ELSEVIER

Theoretical Computer Science 217 (1999) 99–113

Theoretical Computer Science

Multilayered cellular automata

Stefania Bandini [*], Giancarlo Mauri

Department of Computer Science, University of Milan, Via Comelico 39, 20135 Milan, Italy

Abstract

In this paper multilayered cellular automata are formally defined as a generalization of multilayered automata networks. They are hierarchically organized on the basis of nested graphs, and can show different kinds of dynamics, which allow to use them to model, e.g., complex biological systems comprised of different entities organized in a hierarchical framework. Finally, the simulation of the dynamic regulation of calcium-ion distribution in the subcompartments of a living cell by means of Multilayered Cellular Automata is presented, as an example that allows to show their modeling power. © 1999—Elsevier Science B.V. All rights reserved

Keywords: Cellular automata; Reaction–diffusion

1. Introduction

In this paper a formal definition of *multilayered automata networks* and a derived definition of *multilayered cellular automata* and the related dynamics are presented in order to give a formal basis for those models which have been successfully developed and implemented adopting the general computational features of cellular automata, but introducing explicit notions and structures in order to represent *hierarchical* features. Some examples refer to complex biological processes produced by the interaction of several subsystems: the simulation of cellular interaction [1, 4], of HIV infection in the immune system [10], and of calcium-ion distribution in the living cell [2].

Multilayered automata networks are here defined as a generalization of the definition in terms of graphs of automata networks given in [8]. Like in [8] for cellular automata, the definition of multilayered cellular automata is then derived as a particular case of multilayered automata networks by introducing the notion of cellular space (a regular grid on the highest level of the hierarchy).

The main feature of the proposed definitions consists of the explicit introduction of a hierarchical structure based on *nested graphs*, which are graphs composed by vertices and arcs where each vertex can be in turn a nested graph of lower level.

[*] Corresponding author. E-mail: bandini@dsi.unimi.it.

By introducing states and transition functions, a multilayered automata network is directly obtained from the nested graph structure. Furthermore, the different possibilities for the dynamical evolution of a multilayered automata network, i.e. *sequential*, *synchronous* or *combined*, are defined.

A non formal notion of nested graph has been previously introduced in [10] for the computational treatment of hierarchical cellular automata in an object oriented programming paradigm. A cellular automaton model representing a multilayered structure (called hyper-cellular automaton) and adopting a combined dynamics for representing and simulating biological phenomena as a reaction–diffusion model was described in [1, 2].

In the first part of this paper, the formal definitions of multilayered automata networks, multilayered cellular automata and the related dynamics are given. In the second part, a description of the simulation of the dynamic regulation of calcium-ion distribution in the subcompartments of a living cell by means of multilayered cellular automata is given, as an example that allows to show their modeling power.

2. Nested graphs

Let us recall first of all some fundamental definitions about graphs.

Definition 1. A *graph* is a pair

$$G = \langle V, l \rangle$$

where
- V is a finite or countable set of elements called *vertexes* or *nodes*;
- $l : V \to \wp(V)$ is the *neighborhood function*, which determines for each node $v \in V$ the set of *adjacent* nodes.

The set of graphs will be denoted by $\mathscr{G}$; given $G = \langle V, l \rangle \in \mathscr{G}$, $v \in V$, for sake of simplicity, we will use the notation l_v instead of $l(v)$ when no ambiguity arises.

Definition 2. A graph $G = \langle V, l \rangle \in \mathscr{G}$ is *locally finite* if and only if $\forall v \in V \ (|l_v| < \infty)$.

In the following, we will consider only locally finite graphs. Now, we can formalize the hierarchical structure, through the recursive definition of the set $\mathscr{HG}$ of nested graphs.

Definition 3.
- The set of nested graphs of level 0 is the set of graphs:

$$\mathscr{HG}_0 = \mathscr{G};$$

- The set of nested graphs of level $i + 1$ is the set of pairs:

$$\mathscr{HG}_{i+1} = \{ \langle G, \varphi \rangle \mid G = \langle V, l \rangle \in \mathscr{G} \text{ and } \varphi : V \to \mathscr{HG}_i \},$$

where G is a graph called the *support graph* of the pair $\langle G, \varphi \rangle$ and φ is a map which associates with every node of G a nested graph of level i;
- The set $\mathscr{HG}$ of nested graphs is the union of all the sets $\mathscr{HG}_i$ for $i \in N$.

Notation. In the following, a nested graph of level i will be denoted by $HG^i = \langle \langle V^i, l^i \rangle, \varphi^i \rangle$, and each node of HG^i will be identified by a pair $\langle i, v \rangle$ where i denotes the level and v is the name of the node in the graph of level i.

