ISSN 1995-4239, Numerical Analysis and Applications, 2018, Vol. 11, No. 1, pp. 4–15. © Pleiades Publishing, Ltd., 2018.
Original Russian Text © O.L. Bandman, 2018, published in Sibirskii Zhurnal Vychislitel'noi Matematiki, 2018, Vol. 21, No. 1, pp. 5–21.

A Discrete Stochastic Model of Water Permeation through a Porous Substance: Parallel Implementation Peculiarities

O. L. Bandman*

Institute of Computational Mathematics and Mathematical Geophysics, Siberian Branch, Russian Academy of Sciences, pr. Akad. Lavrent'eva 6, Novosibirsk, 630090 Russia

Received August 10, 2017; in final form, August 30, 2017

Abstract—In this paper, some peculiarities of parallel implementation of a discrete stochastic model simulating water permeation through a porous substance (soil) with complex morphology are studied. The model simulates fluid flow along pore curves and filling wets and cavities. The discrete stochastic model of the process, which was proposed earlier, is a stochastic cellular automaton (SCA) whose functioning is represented by a set of elementary local operators acting in the cellular space and imitating displacements (by diffusion, convection, adsorption) and transformations (by reactions, phase transitions) of abstract or real particles. The microlevel representation of the process requires a cellular space of huge size. Hence, the computations have to be implemented on supercomputers. The main problem is that obtaining an acceptable parallelization efficiency is possible only by introducing some determinism into the computation algorithm, that is, by decreasing the model stochasticity. Although stochastic models have been intensively investigated, parallel implementation methods for them have been poorly studied. This gap is partially filled by the results of computational experiments presented in this paper. These allow assessing the advantages and shortcomings of various methods of implementation on a multicore cluster of the discrete stochastic model of water permeation through a porous medium.

DOI: 10.1134/S1995423918010020

Keywords: *discrete simulation, stochastic cellular automata, algorithm stochasticity, transition rules, parallel computing, block-synchronous mode of functioning, porous material, discrete permeation model.*

1. INTRODUCTION

Mathematical models of spatially distributed natural processes based on partial differential equations imply that the space-time functions describing the process being simulated are continuous and differentiable. There is a growing interest in science and in the industry in the development of new materials and technologies. For this it is necessary to simulate phenomena that are mostly nonlinear and discontinuous, for instance, chemical reactions, formation of crystals, electric discharges in gases, growth and division of cells in living organisms, etc. These are processes of the "reaction-diffusion" type (RD-processes) [1–3]. Such processes may be dissipative, contain phase transitions, and take place in spaces with complex discontinuous geometry. To simulate such processes, attempts have been made to study the capabilities of event-stochastic methods of simulation. This approach has been used for several decades, with methods of different names in different scientific fields. The first method was that of simulating the reaction of chemical oxidation of carbon monoxide (the ZGB-reaction [4]). This method initiated the development of a direction of research called by chemists the "kinetic Monte Carlo method" (KMC-method) [5–8]. A similar method, called the "method of probabilistic cellular automata," is used in material science to simulate the destruction of materials [9–11]. The term "asynchronous cellular automaton" [12, 13] or "stochastic cellular automaton" [14] is most often used in the simulation of biological self-organization phenomena [12] and the formation of spatial and surface structures. These

*E-mail: bandman@ssd.sscc.ru

names exactly correspond to the standard classification of mathematical models of space dynamics. They also comply with a modern method of representing models in the form of rules (rule based models), which has been implemented in a programming language called "Wolfram language" [15] and used in the popular programming system *Mathematica* [16].

In discrete stochastic methods, the mathematical description of RD-processes is usually made using a set of elementary local operators (transition rules) acting on abstract or real particles capable of moving (by diffusion, convection) in a discrete space and undergoing various transformations in discrete time (by reactions, phase transitions). The local operators have the form of probabilistic substitutions reflecting the local changes in the state of the simulation space under certain conditions. The simulation space is an ordered set of cells. A cell is characterized by its space coordinates and state. The cell states may be integer numbers or symbols reflecting the presence of particles of a certain type at a given time. The actions of substitutions on the cells are stochastic: at any discrete moment of time, a randomly chosen substitution (time stochasticity) is applied to a randomly chosen cell (space stochasticity or asynchronism) [13, 14].

Although stochastic models have been widely used, very few methods of their parallel implementation on supercomputers have been developed. However, a simulation of real processes is reasonable at such sizes of cellular spaces only if it is performed on modern computer systems with parallel architectures. The parallel implementation of SCA-models is not trivial at all. The problem is that their space stochasticity calls for very fast transfer of every change in a boundary cell to a neighboring processor. Therefore, the domain decomposition method cannot be efficiently used when implementing the models on supercomputer clusters. First attempts to construct acceptable parallel implementation algorithms [17, 18] led to cumbersome and impractical implementations. A method of transition from an asynchronous mode to a block-synchronous one [19] turned out to be more efficient and convenient for the simulation of complex stochastic processes [14]. It has been successfully used in some investigations [20, 21]. However, this method is heuristic and needs additional justification, which can be obtained in experimental studies of the following features of a stochastic model [13]:

1. Acceptable efficiency of parallel implementation can be achieved only by introducing synchronization, that is, by decreasing the stochasticity of the model.

2. Introduction of synchronization into the calculation process must be coordinated in all processors participating in the parallel implementation.

3. Interrelation between the calculation time and interprocessor exchanges makes deep paralleling inefficient because of simplicity of local CA operators.

These features impose some constraints on the choice of parameters of the parallel algorithm for implementation of the model on supercomputers. In this paper, an attempt is made to qualitatively estimate the above constraints and verify the estimates obtained by computational experiments. As a representative example we take a stochastic CA model of a convection-reaction process of permeation of a liquid through a porous material with a complex intricate structure of the pores.

