Hermann Eberl, Cristian Picioaneu, and Mark van Loosdrecht

Kluwer Laboratory for Biotechnology, Delft University of Technology, The Netherlands
{h.j.eberl, M.C.M.vanLoosdrecht}@stm.tudelft.nl

Abstract: Biofilms are accumulations of microorganisms on surfaces in aquatic systems developing in highly irregular morphologies. The smallest length scale in a structural biofilm model is orders of magnitude smaller than the biofilm itself. This allows only simulation of small parts of the system and makes computing resources the restricting factor for problem size. The aim of this paper is to present an overview of a state of the art computational model for the formation of biofilm morphology in a hydrodynamic environment and to establish it as a High Performance Computing application. Data parallelism in model and solution algorithms is discussed, and experiences of a first implementation in High Performance Fortran on a CRAY T3E are reported. The biofilm model is combined from different branches of mathematical modeling: a discrete cellular automaton, continuous kinetic equations and conservation laws for fluid flow and substrate concentrations. Due to the irregularity of biofilm structure it is of major importance to apply numerical methods which are capable of dealing with virtually any geometrical structure. Numerical discussion of algorithms focuses on the computational expensive parts in the general solution procedure: a Lattice-Boltzmann-Method for flow field calculation and a combined HOC/CDS finite difference scheme for substrate concentration.

Keywords: biofilm morphology, irregular geometry, convection-diffusion/diffusion-reaction, fluid dynamics, Lattice-Boltzmann method, finite differences, High Performance Fortran.

1.1 INTRODUCTION AND OUTLINE OF THIS PAPER

Biofilms are accumulations of microorganisms on surfaces and interfaces in aquatic systems. They form in all kind of situations where nutrients are available to bacteria. Their occurrence is ubiquitous and their biochemical and biophysical properties are widely used since a long time in many biotechnological
Biofilms can have many different irregular structures and occur rough (a), hairy (b), patchy (c), or as isolated colonies (d). One photography covers an area of approximately $0.5 \times 0.3 \text{ cm}^2$.

Figure 1.1  Biofilms can have many different irregular structures and occur rough (a), hairy (b), patchy (c), or as isolated colonies (d). One photography covers an area of approximately $0.5 \times 0.3 \text{ cm}^2$.

applications, from food production (e.g., vinegar) to waste water engineering. Apart from processes exploiting the properties of biofilms also harmful biofilms develop naturally in technical and medical systems and lead to biofouling and biocorrosion; examples are biofilms on artificial organs like heart valves, in drinking water tanks, as plaque on teeth, or on ship hulls.

First mathematical models of biofilms have been developed since the 1970s [4] for the purpose of biofilm reactor design and description of metabolic conversion processes. There, primarily holistic mass balances are of importance which lead to sets of ordinary differential equations or one dimensional partial differential equations, assuming the biofilm has homogenous structure or heterogeneity in one dimension only. These models can easily be implemented and run on single processor workstations and personal computers.

In reality, however, biofilm structures can be highly irregular. They occur in a broad variety of different forms and no typical biofilm structure per se exists. Biofilms can be porous, hairy, filamentous, they might have internal mi-
crochannels, occur as patchy clusters of microorganisms, but may occur rather smooth and homogeneous as well. Examples for biofilms with different geometrical properties are shown in Fig.1.1. Besides type of the microorganisms and physiological group, major factors influencing biofilm structure are environmental conditions in the aquatic system: substrate availability, fluid flow, and shear forces working on the biofilm [10][13][14]. In return, these factors are also influenced by biofilm structure. This leads to a closed cycle of causalities.

Interest in detailed spatially resolved modeling of biofilm processes and in particular in the development of biofilm structure only evolved recently. Such models must be able to deal with highly irregular morphologies and require full spatial resolution of the system. The most comprehensive biofilm model on the scale of biofilm growth up to now was formulated in [19] and [20]. It is a composition of a discrete cellular automaton to describe spatial spreading and division of biomass, and continuous conservative equations to describe internal transfer processes. The two dimensional implementation presented with the model description was the first High Performance Computing application motivated from biofilm research. The present paper follows this work in the third dimension, though not always the same computational methods are used.