Definition 4. A nested graph $HG^i \in \mathscr{HG}_i$ is locally finite if and only if
- $i = 0$, HG^i is a locally finite graph;
- $i > 0$, $HG^i = \langle G, \varphi \rangle$, G is a locally finite graph and $\mathrm{Im}(\varphi) \subseteq \mathscr{HG}_{i-1}$ is a set of locally finite nested graphs.

We must now redefine the notion of neighbourhood of a node, taking into account the hierarchical structure, too.

Definition 5. Let $HG^i \in \mathscr{HG}_i$, with $i > 0$. The *neighbourhood* of a node $\langle i, v \rangle$ is defined by the neighbourhood function l^i related to the graph of that level. Furthermore, we define the *nesting* neighborhood function δ, which associates to the node $\langle i, v \rangle$ the set $\delta(v) = V_v^{i-1}$ of nodes of the graph

$$\varphi^i(v) = HG_v^{i-1} = \langle \langle V_v^{i-1}, l_v^{i-1} \rangle, \varphi_v^{i-1} \rangle$$

at the level $i-1$ (or if $i - 1 = 0$, $\varphi^i(v) = HG_v^{i-1} = \langle V_v^{i-1}, l_v^{i-1} \rangle$).

As an example, let us consider the nested graph of level 2 illustrated in Fig. 1. Both the level 0 and the level 1 are constituted by a family of graphs, each one corresponding to a node at the preceding level, and the level 2 is composed by one connected graph consisting of two vertexes. If we take two vertexes of two distinct

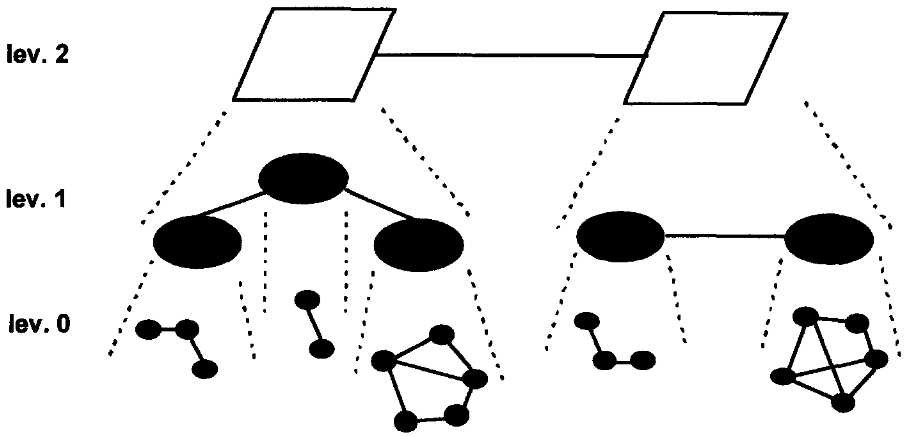

Fig. 1. A nested graph of level 2.

subgraphs of level 1, there is no path from one to the other given by arcs of level 1, but if the two subgraphs correspond to adjacent vertexes of level 2, then a path which connects them exists. It is given by the arc of level 2 which links the two counterimages of the given subgraphs. The same considerations hold for the vertexes of the subgraphs of level 0, among which, although a path of arcs of level 0 that connects them does not exist, a path composed by arcs of level 1 or 2 may exist.

3. Multilayered automata networks and multilayered cellular automata

In this section we will give the definition of an extension of the automata network introduced in [8] called *multilayered automata network.*

3.1. The structure

Our definition will be based on the above introduced nested graphs: if we have a graph of level 0, then each node will be a finite state automaton, while for a graph of level $k \neq 0$, every node will correspond to an automata network of level $k-1$. In the particular case in which $k=0$ the multilayered automata network has only one level, giving an automata network in the sense of [8].

The state of a node v at any level i can be modified as a function of the states of nodes in its "horizontal" neighbourhood (i.e., l_v^i) or as a function of the states of the nodes belonging to its associated automaton at level $i-1$.

Definition 6. (a) A *multilayered automata network* (MAN) of level 0 is a triple

$$A^0 = \langle G^0, Q^0, F^0 \rangle,$$

where

- $G^0 = \langle V^0, l^0 \rangle \in \mathscr{G}$ is the *support graph* of A^0;
- Q^0 is the finite set of *states* which the nodes of the graph can assume;
- $F^0 = \{f_v^0 \mid v \in V^0\}$ is a set of transition functions, where $f_v^0 : (Q^0)^{l_v} \to Q^0$.