This paper consists of four sections. Section 2 provides a formal presentation of a stochastic CA model of an RD-process. Section 3 is devoted to an SCA model of permeation of a liquid into the soil and problems of its parallel implementation. The results are analyzed in the conclusions.

2. A STOCHASTIC DISCRETE MODEL OF A REACTION-DIFFUSION PROCESS

2.1. Formal Presentation of a Stochastic Cellular Automaton

The mathematical presentation of the discrete stochastic model of space dynamics (in what follows, the model) is based on the formalism of a *parallel substitution algorithm* [22], which was adapted in [23] to a description of CAs simulating natural processes. According to [22], the model is specified by the following three concepts:

$$\aleph = \langle X, A, \Theta(X) \rangle,$$

where

$X = \{\boldsymbol{x}_k \mid k = 1, \ldots, N\}$ is a finite set of cell *names*;

A, *alphabet of cell states*, is a finite set of symbols;

$\Theta(X) : A \times X \to A \times X$ is a *global operator of transition*.

The time is assumed to be uniformly discrete: $t = 0, 1, \ldots, t, \ldots, T$. At each time t the model is characterized by a *cellular array* $\Omega(t) \in A \times X$, $\Omega(t) = \{(u_k, \boldsymbol{x}_k) : u_k \in A, \boldsymbol{x}_k \in X\}$, which is an ordered (by names) set of pairs of the form $(u_k, \boldsymbol{x}_k)$, called *cells*, and the projection of the cellular array $\Omega_A(t) = (u_1, u_2, \ldots, u_{|X|})$ onto the alphabet of states is called a *global state* of the CA. On the set X, we define a *neighborhood of a cell* with name $\boldsymbol{x}$:

$$T(\boldsymbol{x}) = \{\boldsymbol{x}, \boldsymbol{x} + \boldsymbol{a}_1, \ldots, \boldsymbol{x} + \boldsymbol{a}_{q-1}\}, \tag{1}$$

where $\boldsymbol{a}_j$ is a vector of displacement of $\boldsymbol{x}$ in the coordinate j and q is the size of the neighborhood. The subset of cells with names from $T(\boldsymbol{x})$ form a *local configuration*

$$S(\boldsymbol{x}) = \big\{(u_0, \boldsymbol{x}), (u_1, \boldsymbol{x} + \boldsymbol{a}_1), \ldots, (u_{q-1}, \boldsymbol{x} + \boldsymbol{a}_{q-1})\big\} \tag{2}$$

with the *defining neighborhood* $T(\boldsymbol{x})$.

The global operator $\Theta(X)$ is a composition of a certain number of *local operators* $\theta_j(\boldsymbol{x})$, $j = 1, \ldots, n$, from a finite set $\Theta = \big\{\theta_1(\boldsymbol{x}), \ldots, \theta_n(\boldsymbol{x})\big\}$. The local operator $\theta_j(\boldsymbol{x})$ has the form of a substitution:

$$\theta_j(\boldsymbol{x}) : S_j(\boldsymbol{x}) \to S'_j(\boldsymbol{x}). \tag{3}$$

Applying $\theta_j(\boldsymbol{x})$ to a specific cell with name $\boldsymbol{x}$ is reduced to replacing the states u_i in the cells from $S_j(\boldsymbol{x})$ (2) with the new values u'_i in the cells from $S'_j(\boldsymbol{x})$ (3), and

$$u'_i = f_k(u_0, \ldots, u_{q-1}), \quad q = |S_j(\boldsymbol{x})|, \;\; k = 0, \ldots, |S'_j(\boldsymbol{x})|, \;\; |S'_j(\boldsymbol{x})| \le |S_j(\boldsymbol{x})|, \tag{4}$$

where $f_k(u_0, \ldots, u_{q-1})$ is a *transition function*.

Assume that the application of the substitution $\theta_j(\boldsymbol{x})$ to one cell $\boldsymbol{x} \in X$ takes place in one time step denoted by τ, and the execution of $\Theta(X)$ consists of applying all $\theta_j \in \Theta$ to all cells $\boldsymbol{x} \in X$, which constitutes a global transition $\Omega(t+1) = \Theta(\Omega(t))$ called an *iteration*. Thus, each iteration contains $|X|\,|\Theta| = Nn$ steps, that is, applications of the substitutions $\theta_j(\boldsymbol{x})$, $j = 1, \ldots, |\Theta|$, $\boldsymbol{x} \in |X|$, and the result of the simulation is an iterative process

$$\Omega^* = \Omega(0), \ldots, \Omega(t), \ldots, \Omega(T_{\text{fin}}), \tag{5}$$

called *evolution*.

The calculation of the global operator $\Theta(X)$ must satisfy a *correctness condition*. According to this condition, there must be no contradictions when passing from one global state to another. In other words, a cell must not simultaneously be subject to two attempts to change its state. Formally this condition means that

$$\forall\, \boldsymbol{x}, \boldsymbol{y} \in X \;\&\; \forall\, k, m \in \{1, \ldots, n\} : \; T'_k(\boldsymbol{x}) \cap T'_m(\boldsymbol{y}) = \emptyset, \;\; n = |\Theta(\boldsymbol{x})|. \tag{6}$$

It is possible that $m = k$.

Evolution essentially depends on the method (stochastic or deterministic) of choosing cells and substitutions in the calculation of $\Theta(X)$ at each iteration. A quantitative characteristic that greatly affects the behavior of the model is its *stochasticity* (λ), which is defined as the fraction of elementary probabilistic operations in the calculation of the global operator [26]:

$$\lambda = E_{\text{rand}}/E, \tag{7}$$

where E and E_{rand} are the total number of elementary operations and the number of elementary probabilistic operations in the calculation of $\Theta(X)$, respectively. An operation is considered to be probabilistic if its application is defined as a random number.