The paper starts with a brief overview of the multidimensional biofilm model. Then numerical algorithms for its computationally expensive parts are discussed. This focuses on the data parallelism inherent in model and algorithm and on an implementation in High Performance Fortran. The final chapter will present a result of a numerical experiment. Computations are performed on a CRAY T3E of the Center for High Performance Applied Computing at Delft University of Technology (HPaC).

1.2 THE BIOFILM STRUCTURE MODEL

The mathematical biofilm system $\Omega$ consists of two compartments: the solid biofilm $\Omega_2$ containing all the biomass and the bulk liquid $\Omega_1$, separated by an interface $\Gamma$. To describe development of a biofilm structure, cellular automaton like models based on discrete stochastic functions have been suggested by several authors recently [11][19][26]. Originally, cellular automata have been developed since the 1940s as selfreproducing systems and adapted for artificial life in theoretical biology [9]. Despite their lack of qualitative biochemical explanation [17], they can also be applied to describe at least in a quantitative way selforganizing living systems of the real world. For this purpose, the computational domain $\Omega$ is divided into equidistant rectangular grid cells. The grid cell size is the smallest length scale in the model. A grid cell is either occupied by biomass (i.e. it is a biofilm grid cell in $\Omega_2$) or it is not (i.e. it is a bulk liquid grid cell in $\Omega_1$). Thus, computational biofilm geometries can be as heterogeneous as their real world counterparts (see Fig.1.2 for some examples). Biomass in a grid cell will divide according to local rules and occupy an empty neighbor grid cell with biomass if biomass concentration $M$ reaches a certain critical value $M_c$. An appropriate set of rules has been formulated in [19]:

\[ \text{If } M > M_c \text{ then biomass will divide.} \]
Figure 1.2 Some computational biofilm geometries: rough, patchy, and isolated colonies; (a), (b), (d) are model created, (c) is a mushroom type biofilm, suggested as an idealized biofilm concept in [7]

local rules for biomass division

[CA1] WHERE \((M > M_c)\)

[CA2] divide biomass into two parts, one remains at this position

[CA3] IF (empty neighbor cells exist)

choose one randomly and put the second part there

[CA4] ELSE

displace a full neighbor cell, the displaced cell again

searches for a free neighbor [CA3]

Biomass concentration \(M\) in point \(x = (x_1, x_2, x_3)\) is governed by balancing biomass growth and biomass decay according to a kinetic equation

\[
\frac{\partial M}{\partial t}(t, x) = r_M(M; C_1, \ldots, C_s)
\]

which reduces to one ordinary differential equation per grid cell. Substrate concentration \(C_i, i = 1, \ldots, S\) in (1.1) quantifies the \(i\)th dissolved nutrient in the system. The particular form of \(r_M\) depends on the microorganism species and their kinetics. In the solid film phase \(\Omega_2\) the \(C_i\) are transported by a diffusion mechanism and consumed in biochemical reactions, in the bulk liquid \(\Omega_1\) they are due to convection and diffusion. Connection across the interface, is given by diffusive transport normal to the biofilm surface according to Fick’s Law. Thus, the convection-diffusion/diffusion-reaction model reads

\[
\frac{\partial C_i}{\partial t} + \sum_{j=1}^{3} u_j \frac{\partial C_i}{\partial x_j} = D_{1i} \sum_{j=1}^{3} \frac{\partial^2 C_i}{\partial x_j^2}, \quad i = 1, \ldots, S \quad \text{in} \quad \Omega_1 \quad (1.2)
\]
Figure 1.3 Characteristic time scales of processes in the biofilm model, taken from [20]

\[
\frac{\partial C_i}{\partial t} = D_{ii} \sum_{j=1}^{3} \frac{\partial^2 C_j}{\partial x_j^2} + r_i(M; C_1, \ldots, C_S), \quad i = 1, \ldots, S \quad \text{in} \quad \Omega_2 (1.3)
\]

\[
-D_{ii} \frac{\partial C_i}{\partial n} = -D_{ii} A \frac{\partial C_i}{\partial n} + R_i(M; C_1, \ldots, C_S) \quad i = 1, \ldots, S \quad \text{at} \quad \Omega_2 (1.4)
\]

Again, reaction rates \( r_i \) depend on the species and their kinetics. Reaction term \( R_i \) comes from volume integration of \( r_i \). \( A \) is the interface section area.