(b) For $i > 0$, let $\mathscr{A}^{i-1} = \{A_1^{i-1}, A_2^{i-1}, \ldots\}$ be a finite or countable set of MAN's of level $i-1$, such that different MAN's in the set have different support graphs. A multilayered automata network of level i is a fourtuple

$$A^i = \langle HG^i, Q^i, F^i, g^i \rangle,$$

where

- $HG^i = \langle G^i, \varphi^i \rangle \in \mathscr{HG}_i$, $G^i = \langle V^i, l^i \rangle \in \mathscr{G}$ is the *support graph* of A^i;
- $\mathrm{Im}(\varphi^i) = \{HG^{i-1} \in \mathscr{HG}_{i-1} \mid \exists A_k^{i-1} \in \mathscr{A}^{i-1}$ such that HG^{i-1} is the support graph of $A_k^{i-1}\}$;
- Q^i is a finite set of *states* the elements of V^i can assume;
- $F^i = \{f_v^i \mid v \in V^i\}$ is a set of transition functions such that

$$f_v^i : (Q^i)^{l_v^i} \times Q^i \to Q^i;$$

- $g^i = \{g_v^i \mid v \in V^i\}$ is a set of indexed functions such that

  $$g_v^i : (Q_v^{i-1})^{\delta(v)} \to Q^i$$

  where Q_v^{i-1} is the set of the states of the network having $\varphi^i(v)$ as its support graph.

The elements of $\mathscr{A}^{i-1}$ are called *nodes* of level $i-1$ of A^i : recursively, for $i-1>0$:

$$\mathscr{A}^{i-2} = \{A^{i-2} \mid \exists A_k^{i-1} \in \mathscr{A}^{i-1} \text{ such that } A^{i-2} \text{ is a node of level } i-2 \text{ of } A_k^{i-1}\}.$$

We say that the elements of $\mathscr{A}^{i-2}$ are the nodes of level $i-2$ of A^i, until the nodes of level 0 are defined. A^i is the node of level i of $\mathscr{A}^i$.

3.2. *The dynamics*

Let us now introduce for the multilayered automata networks the equations which describe the network dynamics, i.e. the sequence of the states depending on time. We will consider a totally ordered discrete set of temporal steps $T = \{t_0, t_1, \ldots\}$, where t_0 is the minimum element. Such a set represents the time steps for observing the evolution of the states. Since the "next state" of each node in the network depends on the "current state" of the node and of its (horizontal or vertical) neighbours, the evolution of the networks depends also from the updating policy we choose. As for the automata networks [8], it is possible to give different types of dynamics also for the multilayered automata networks, depending on the fact that the state transition occurs in parallel in all the nodes of the network or that only a node is updated at every time step, in a sequential way.

Notation. Given a multilayered automata network $A^i = \langle HG^i, Q^i, F^i, g^i \rangle$ of level i (in the case of $i=0$ it is $A^0 = \langle G^0, Q^0, F^0 \rangle$) let
$V = \{v \mid \exists A_k^i$ node of level i of A^k such that v is a node of the support graph of $A_k^i\}$;
$Q = \{q \mid \exists A_k^i$ node of level i of A^k such that q belongs to the set of states of $A_k^i\}$.
$q(t) : V \to Q$ (with $t \in N$) the state of the nodes at the instant t.

3.2.1. *Synchronous iteration*

In this case, all the nodes are updated in parallel, with the following rules:

(a) Let v be a node of the support graph of a network $A = \langle\langle V^0, l^0\rangle, Q^0, \{f_k^0 \mid v \in V^0\}\rangle$ of level 0. Then

$$q_v^0(t+1) = f_v^0(q(t)\,|_{l_v^0}),$$

where with $q(t)\,|_{l_v^0}$ we denote the restriction of $q(t)$ to l_v^0.

(b) Let v be a node of a graph of a network $A = \langle\langle\langle V^i, l^i\rangle, \varphi^i\rangle, Q^i, \{f_v^i \mid v \in V^i\}, \{g_v^i \mid v \in V^i\}\rangle$ f level $i>0$. Then:

$$q_v^i(t+1) = f_v^i(q(t)|_{l_v^i},\ g_v^i(q(t)|_{\delta(v)})).$$

3.2.2. Sequential iteration

In the *sequential* iteration the states of the nodes are updated one at a time, by levels from k to 0 and, at every level, following a given schedule. More precisely, within each V_k^i, set of the nodes of the support graph of a network A_k^i, a total order relation $<$ is defined such that $\langle V_k^i, < \rangle$ is a well ordered set. Hence, we have the rules:

(a) For the level 0 (with the same notation of the synchronous iteration):

$$q_v^0(t+1) = f_v^0(y_v^0)$$

with $y_v^0 : l_v^0 \to Q^0$ such that

$$y_v^0(p) = \begin{cases} q_p^0(t+1) & \text{for } p<v, \\ q_p^0(t) & \text{otherwise.} \end{cases}$$

