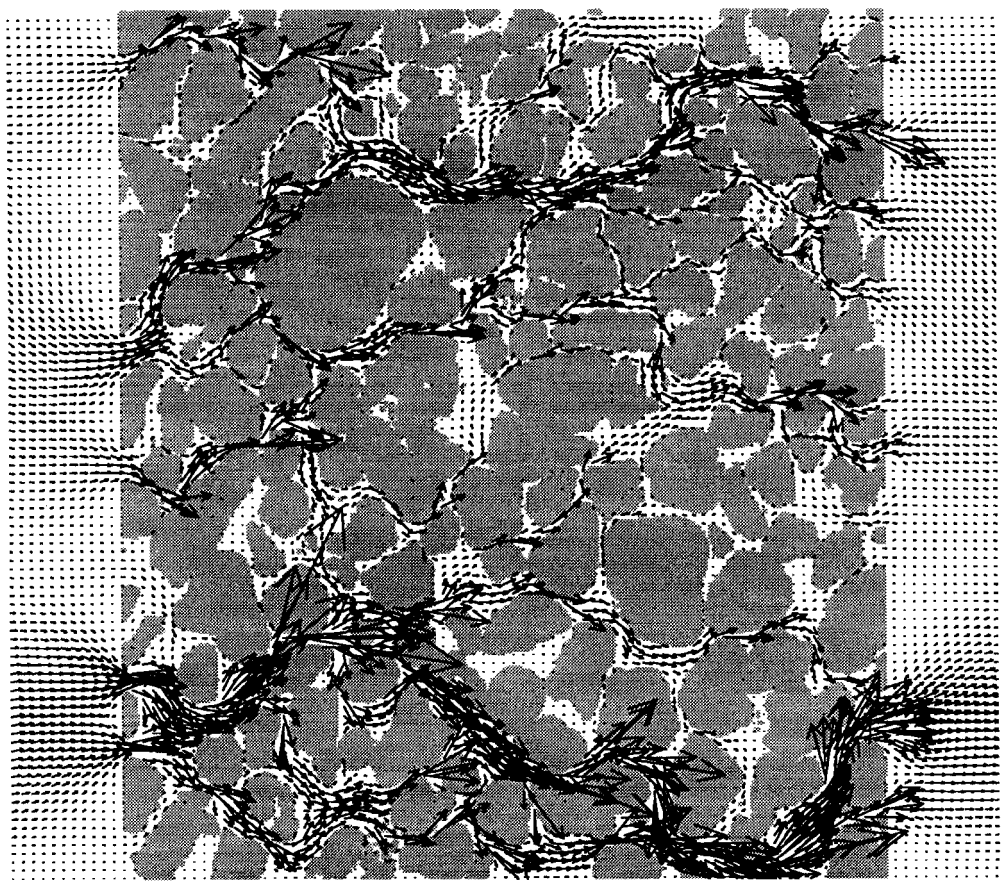

Figure 18. Numerically predicted velocity field for the flow in the sedimentary layer (every tenth velocity vector in both dimensions is plotted).

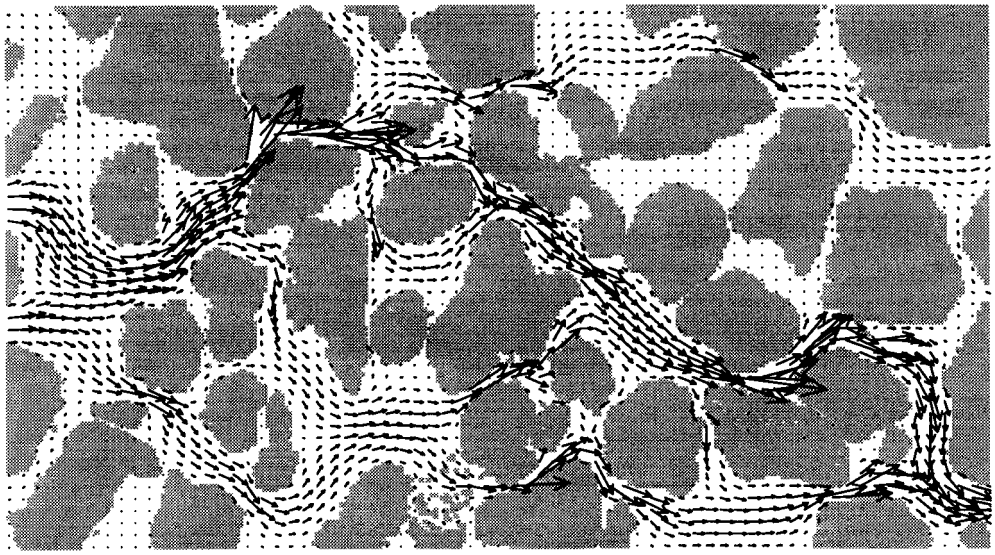

Figure 19. Numerically predicted velocity field for an enlarged section of the sedimentary layer (every third velocity vector in both dimensions is plotted).

for more classical engineering problems. Also, as it is common practice for FVMs, it is possible to introduce some statistical turbulence modelling in the LBA method [22].