The hydrodynamic flow field \( u = (u_1, u_2, u_3) \) in the bulk liquid is given by the incompressible Navier-Stokes equations

\[
\sum_{j=1}^{3} \frac{\partial u_j}{\partial x_j} = 0 \quad (1.5)
\]

\[
\frac{\partial u_i}{\partial t} + \sum_{j=1}^{3} u_j \frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho} \frac{\partial P}{\partial x_i} + F_i + \nu \sum_{j=1}^{3} \frac{\partial^2 u_i}{\partial x_j^2} \quad i = 1, 2, 3 \quad (1.6)
\]

with external forces \( F_i \) and kinematic viscosity \( \nu \). As usual, \( \rho \) and \( P \) are fluid density and pressure, respectively.

Another influence on biofilm structure which is counteracting growth is biofilm detachment due to shear stress. This, however, is neglected in this first step of biofilm modeling, partly because of lack of quantitative knowledge of material parameters and detachment mechanisms in biofilms.

In [20], it was stressed, that different processes in a biofilm system happen at different characteristic time scales (see Fig.1.3). Since the slowest processes are determining the geometrical structure of the biofilm (biomass growth and detachment), constant film geometry can be assumed for the faster transport and conversion processes. Giving non-changing (or: changing slower than growth) external boundary conditions for hydrodynamics and substrate concentration, the transient model can be decoupled into stepwise solution of steady state fluid flow and mass transfer equations. A general solution procedure is sketched in Fig.1.4.
1.3 NUMERICAL METHODS AND IMPLEMENTATION IN HPF

Nearest neighbor problems like the biofilm growth automaton and some discretization schemes for partial differential equations are considered to be ideal candidates for the data parallel programming paradigm: The same arithmetic operations are applied to every element of a large data set independently. If these operations have no side effects and computing resources are available, they can be executed in parallel. An efficient data parallel programming language for distributed memory systems is High Performance Fortran (HPF, [12] [21]). It is used for a first implementation of the biofilm model. Simulations are carried out on a CRAY T3E.

In HPF, programs are coded in a global address space. The programmer distributes data across processors using distribution directives and – unlike MPI or PVM – leaves memory allocation and the actual communication to the compiler. Array data can be distributed along particular dimensions in block or cyclic patterns which are to be specified at compile-time independent of the actual number of processors at run-time. Arithmetical computations are performed locally according to the owner computes rule, i.e. they are executed by the processor owning the variable on the left hand side of an assignment statement. Parallel Fortran 90/95 language elements allow writing readable and short codes. The HPF compiler adds communication primitives to transfer data between nodes. Optimization techniques are automatically applied to reduce communication costs [3]. HPF programs are loosely synchronous.
The computationally expensive parts in the overall biofilm structure model are flow field calculation and mass transfer for the dissolved components. In the sequel we focus on these tasks, restricted to one species biofilms. Implementation for multispecies films is straightforward and bears no additional conceptual difficulties. In [18] and [19], this was performed for biofilms in a hydrostatic environment without convective transport.

1.3.1 Data parallelism in the biofilm growth model

Biomass growth (1.1) is numerically the cheapest part in the algorithm. It is strictly grid point local in computation. Full data parallelism is inherent in the model and no communication is required at all. This leads naturally to perfect parallel speedup. The actual amount of computation depends on the species and its kinetics, described by $r_M$. Often analytical solutions are known [18] which require only a few arithmetic operations per time step or semi-analytical solutions. In other cases an Euler-step has been applied for numerical treatment [19]. Thus, biomass growth is also not critical in computation time.

The local rule biofilm growth automaton is a next neighbor problem and requires at most overlap-shift communication. The only arithmetical operations per grid point are recomputing biomass concentration (one multiplication) and a call of the random number generator in the case of biomass division.