The method of choosing cell names $\boldsymbol{x} \in X$ determines *spatial stochasticity*, whereas that of choosing substitutions determines *operational stochasticity*. Both of them determine *model stochasticity*.

2.2. Operation Regimes

There are three main types of spatial stochasticity of SCAs. In [27] they are determined by *SCA operation regimes* and denoted by indices from a set $\rho = \{\sigma, \alpha, \beta\}$.

Synchronous regime, $\rho = \sigma$, implies the following algorithm of applying $\theta(\boldsymbol{x})$ to all $\boldsymbol{x} \in X$ at each iteration:

1. $\theta(\boldsymbol{x})$ (3) is applied to all $\boldsymbol{x} \in X$ (chosen in any order), and the new cell states are written into an auxiliary array $\Omega'(t)$;

2. $\Omega'(t)$ is initial for the next iteration, that is, $\Omega(t+1) = \Omega'(t)$.

In synchronous regime we have classical CAs [28, 29]. A peculiarity is that $|T'(\boldsymbol{x})| = 1$, that is, the application of a substitution changes the state of only one cell and, hence, the correctness condition (6) is satisfied automatically. This limits the capabilities of synthesis of synchronous CA-models, but simplifies their parallel implementations, allowing interprocessor exchanges once per iteration. Since all spatial operations are deterministic, $E^{\sigma}_{\text{rand}} = 0$.

Asynchronous regime, $\rho = \alpha$, consists of the following order of calculations of the global operator:

1. Choose a cell $\boldsymbol{x} \in X$ with probability $p = |X|^{-1}$;

2. Apply $\theta(\boldsymbol{x})$ to the chosen cell; immediately replace the states of the cell from $S'(\boldsymbol{x})$ by the corresponding new ones obtained from (4);

3. The iteration is terminated in $|X|$ cycles of steps 1 and 2 for each $\theta \in \Theta$.

In asynchronous regime, the condition (6) does not impose any constraints on the number of cells being updated simultaneously, which increases the capabilities of local operators. However, it complicates calculations on processors operating in parallel. For asynchronous regimes, an iteration usually means an execution in $|X|$ steps, as in the kinetic Monte Carlo method [7, 8]. In fact, there is no difference in the calculation process between the iterations. In asynchronous regime $\forall\ \theta \in \Theta$: $E^{\alpha}_{\text{rand}} = N$.

Ordered asynchronous regime, $\rho = \omega$, is a degenerate case of asynchronous regime when cells $\boldsymbol{x} \in X$ are chosen in a given (deterministic) order. Each time the states of the cells from $T'(\boldsymbol{x})$ are updated immediately. All operations are deterministic, $E^{\omega}_{\text{rand}} = 0$.

Block-synchronous regime, $\rho = \beta$, is intermediate between synchronous and asynchronous regimes [19]. It allows substitutions changing more than one cell simultaneously. However, for the condition (6) to be satisfied, simultaneous (synchronous) updates of states are performed on subsets $\Pi_l \in X$ in which the cells are rather far from each other. If a three-dimensional ($D = 3$) cellular array has a Cartesian structure, these subsets are defined as follows:

1. Choose a compact block of cells called a β-block, $B_\beta(i,j,k) = \big\{(i+a_i, j+a_j, k+a_k) : a_i, a_j, a_k = -R_\beta, 1, \ldots, 0, \ldots, R_\beta\big\}$, where $R_\beta \ll I$ is the *radius of the block*. The size of the block must satisfy the condition

$$B_\beta(i,j,k) \supseteq T''(i,j,k), \quad T''(i,j,k) = \bigcup_{l=1}^{n} T'_l(i,j,k),\ n = |\Theta|. \tag{8}$$

Here $T''(i,j,k)$ is the generalized defining neighborhood of the SCA, and T'_l is the defining neighborhood of the local configuration $S'_l(i,j,k)$ (2) in the substitution $\theta_l(i,j,k) \in \Theta$, $q' = |S'_l(i,j,k)|$. The linear size and the number of cells in the β-block are

$$b = 2R_\beta + 1, \qquad m = b^D,$$

respectively, where D is the dimension of the simulation space.

2. Correct the linear sizes I, J, and K of the cellular array X so that $I_b = I/b$, $J_b = J/b$, $K_b = K/b$ are integer numbers. Then select a subset $X_0 \subset X$:

$$X_0 = \big\{(i,j,k) \in X : \\ i = 0, b, 2b, \ldots, I-1;\ j = 0, b, 2b, \ldots, J-1;\ k = 0, b, 2b, \ldots, K-1\big\}. \tag{9}$$

3. Construct a partition $\Pi = \{\Pi_1, \ldots, \Pi_m\}$ of the set X generated by the β-block such that

$$\Pi_l \bigcap \Pi_h = \emptyset, \quad \bigcup_{l=1}^{m} \Pi_l = X \quad \forall\, h, l \in \{1, \ldots, m\}. \tag{10}$$

Each subset $\Pi_l \in \Pi$ contains cells obtained by shifting cell coordinates $(i.j,k) \in X_0$ by corresponding cell coordinates $(a_i, a_j, a_k)_l \in B_\beta(R_\beta, R_\beta, R_\beta)$ (8):

$$\Pi_l = \big\{(i + a_i, j + a_j, k + a_k) : \\ \forall\,(i,j,k) \in X_0,\ \forall\,(a_i, a_j, a_k)_l \in B(R_\beta, R_\beta, R_\beta)\big\}, \quad l = 0, \ldots, m. \tag{11}$$

4. An iteration consists of m *stages* chosen in random order. At each lth stage all $\theta \in \Theta$ are applied to all $(i,j,k) \in \Pi_l$ in synchronism. To provide correctness of (6) in parallel implementation, this random order of choosing stages must be the same for all processors. The number of probabilistic operations for each $\theta \in \Theta$: $E^{\beta}_{\text{rand}} = m$.