In summary, the presented results strengthen the often stressed opinion, that cellular automata-based methods are quite competitive tools with respect to the application of CFD for problems involving complex geometries such as porous media.

ACKNOWLEDGMENTS

The authors would like to thank S. Schreiber for carrying out the finite volume computations. The financial support by the Deutsche Forschungsgemeinschaft in the special programme 'Flow Simulation with High-Performance Computers' and the Bavarian Consortium of High-Performance Scientific Computing (FORTWIHR) within the research area 'Simulation of Flows' is gratefully acknowledged.

REFERENCES

1. J.G.M. Eggels, 'Direct and large-eddy simulation of turbulent fluid flow using the lattice Boltzmann scheme', *Int. J. Heat Fluid Flow*, **17**, 307–323 (1996).
2. F. Durst and M. Schäfer, 'A parallel block-structured multigrid method for the prediction of incompressible flows', *Int. J. Numer. Method Fluids*, **22**, 549–565 (1996).
3. D.R. Noble, J.G. Georgiadis and R.O. Buckius, 'Comparison of accuracy and performance for the lattice Boltzmann and finite volume difference simulations of steady viscous flow', *Int. J. Numer. Methods Fluids*, **23**, 1–18 (1996).
4. N. Satofuka, H. Nishida and N. Okaza, 'Comparison of lattice Boltzmann and Navier–Stokes simulations of homogeneous isotropic turbulence', *XIXth Int. Cong. of Theoretical and Applied Mechanics*, Kyoto, Japan, August 25–31, 1996.
5. F.J. Higuera and J. Jiménez, 'Boltzmann approach to lattice gas simulations', *Europhys. Lett.*, **9**, 663–668 (1989).
6. R.D. Kingdon, P. Schofield and L. White, 'A lattice Boltzmann model for the simulation of fluid flow', *J. Phys. Math.*, **25**, 3559–3566 (1992).
7. S. Succi, R. Benzi and F. Higuera, 'The lattice Boltzmann equation. A new tool for computational fluid dynamics', *Physica D*, **47**, 219–230 (1991).

8. S.V. Patankar and D.B. Spalding, 'A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows', *Int. J. Heat Mass Transf.*, **15**, 1787–1806 (1972).
9. U. Bückle, F. Durst and M. Schäfer, 'Numerical investigation of melting processes affected by natural convection', *J. Mater. Proc. Manuf. Sci.*, **4**, 69–78 (1995).
10. F. Durst, L. Kadinskii, M. Peri and M. Schäfer, 'Numerical study of transport phenomena in MOCVD reactors using a finite volume multigrid solver', *J. Cryst. Growth*, **125**, 612–626 (1992).
11. M. Hortmann and M. Schäfer, 'Numerical prediction of laminar flow in plane, bifurcating channels', *Comput. Fluid Mech.*, **2**, 65–82 (1994).
12. M. Hortmann, M. Perić and G. Scheuerer, 'Finite volume multigrid prediction of laminar natural convection: benchmark solutions', *Int. J. Numer. Methods in Fluids*, **11**, 189–207 (1990).
13. U. Frisch, D. d'Humières, B. Hasslacher, P. Lallemand, Y. Pomeau and J.-P. Rivet, 'Lattice gas hydrodynamics in two and three dimensions', *Complex Syst.*, **1**, 649–707 (1987).
14. Y.H. Qian, 'Lattice BKG models for Navier–Stokes equation', *Europhys. Lett.*, **17**, 479–484 (1992).
15. G.A. Kohring, 'Effect of finite grain size on the simulation of fluid flow in porous media', *J. Phys. II*, **1**, 87–90 (1991).
16. G.A. Kohring, 'Limitations of a finite mean free path for simulating flows in porous media', Universität Köln, April 1991.
17. P. Lavallée, J.P. Boon and A. Noullez, 'Boundaries in lattice gas flows', *Physica D*, **47**, 233–240 (1991).
18. T. Inamuro *et al.*, 'A non-slip boundary condition for lattice Boltzmann simulations', *Phys. Fluids*, **7**, 2928–2930 (1995).
19. D.P. Ziegler, 'Boundary conditions for lattice Boltzmann simulations', *J. Stat. Phys.*, **71**, 1171–1177 (1993).
20. P.A. Skordos, 'Initial and boundary conditions for the lattice Boltzmann method', *Phys. Rev. E*, **48**, 4823–4842 (1993).
21. D.R. Noble, S. Chen, J.G. Georgiadis and R.O. Buckius, 'A consistent hydrodynamic boundary condition for the lattice Boltzmann method', *Phys. Fluids*, **7**, 203–209 (1995).
22. Y.H. Qian, S. Succi and S.A. Orszag, 'Recent advances in lattice Boltzmann computing', In: D. Stauffer (ed.) *Annual Reviews of Computational Physics III*, 1995, p. 209.
23. J. Kropp, Analyse der inhomogenen Elementverteilung in Wattsedimenten, *Diplomarbeit*, Universität Oldenburg, 1992.