1.3.2 A Lattice-Boltzmann method for the Navier-Stokes equations

Though discrete models for hydrodynamics and diffusive processes have also been developed in the context of cellular automata, due to their restrictions in efficiency continuous approaches are applied for the biofilm model: a Lattice-Boltzmann method (LBM) for fluid flow and a combined HOC/CDS Finite Difference scheme for mass transfer.

The Lattice-Boltzmann method is considered to provide easy handling of irregular geometries [25]. Historically, it has been derived as a floating point extension of the Lattice Gas Cellular Automaton [23]. In the meantime, also an interpretation as an approximation to a continuum Boltzmann equation has been worked out. Mathematically rigorous derivations of the method are to be found in [5] and its references for both approaches.

1.3.2.1 The Lattice-Boltzmann Method. Lattice-Boltzmann methods are kinetic methods for macroscopic flow variables motivated by the idea that the Navier-Stokes equation (1.6) can be recovered from a Boltzmann equation through an analytical Chapman-Enskog procedure in the nearly incompressible limit [5]. In order to compute primitive flow variables as 1st moments of the solution of the Boltzmann equation, a lattice with discrete speed links $e_{\alpha} \in \mathbb{R}^P, \alpha = 1, ..., N_0$ is introduced in the phase space. An incompressible version [6] of the discrete Lattice-Boltzmann method reads

\[ f_{\alpha}(t + \Delta t, x + \Delta x e_{\alpha}) = f_{\alpha}(t, x) - \omega \left( f_{\alpha}(t, x) - f_{\alpha}^{eq}(t, x) \right) \]  

(1.7)
Figure 1.5 D3Q15 Lattice-Boltzmann stencil, matrix of discrete speed links and model parameters

\[
(e)_{\alpha,j} = \begin{pmatrix}
1 & -1 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & 0 \\
0 & 0 & 1 & -1 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & 0 \\
0 & 0 & 0 & 1 & -1 & 1 & -1 & 1 & 1 & -1 & 1 & -1 & 1 & 0
\end{pmatrix}^T
\]

\[f_{\alpha}^{eq} = A_\alpha \left( p + B\epsilon_{\alpha}^T u + C(\epsilon_{\alpha}^T u)^2 - D u^T u \right) \tag{1.8}\]

\[p = \sum_{\alpha=1}^{N_q} f_{\alpha}, \quad u = \sum_{j=1}^{N_q} \epsilon_{\alpha,j} f_{\alpha}, \quad j = 1, 2, 3 \tag{1.9}\]

with \(\nu = \left( \frac{1}{3\omega} - \frac{1}{6} \right) \frac{\Delta x^2}{\Delta t}\) \tag{1.10}\]

Model parameters \(A_\alpha, B, C, D\) are obtained from the Chapman-Enskog procedure and depend on the selection of \(\epsilon_\alpha\). One 3D standard method is D3Q15 with \(N_q = 15\) discrete speed links [6] [15] [22]. Here, \(\epsilon_\alpha\) are the links pointing to the next and the third next neighbors in the grid, and one rest particle link. The stencil of the method is sketched in Fig.1.5. External forces \(F_i\) driving the flow are added to (1.7) [23]. The scheme is an \(O(Ma^2)\) accurate approximation of the Navier-Stokes equation (1.6), where \(Ma\) denotes the grid Mach number.

For steady state computations, the method described is sufficient. Step size \(\Delta t\) then does not describe evolution in time anymore but is an iteration constant. Thus, \(\omega\) becomes a relaxation parameter [22] controlling speed of convergence and counteracting accuracy of the method. In the framework of traditional Computational Fluid Dynamics it can compared to an artificial compressibility method. Transient LBM flow calculations additionally require the solution of a Poisson equation like in classical CFD [6].