Operational stochasticity determines the order of choosing substitutions from $\big\{\theta_1, \ldots, \theta_n\big\}$ in the execution of the global operator. The most popular are two types of operational stochasticity (although there are also intermediate ones [21]), which are denoted in [13] by indices from $\mu = \{\delta, \gamma\}$.

Deterministic choice, $\mu = \delta$, is when to each cell $\boldsymbol{x} \in X$ all substitutions $\theta_i \in \Theta$ are applied in a given order, $E^{\delta}_{\text{rand}} = n$.

Random choice, $\mu = \gamma$, is when to each cell $\boldsymbol{x} \in X$ one randomly chosen substitution $\theta_i \in \Theta$ is applied with a probability

$$p_i = \frac{k_i}{\sum\limits_{i=1}^{n} k_i}, \tag{12}$$

Stochastic characteristics of global operators $\Theta_{\rho,\mu}$

Characteristics	ρ,μ							
	α,γ	α,δ	β,γ	β,δ	ω,γ	ω,δ	σ,γ	σ,δ
E_{rand}	Nn	N	mn	m	n	0	n	0
λ	1	$1/n$	m/N	m/Nn	$1/N$	0	$1/N$	0

where k_i, $i = 1,\dots,n$, are the rates of execution of events corresponding to substitutions $\theta_1,\dots,\theta_n$, $E^{\gamma}_{\text{rand}} = n$.

In what follows, for stochastic models we use an extended concept of model operation regime defined by a pair of characteristics from $\rho \times \mu$. An SCA-model with global operator $\Theta_{\rho,\mu}$ is denoted by $\aleph_{\rho,\mu}$. The stochastic characteristics for all cases are presented in the table. In complex SCA-models where methods of composition of global operators [25] are used, each of them can operate in its own regime.

In [14], a CA with a probabilistic choice of both spatial and operational regimes, that is, an (α,γ) regime, is called a stochastic one. However, due to the great diversity of properties of the processes being simulated with an efficient parallel implementation, the concept of stochastic model must be extended to incorporate all regimes presented in the table, including a degenerate synchronous (σ,δ)-regime.

3. SIMULATION OF A PERMEATION PROCESS

3.1. Stochastic Permeation Model

A CA model of permeation of a liquid into the soil was proposed in [30], providing a detailed description of its simulation abilities. We present a simplified version to call attention to some problems of parallel implementation.

A sample of soil through which the liquid permeates is specified by a digitized tomogram represented by a set of 1480 files. Each of these files imitates a horizontal layer occupying 700×700 bytes of a three-dimensional Boolean array $X = \{(i,j,k) : i,j = 0,\dots,699;\ k = 0,\dots,1479\}$ (Fig. 1), which corresponds to a soil sample of $2 \times 2 \times 15$ mm^3 (Fig. 2). If the binary value of a byte $u(i,j,k) = 0$, the cell (i,j,k) imitates the empty space of a pore. If $u(i,j,k) = 1$, the cell (i,j,k) imitates hard rock.

$\aleph = \langle A, X, \Theta(X)\rangle$ is an SCA simulating the liquid permeation with a symbol alphabet of states $A = \{s,w,a\}$, where s corresponds to hard rock, w, water, and a, pore space air. $X = \{(i,j,k) : i = 0,\dots,I-1;\ j = 0,\dots,J-1,\ k = 0,\dots,K-1\}$ is a discrete space, where $I = J = 700$, $K = 1000$ (the array height is decreased, since preliminary calculations have shown that the liquid does not permeate through the lower layer).

$$\Theta(X) = \Theta_{\text{C}}(\Theta_{\text{D}}(X)) \tag{13}$$

is a global operator, where $\Theta_{\text{D}}(X)$ simulates diffusion and $\Theta_{\text{C}}(X)$ simulates convection.

The diffusion operator $\Theta_{\text{D}}(X)$ imitates liquid spreading (that is, free surface leveling) in caverns (pores) and on the upper surface by using a substitution [30]:

$$\theta_{\text{D}}(i,j,k) : \big\{\big(w,(i,j,k)\big),\big(a,\phi_l(i,j,k-1)\big)\big\} \xrightarrow{p_{\text{D}}} \big\{\big(a,(i,j,k)\big),\phi_l\big(w,(i,j,k-1)\big)\big\}, \tag{14}$$

$$l = 1,\dots,c\,.$$

The substitution $\theta_{\text{D}}(i,j,k)$ performs exchanges of states between the cell (i,j,k) and the lth neighbor of the cell $l(i,j,k-1)$, $l = 1,\dots,c$, with name $\phi_l(i,j,k-1)$ for which $u_l \neq 0$ (not hard rock) and $c \leq 4$. The probability $p_{\text{D}} = 1/c$, $\theta_{\text{D}}(i,j,k)$ is successively applied to all kth planes from $k = 0$ to $k = K-1$ in an ordered asynchronous regime. In every kth plane $\theta_{\text{D}}(i,j,k)$ is applied to all (i,j,k) in synchronism (Fig. 3).

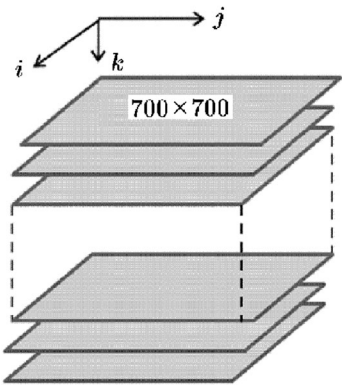

Fig. 1. File representation structure of soil sample.

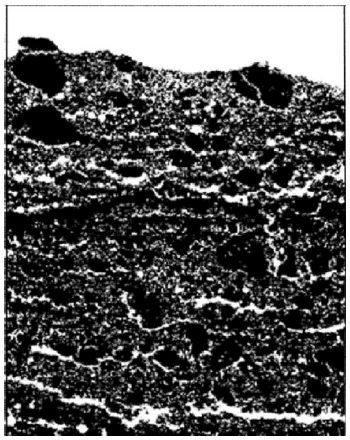

Fig. 2. Soil sample cross section at $j = 400$.

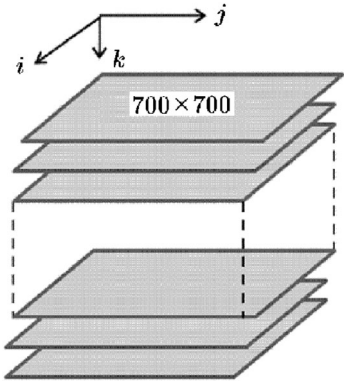

Fig. 3. Scheme of algorithm of permeation.

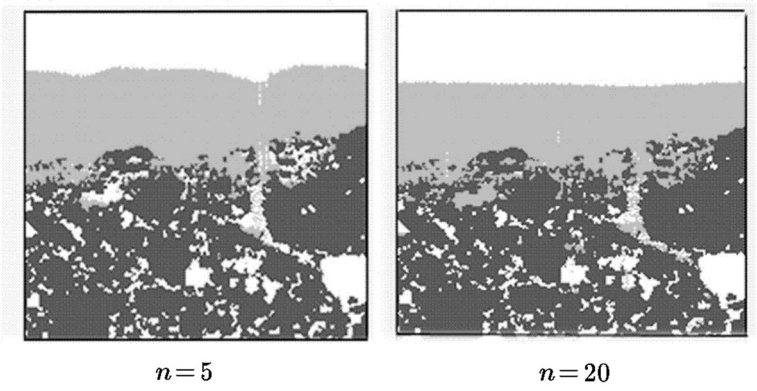

Fig. 4. Specifying the number of diffusion cycles per each iteration to provide a smooth liquid surface.

Since diffusion is more complex and slower than convection, to coordinate their rates the global diffusion operator $\Theta_{\mathrm{D}}(X)$ is specified to perform n cycles of applying $\theta_{\mathrm{D}}(i,j,k)$ to all $(i,j,k) \in X$ per iteration, that is,

$$\Theta_{\mathrm{D}}(X) = \big(\theta_{\mathrm{D}}(X)\big)^n. \tag{15}$$

The value of n depends on the properties and specific conditions of the substance. It can be chosen

in a test simulation visually on the monitor or by checking the simulation program for which value of n the free surface of the liquid becomes smooth (Fig. 4). To increase the efficiency of the parallel implementation, the diffusion operator is executed in a block-synchronous regime (β, γ).

The convection operator $\Theta_C(X)$ imitates the motion of abstract particles in the direction of gravity. At each step, the substitution $\theta_C(i, j, k)$ is applied in an asynchronous ordered regime to all layers of the array, from $k = K - 1$ to $k = 0$, in synchronism to all cells (i, j, k) in each kth layer:

$$\theta_C(i,j,k) : \big\{\big(a,(i,j,k)\big),\big(w,(i,j,k-1)\big)\big\} \xrightarrow{p_C} \big\{\big(w,(i,j,k)\big),\big(a,(i,j,k-1)\big)\big\}. \tag{16}$$

The probability p_C is set to be equal or close to unity, since this is the fastest action.

A major peculiarity of the three-dimensional CA permeation model is that its convection and diffusion components are divided in space so that the convection component is one-dimensional (along the vertical k axis) and the diffusion component is two-dimensional (in the horizontal planes $k = \text{const}$).

3.2. Peculiarities of Parallel Implementation of the Stochastic Permeation Model

To implement the SCA permeation model on a supercomputer, we use the domain decomposition method, which has been widely used in computational mathematics. It consists in dividing a cellular array into P parts, $\Omega = \Omega_0 \cup \ldots \cup \Omega_{P-1}$, called *domains*, and locating them on P processors operating in parallel. The division into domains is made along the vertical k axis in two ways: a) into plates (one-dimensional decomposition) and b) into columns (two-dimensional decomposition) (Fig. 5).

At such decomposition, the boundary data are exchanged only when using the diffusion operator, that is, between the faces that are parallel to the k axis (Fig. 5).

To provide the interprocessor exchange, the domains are supplemented by peripheral "shadow" layers containing V cells, where

$$V_{\text{dom}} = 2R_\beta K_{\text{dom}}(I_{\text{dom}} + J_{\text{dom}}), \tag{17}$$

R_β is the radius of the generalized neighborhood T'' (8), and I_{dom}, J_{dom}, and K_{dom} are the linear dimensions of the domain.

The problem of efficient parallel implementation is in constructing a calculation that is optimal from the point of view of minimizing the time and costs. In most studies on parallel calculations, the following ratio called, *paralleling efficiency*,

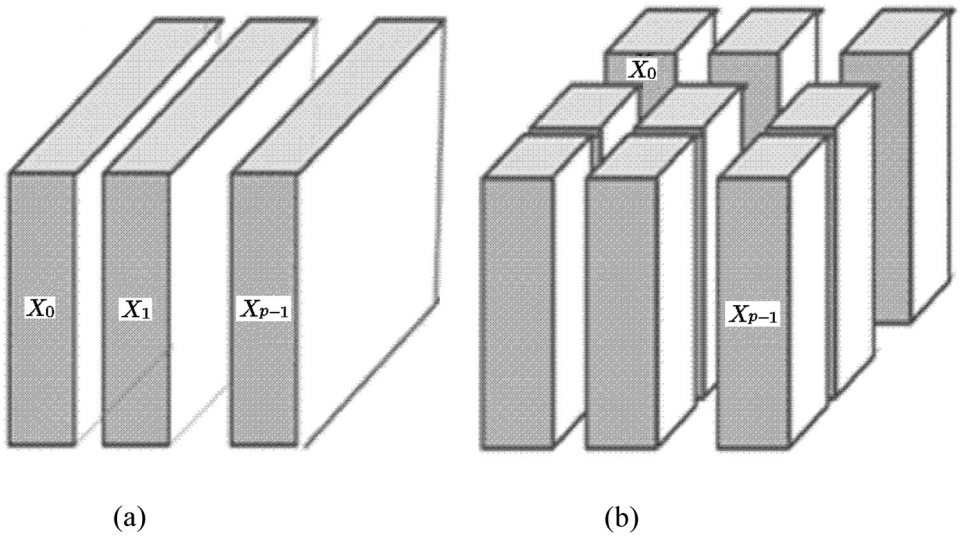

Fig. 5. Two methods of dividing a cellular array into domains.