1.3.2.2 Boundary conditions. An efficient implementation of boundary conditions at the irregular interface , between bulk liquid and biofilm is of major importance for flow field calculation. The appropriate hydrodynamic boundary condition for the macroscopic flow field is the no-slip condition \(u \equiv 0\) in . Implementing this in the Lattice-Boltzmann Method, however, bears problems, since (1.7) actually is a finite difference discretization of a first order
semilinear equation. The boundary requirements herefore differ from those of the elliptic steady state Navier-Stokes equations (1.6). In LBM, the number of restrictions to be prescribed depends on the type of the boundary node, which is made up from the eight adjacent grid cells. These are situated either in the solid or in the liquid region. Hence, there are \(2^8 = 256\) different types of nodes, including flow node (in \(\Omega_1\)) and solid biofilm node (in \(\Omega_2\)) and symmetries. For every link \(\epsilon_a\) emanating from the solid into the flow region one condition must be specified.

In a reliable approach, macroscopic conditions and mass conservation should be fulfilled, as a consequence, automatically at least in a very good approximation. Most widely used is the bounce back scheme and its modifications: links are returned at the solid interface. This approach was adapted from LBM’s Lattice Gas predecessor. It agrees with the boundary concept for the continuous Boltzmann equation known as inverse reflection [1]. If \(B\) is the index set of links emanating from the wall, the bounce back rule reads

\[
f_B := f_{\pi(\beta)} \quad \forall \beta \in B \quad \text{with} \quad \pi : \{1...N_0\} \rightarrow \{1...N_0\}, \epsilon_{\pi(a)} := -\epsilon_a \quad (1.11)
\]

Buried links, i.e. pair of links \(a, \pi(a) \in B\), do not contribute to momentum but to conservation of mass. They are computed by averaging their values from the iteration step before [15].

Due to restrictions in performance it is only possible to compute small sections of biofilms (\(\ll 1 \text{ cm}\)). Cyclic and/or symmetric conditions are applied at the boundaries of \(\Omega\) to simulate infinite biofilms. They can also be implemented as bit operations, e.g. as momentum conserving elastic reflections (see [1] for the continuum Boltzmann equation formulation) and shift operations.

### 1.3.2.3 Implementation of the Lattice Boltzmann method.

For implementation of the method, grid node types are numbered according to an appropriately ordered binary pattern of their adjacent grid cells, which are labeled 1 for solid and 0 for liquid. Thus, in the interior of the film grid points are of type \(T_G = 255\) (all adjacent cells are in the biofilm) and in the bulk liquid of type \(T_G = 0\) (no adjacent cell is in the biofilm). Interface points have \(1 \leq T_G \leq 254\). The Lattice Boltzmann iteration (1.7) is splitted into five steps according to

**Lattice-Boltzmann Algorithm**

1. **Collision + Equilibrium + Forcing.** compute right hand side of (1.7) and add external forces where \(T_G \neq 255\)
2. **Streaming.** shift \(f_a\) along \(\epsilon_a\)
3. **Boundary Conditions.** (1.11) where \(0 < T_G < 255\)
4. **Recovery.** compute macroscopic variables according to (1.9) where \(T_G \neq 255\)
5. **Stopping criterion.** evaluate an appropriate stopping criterion, return to [LBM1] or exit.

The method can be implemented efficiently in FORTRAN 90/95 array syntax. Explicit indexing of grid points is only necessary for buried links, where
explicit *a posteriori* access to the values of the preceding iteration is needed. Particular steps of the algorithm have different parallelism properties. All arithmetical operations of the method are performed at marks [LBM1] and [LBM4]. They are fully grid point local and no communication is required in both steps. Bouncing back at solid walls and elastic reflection at [LBM3] can be implemented as simple bit-operations, also grid point local. *Overlap shift* communication is necessary if there are buried links across the data distribution borders, which depends on biofilm geometry, data distribution pattern and actual number of processors. Streaming [LBM2] carries information between grid points and communication is necessary. It can be easily and efficient implemented with the HPF version of the Fortran 90 CSHIFT function. Evaluation of a stopping criterion [LBM5] typically requires array reduction and, hence, communication.