$$\eta = \frac{\mathrm{Time}_1}{\mathrm{Time}_P} = \frac{\mathrm{Time}_1}{\mathrm{Time}_1 + l_\nu \mathrm{Time}_{\mathrm{ex}}} \tag{18}$$

is used as a quantitative estimate of parallel implementation. Here Time_1 is the calculation time of one iteration for one domain, Time_P is the calculation time of one iteration in parallel for P domains, $\mathrm{Time}_{\mathrm{ex}}$ is the time needed for one data exchange between P processes, and l_ν is the number of data exchanges in one iteration.

It follows from (18) that the efficiency is related to the model parameters by means of a quantity l_ν which for the regime (β, γ) (see the table) is

$$l_{\beta,\gamma} = |B_\beta|\,|\Theta| = mn = \lambda N n. \tag{19}$$

For the problem being considered, at a minimal β-block size $m = 9$ and at 20 ($n = 20$) cycles of Θ_{D} the number of exchanges at each iteration $l_{\beta,\alpha} = 180$, and at $m = 25$ $l_{\beta,\alpha} = 500$.

Substituting (17) and (19) into (18), we obtain the following relation between the efficiency and stochasticity:

$$\eta_{\beta,\gamma} = \frac{1}{1 + (\lambda/N)\,\sqrt[D]{P/N}\,(\tau_{\mathrm{ex}}/n\tau_{\mathrm{op}})}, \tag{20}$$

where τ_{ex} and τ_{op} are the time of interprocessor byte exchange and the execution time of the local operator, respectively.

From (20) we have three conclusions:

- The efficiency of parallel implementation is the greater, the less is the stochasticity of the SCA-model, which depends on the choice of block size in a block-synchronous regime.
- The efficiency weakly decreases with increasing number of parallel processes P.
- The more complicated is the calculation of local operators (the less is the ratio $\tau_{\mathrm{ex}}/n\tau_{\mathrm{op}}$), the greater is the efficiency.

The first conclusion makes it necessary to choose a minimal β-block size m in β-regimes. The question as to what extent m affects the simulation "quality," that is, the accuracy of representation of the phenomenon being simulated, remains open.

Few results of stochastic simulation on supercomputers are known so far [19, 20, 31]. They have revealed no difference between the asynchronous and block-synchronous regime simulations.

The second conclusion calls for the choice of domains having large sizes but the same sizes of "shadow" layers (domains in the form of cubes and squares for which the ratio V/P at equal N is smaller).

Finally, the third conclusion implies that clusters with fast interprocessor communications should be preferred.

The requirement of correctness (6) of collective communications (MPI-Sendrecv [32]) imposes additional constraints on the organization of interprocessor exchanges. This means that although β-stage $\Pi_l \in \Pi$ is chosen randomly, it must be the same at any time in all domains. For this, at each iteration a random sequence of m natural numbers must be formulated in one of the domains and translated to all processors.

3.3. Simulation Results

The process of permeation was experimentally simulated in two stages. At the first stage, permeation was simulated in a serial regime for a $200 \times 200 \times 500$ cellular array fragment on an Intel Core-i7 computer (2.66 GHz). At this stage, the program modules of the convection and diffusion operators were debugged, and the needed number of cycles of the diffusion operator was specified to provide full smoothness of the liquid surface.

The parallel version was investigated on clusters NKS-30T Intel Xeon 5540 (at the Siberian SuperComputer Center of the Institute of Computational Mathematics and Mathematical Geophysics SB RAS, Novosibirsk) and MVS-10p Intel Xeon Phi 7120D (SuperComputer Center, Moscow). The following initial conditions were taken: all soil pores were empty, and a small amount of liquid ($16 \cdot 10^6$ particles (60 grams)) was poured on the soil surface. The boundary conditions along the i and j axes were periodic. The programming work was made in C^{++} programming language using the MPI library [32].

The following parameters and properties of parallel implementations of the SCA-model were studied:

1. *Depth and rate of liquid permeation into the soil.* The dependence of the permeation depth on the number of iterations (model time) is shown by the curve (Fig. 6). This dependence is not affected by the parameters of the SCA simulation algorithm (the regime, the number of parallel processes, and the topology of their connections). One can see in Fig. 6 that the permeation rate varies with depth, which indicates that the porous structure is inhomogeneous. Specifically, one can see voids at depths $k \approx 828$ and $k \approx 660$.

2. *Efficiency of paralleling versus stochasticity of the algorithm.* Figure 7 shows the full calculation time and the time of interprocessor exchanges as a result of simulation for various model stochasticity values. The simulation was done in parallel on 10 cores of MVS-10p cluster at one-dimensional decomposition (Fig. 4a).

The lines in Fig. 7 show that the efficiency

$$\eta = \frac{\mathrm{Time}_P - \mathrm{Time}_{\mathrm{ex}}}{\mathrm{Time}_P}$$

practically does not depend on the size of the β-block and does not exceed 0.2 when the β-block size is larger than the minimal size ($m > 3^2$).

3. *Speeding up of calculations versus the number of processors and the topology of core communications*. The simulation was made on the MVS-10p cluster at two kinds of the simulation space decomposition. The simulation results in Fig. 8 show that calculations at two-dimensional decomposition are more efficient. One can also see that paralleling of $P \geq 10$ processes is not reasonable for the SCA being considered.

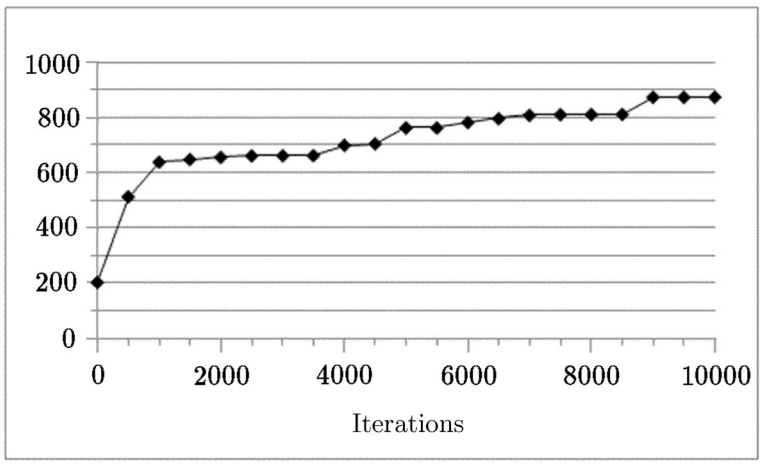

Fig. 6. Liquid permeation depth versus time at model implementation in (β, γ) regime on $P = 4 \times 10 = 40$ processors.

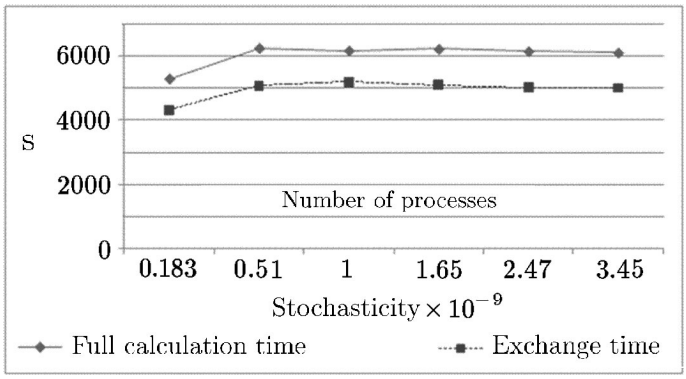

Fig. 7. Calculation time versus SCA-model stochasticity $\aleph_{\beta,\gamma}$ on $P = 10$ processors at one-dimensional decomposition of the simulation space. The stochasticity values correspond to the following sizes of β-blocks of the SCA: $|B_\beta| = \{3^2, 5^2, 7^2, 9^2, 11^2, 13^2\}$.

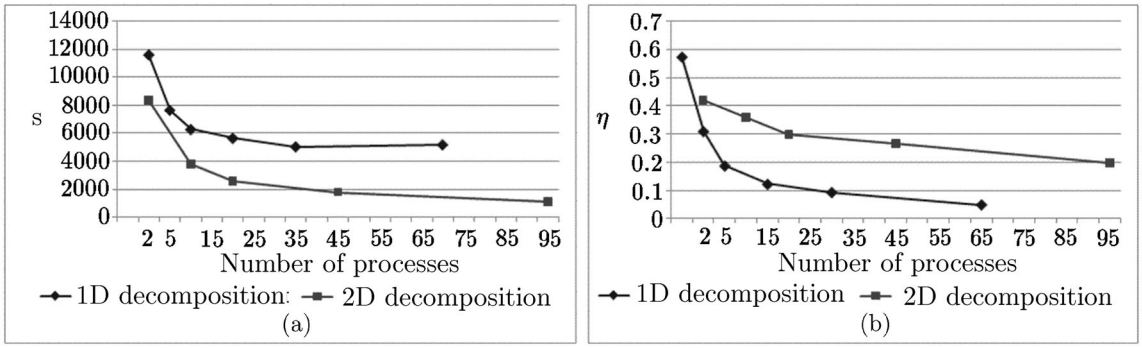

Fig. 8. (a) Execution time of $T = 1000$ iterations; (b) efficiency of parallel implementation versus the number of parallel processes at one- and two-dimensional decomposition of the simulation domain.

4. CONCLUSIONS

The parallel implementation of the stochastic model of permeation has made it possible to obtain stochasticity and efficiency as functions of the parameters of parallel implementation and estimate their influence on the major characteristics of the model. This is necessary for a reasonable choice of the model parameters. The results of the investigation are as follows:

(1) For an efficient parallel implementation of a spatially stochastic SCA it must be transformed to a block-synchronous regime (β-regime).

(2) SCA stochasticity depends on the size of the β-block. Increasing this size does not speed up the calculations.

(3) Correctness of parallel calculations demands some coordination between the stages of the β-regime in all processors.

ACKNOWLEDGMENTS

This work was supported by the Presidium of RAS, project no. 15-6-2016.

REFERENCES

1. Boccara, N., *Reaction-Diffusion Complex Systems*, Berlin: Springer, 2004.
2. Vanag, V.K., *Dissipativnye struktury v reaktsionno-diffuzionnykh sistemakh. Eksperiment i teoriya* (Dissipative Structures in Reaction-Diffusion Systems: Experiment and Theory), Izhevsk: IKI, 2008.