Since all arithmetical operations are grid point local, all the local data are stored on the same processor. This means data are distributed by geometrical decomposition of the computational domain. Blockwise distribution patterns are chosen in order to keep data sets compact and to reduce communication. Since solid grid points (with no arithmetical work) are situated in the bottom region of the computational domain whereas only flow grid points (with arithmetical work) are in the top region, data are not distributed in the vertical direction $x_3$ in order to get better load balance, and neither are $f_o$ distributed in the $a$-dimension. During execution no redistributing of data is necessary. The performance of two distribution templates, $(\text{block} , *, *)$ and $(\text{block} , \text{block} , *)$, is compared in Fig.1.6 for two test cases with the computational biofilms from Fig.1.2a (256x64x48 grid points) and Fig.1.2b (128x64x32 grid points) with up to 64 processors. It turns out that the $(\text{block} , \text{block} , *)$ template achieves better speedup with increasing number of processors. This result has also been found for all the other test cases qualitatively and, thus, can be considered to be typical for this problem.

**Figure 1.6** Effect of data mapping on computing time: compared is number of iterations per second for $(\text{block} , *, *)$ and $(\text{block} , \text{block} , *)$ distribution for biofilms from Fig.1.2a,b.
If a simulation is to be executed not at once but in a batch mode, a disadvantage of the LBM becomes evident. To continue work without losses after a restart, it is necessary to save the whole LBM function array $f_\alpha(x), \alpha = 1, \ldots, 15$. This not only requires enormous disk space but also time for writing data and reading them again. An alternative is storing only primitive variables $p, u_j, j = 1, 2, 3$ and continue with equilibrium functions $f^{eq}_\alpha$ according to (1.8) instead of the actual $f_\alpha$ after restart. This typically takes several iterations to recover the old state. However, after the flow field is almost converged this costs a lot of computing time. The problem does not occur, if the relaxation parameter is chosen $\omega = 1$. In this case the whole LBM can be reformulated in terms of the primitive variables without storing $f_\alpha$ at all. This, however, is at cost of accuracy of the method.

### 1.3.3 Finite Difference approximation of the Mass Transfer equation

Mass transfer equations (1.2) in $\Omega_1$ and (1.3) in $\Omega_2$ are solved together over $\Omega$. They form a convection-diffusion-reaction equation with nonconstant coefficients $D = D_{1/2}, r(C)$ and nonsmooth coefficients $u_j, j = 1, 2, 3$ at the interface, $(u_j \equiv 0 \text{ in } \Omega_1 \cup \Omega_2)$. Under assumption of Monod kinetics for the reaction term it is

$$\frac{\partial C}{\partial t} + \sum_{j=1}^{3} u_j \frac{\partial C}{\partial x_j} = D \sum_{j=1}^{3} \frac{\partial^2 C}{\partial x_j^2} + r(C), \quad r(C) = \begin{cases} 0 & \text{in } \Omega_1 \\ \frac{K_r C}{K_s + C} & \text{in } \Omega_2 \end{cases} \quad (1.12)$$

The physically relevant weak solution is determined by internal boundary condition (1.4). Dirichlet boundary conditions are implemented at inflow, open-vessel conditions at outflow, and Neumann conditions at the other boundaries of the computational domain.

In the biofilm $\Omega_2$ the standard central difference scheme (CDS) with a seven-point stencil is used for discretization of the Laplace operator. It is well known, that this method will become unstable if applied to a convection-diffusion operator. Hence, another scheme was developed for bulk liquid $\Omega_1$, a high order compact (HOC) method which converges to the differential equation with $O(\Delta x^4)$. It is derived by expressing the truncation error of a central difference method by finite differences again and substituting it in the scheme. It can be
represented by a compact 19-point stencil, accessing the 6 nearest neighbors and the 12 second nearest neighbors (Fig.1.7). Recently, the properties of a HOC scheme have been analyzed and applied for the simpler two dimensional linear convection-diffusion problems and the three dimensional Poisson equation in [24]. It was found, that it is stable also for convection dominated problems. Interface condition (1.4) is discretized piecewise in one dimension, normal to the interface. For this purpose finite differences are not applied to the grid points themselves but to the centers of two dimensional interfacial cells. Concentration values are determined as average values of the next regular points of the 19-point stencil. A detailed description of the combined HOC/CDS method will be given in [8].