3. Adamatzky, A., *Reaction-Diffusion Automata: Phenomenology, Localizations, Computation*, Heidelberg: Springer, 2013.
4. Ziff, R.M., Gulari, E., and Barshad, Y., Kinetic Phase Transitions in an Irreversible Surface-Reaction Model, *Phys. Rev. Lett.*, 1986, vol. 56, no. 24, pp. 2553–2556.
5. Chatterjee, A. and Vlaches, D.G., An Overview of Spatial Microscopic and Accelerated Kinetic Monte-Carlo Methods, *J. Comp. Aided Matter Des.*, 2007, vol. 14, pp. 253–308.
6. *Advances in Chemical Physics: Monte Carlo Methods in Chemical Physics*, vol. 105, Prigogine, I. and Rice, S.A., Eds., Willey, 2009.
7. Matveev, A.V., Latkin, E.I., Elokhin, V.I., and Gorodetskii, V.V., Turbulent and Stripes Wave Patterns Caused by Limited CO_{ads} Diffusion during CO Oxidation over Pd(110) Surface: Kinetic Monte Carlo Studies, *Chem. Eng. J.*, 2005, vol. 107, pp. 181–189.
8. Elokhin, V.I., Matveev, A.V., and Gorodetskii, V.V., Self-Oscillations and Chemical Waves in CO Oxidation Reaction on Pt and Pd: Kinetic Monte Carlo Models, *Kin. Kat.*, 2009, vol. 50, no. 1, pp. 45–53.
9. Vanag, V.K., Study of Spatially Extended Dynamical Systems Using Probabilistic Cellular Automata, *Usp. Fiz. Nauk*, 1999, vol. 169, no. 5, pp. 481–505.
10. Alekseev, D.V., Kazunina, G.A., and Cherednichenko, A.V., Cellular Automaton Simulation of the Fracture Process for Brittle Materials, *Prikl. Diskr. Mat.*, 2015, vol. 28, no. 2, pp. 103–117.
11. Nurminen, L., Kuronen, A., and Kaski, K., Kinetic Monte Carlo Simulation of Nucleation on Patterned Substrates, *Phys. Rev. B*, 2000, vol. 63, pp. 035407-1–035407-7.
12. Vitvitsky, A.A., Cellular Automata with a Dynamic Structure for Simulate the Growth of Biological Tissues, *Num. An. Appl.*, 2014, vol. 7, no. 4, pp. 263–273.
13. Bandman, O.L., Cellular-Automata Models of Natural Processes, Implementation on Supercomputers, *Prikl. Diskr. Mat.*, 2017, no. 35, pp. 102–121.
14. Bandman, O.L. and Kireeva, A.E., Stochastic Cellular Automata Simulation of Oscillations and Autowaves in Reaction-Diffusion Systems, *Num. An. Appl.*, 2015, vol. 8, no. 3, pp. 208–222.
15. Wolfram, S., *An Elementary Introduction to the Wolfram Language*, USA: Wolfram Media, 2017.
16. Wolfram MATHEMATICA, https://www.wolfram.com/mathematica/.
17. Lubachevsky, B.D., Efficient Parallel Simulations of Asynchronous Cellular Arrays, *Complex Syst.*, 1987, vol. 1, no. 6, pp. 1099–1123.
18. Overeinder, B.J. and Sloot, P.M.A., Application of Time Warp to Parallel Simulations with Asynchronous Cellular Automata, *Proc. European Simulation Symposium*, The Netherlands, 1993, pp. 397–402.
19. Bandman, O., Parallel Simulation of Asynchronous Cellular Automata Evolution, Cellular Automata, *Proc. ACRI*, 2006, Springer, 2006. pp. 41–48.
20. Kireeva, A., Parallel Implementation of Totalistic Cellular Automata Model of Stable Patterns Formation, Parallel Computing Technologies, *Proc. PaCT 2013*, Malyshkin, V., Ed., Springer, 2013, pp. 330–343.
21. Bandman, O., Implementation of Large-Scale Cellular Automata Models on Multi-Core Computers and Clusters, *Proc. HPCS 2013*, Helsinki, 2013, pp. 304–310.
22. Achasova, S., Bandman, O., Markova, V., and Piskunov, S., *Parallel Substitution Algorithm. Theory and Application*, Singapore: World Scientific, 1994.
23. Bandman, O.L., Cellular Automata Models of Spatial Dynamics, *Sist. Informatika*, Novosibirsk: SB RAS Publ. House, 2006, no. 10, pp. 59–111.
24. Achasova, S.M. and Bandman, O.L., *Korrektnost' parallel'nykh protsessov* (Correctness of Parallel Processes), Novosibirsk: Nauka, 1990.
25. Bandman, O., Parallel Composition of Asynchronous Cellular Automata Simulating Reaction-Diffusion Processes, *Proc. ACRI 2010*, Springer, 2010, pp. 395–398.
26. Bandman, O., Parallelization Efficiency Versus Stochasticity in Simulation Reaction-Diffusion by Cellular Automata, *J. Supercomp.*, 2016, vol. 73, iss. 2, pp. 687–699.
27. Bandman, O.L., Functioning Modes of Asynchronous Cellular Automata Simulating Nonlinear Spatial Dynamics, *Prikl. Diskr. Mat.*, 2015, vol. 27, no. 1, pp. 105–119.
28. Von Neumann, J., *Theory of Self-Reproducing Automata*, Burks, A.W., Ed., London: Univ. of Illinois Press, 1966.
29. Wolfram, S., *Cellular Automata and Complexity: Collected Papers*, USA: Westview Press, 2009.
30. Bandman, O., 3D Cellular Automata Model of Fluid Permeation through Porous Material, Parallel Computing Technologies, *Proc. PaCT 2013*, Malyshkin, V., Ed., Springer, 2013, pp. 278–290.
31. Nedea, S.V., Lukkien, J., Jansen, A., and Hilbers, P., Methods for Parallel Simulations of Surface Reactions; URL: http://arXiv.org > physics > arXiv:physics/0209017.
32. Olenev, N.N., *Osnovy parallel'nogo programmirovaniya v sisteme MPI* (Fundamentals of Parallel Programming in the MPI System), Moscow: Publ. House of Comp. Center of RAS, 2005.