For the ease of implementation, readability, and maintainability of code, the resulting matrix of the discrete semilinear algebraic system

\[ A^c = r(c) \]  

is lexicographically ordered and stored in one single diagonal format with 18 subdiagonals. This is at cost of storing and computing zeros in the matrix for biofilm grid points and in the right hand side vector for bulk liquid grid points. (1.13) is solved with a Newton method. For the Jacobian, only diagonal elements of \( A \) corresponding to biofilm and interface points must be updated, due to the local definition of the reaction terms. The linear subsystem is solved with a Jacobi preconditioned BiCGSTAB method [2].

Per BiCGSTAB iteration step 4 inner products are to be performed, 6 SAXPYs, and 2 matrix-vector multiplications [2] plus evaluation of a stopping criterion which requires additional communication between processors. The Newton step requires one matrix-vector product and the stopping criterion plus vector assignments without communication. In the implementation, vectors have been *block* distributed, matrix \( A \) and hence its Jacobian are aligned to it like \((\text{block}, *)\). The speedup achieved by parallelization was almost linear and approximately the same for all tested scenarios (see Fig.1.8 for typical examples). Absolute computing time depends not only on the problem size but also on biofilm geometry and the hydrodynamic flow field, because these factors influence matrix properties.

### 1.4 An Illustration

Some simulations are carried out to demonstrate model results with a few pictures. A first engineering application of the programs will be discussed in [16] in more detail, some others will follow. The mechanism driving the flow strongly depends on the type of the biofilm reactor. The flow regime in the examples presented here is flow between two parallel plates. The one at the top moves with constant velocity. The biofilm is grown upon the bottom plate. This system can be considered an idealized Roto Torque biofilm reactor, which is described in [10]. Boundary conditions in main flow direction are cyclic. At the boundaries parallel to the main flow direction, symmetric conditions are applied. The ratio between kinematic diffusivity and mass diffusivity in the
Figure 1.8  Computing time for mass transfer for different number of processors. (a) the rough biofilm geometry from Fig.1.2d \( (Re = 0.13) \), and (b), (c) the mushroom type biofilm from Fig.1.2c \( (Re = 0.71, \ Re = 19.5) \). Given is also the size of the problem.

Figure 1.9  (a) Flow streamlines through the patchy biofilm from Fig.1.2b, vertically projected \( (Re = 1.25, Sc = 625) \). (b) Iso-concentration lines on the biofilm surface in the downstream part. Concentration decreases with the brightness of the grey scale to the bottom and in flow direction.

bulk liquid is described by the Schmidt number which is kept at \( Sc = 625 \). In Fig.1.9, a simulation of the computational biofilm geometry from Fig.1.2b is presented. The irregular biofilm structure is on top of a homogenous basefilm of four grid cell layers, which needs not to be encountered in flow field calculation.
The system size is 128x64x36 grid points for mass transfer and 128x64x32 grid points for fluid flow. Due to consumption in the biofilm, concentration decreases from biofilm surface to the bottom and in flow direction. In the liquid phase, nutrient concentration decreases also from bulk liquid to biofilm surface. This is partly because of the convective transport in the bulk and partly because of the reactions in the biofilm which influence the bulk liquid via interface condition (1.4). A more detailed qualitative and quantitative discussion of modeled biofilm processes will be presented elsewhere.

1.5 CONCLUSION

A composite mathematical model for geometrically heterogenous biofilms has been described. It consists of discrete parts like the biofilm growth cellular automaton and differential equations for mass transfer, mass conversion and fluid flow. The particular submodels and the applied solution algorithms have different degrees of inherent natural parallelism. A High Performance Fortran model realization was presented for the computationally expensive parts. It was found, that very efficient parallel speedup can be achieved for the selected algorithms. Incorporation in the described growth model is the next step. Future biofilm research must stress on including additional and internal biofilm processes which are not yet fully understood and therefore not yet mathematically well described. This will increase computing requirements and make structural biofilm modeling a high performance computing application for the long term. This paper should therefore be considered only as at the beginning of spatially resolved biofilm modeling.

Acknowledgments

This study was was supported by the European Community under the TMR Network From Biofilms to Bioreactors (BioToBio) (Contract No. ERBFMRX-CT97-0114).

References