(b) For a level $i>0$:

$$q_v^i(t+1) = f_v^i(y_v^i, g_v^i(q(t)|_{\delta(v)}))$$

with $y_v^i : l_v^i \to Q^i$ such that

$$y_v^i(p) = \begin{cases} q_p^i(t+1) & \text{for } p<v, \\ q_p^i(t) & \text{otherwise.} \end{cases}$$

3.2.3. Combined interaction

In the case of the multilayered automata networks, it is possible to have a dynamics where the levels are updated in a sequential way, while at each level the updating is synchronous, with the following rules:

(a) for the level $i=0$:

$$q_v^0(t+1) = f_v^i(q_p^0) \quad \text{for } p \in l_v^0.$$

(b) for the levels i with $i>0$:

$$\forall i = 1,\dots,K, \quad \forall v \in V^i \quad q_v^i(t+1) = f_v^i(q_e^i, q_p^i(t)) \quad \text{for } p \in l_v^i,$$

where $q_e^i = g^i(v) \in Q^i$.

3.3. Multilayered cellular automata

We can now define multilayered cellular automata as a special case of multilayered automata networks.

Definition 7. A graph $G=(V,l)$ is a *cellular space* of dimension d iff

1. $V = Z^d$;
2. $\forall k \in Z^d, \ \forall i,j \in V \ j \in l(i) \leftrightarrow j+k \in l(i+k)$.

Definition 8. A *multilayered cellular automaton* is a multilayered automata network

$$A^i = \langle HG^i, Q^i, F^i, g^i \rangle$$

whose support graph G is a cellular space, equipped with a synchronous dynamics.

4. Modeling complex biological systems with multilayered cellular automata

The general properties of biological systems arise from complex interactions of lower level subsystems, which can in turn be decomposed, down to the stochastic interactions among single molecules. The first task in modeling biological systems is therefore to define granularity and viewpoint, i.e. the degree of detail required for an adequate description of the system.

In particular, if modeling is aimed at gathering some insight into the mechanisms which underlie the process of interest – rather than merely reproducing the observed behaviour – the viewpoint will often have to be multiple, i.e. it will be necessary to simulate various aspects and subsystems at different degrees of detail and with different temporal and/or spatial resolution. Further problems often arise in defining the simulation space, as the topological features of biological systems are in many cases complex and heterogeneous.

These observations suggest that multilayered cellular automata, as previously defined, endowed with the properties of parallel and local processing of information by means of mutual interactions between neighboring elements at different levels, may constitute an appropriate framework to build efficient and accurate models of biological systems.

Cellular automata are the best characterized local and parallel computational models. However, cellular automata in their classical form were not designed to cope with topological heterogeneity and with the coexistence of multiple subsystems and different "particle" populations. In classical applications of cellular automata, such as fluid dynamics, the cellular space [8, 11] is homogeneous, one or a few types of particles exist, interactions among particles are mostly limited to collision, and neighborhood rules govern the movement of particles. Modeling biological systems will require introducing the representation of several different types of elements and the specific reactions among such elements, as well as the differential diffusion of the mobile species. To this purpose, it can be convenient to split the two aspects of *reaction* and *diffusion*, moving to more complex computational models (an example within the classical framework of standard cellular automata can be found in [5]).

Two-layers multilayered cellular automata (hyper-cellular automata) have been developed and employed to model biological systems [1]: the first level constitutes a bidimensional cellular space (*diffusion space*), while at the second level a totally connected graph corresponds to each cell, so as to generate an intrinsically parallel and local *reaction space*. For the biological systems faced so far, a combined dynamics appeared the most convenient, i.e. updating by the alternate synchronous application of reaction rules or of diffusion rules. *Reaction rules* dictate status changes. The status of each entity (here named as *actor*) is the set of values denoting specific attributes associated to it in order to determine its properties and behaviour; such status is changed (*reaction*) based on the status of all the other entities momentarily present in the same cell; thus, these rules apply to the totally connected graphs which correspond, at the other level of the nested graph, to single sites. *Diffusion rules* determine the diffusion of entities on the cellular space. They operate on the higher-level cellular space to possibly displace the entities from the site where they are momentarily located to one of the neighboring sites.

Modeling a complex biological system entails a series of further complications, such as heterogeneous topological domains (*compartments*), differential constraints to diffusion, delays in the response of specific actors or subsystems to specific stimuli, and the need to simulate different aspects (or subsystems) with different temporal/spatial resolution.

The purpose of this example is to describe the development of a two-levels multilayered cellular automaton, which has evolved from hyper-cellular automata models of reaction–diffusion systems, to model the distribution of calcium ions (Ca^{2+}) in the subcompartments of a living cell.

4.1. *Modeling calcium-ion distribution in the living cell*

Many biochemical processes in living cells are regulated by the concentration of free calcium ions (Ca^{2+}), from contraction of muscle fibres to secretion, proliferation and cellular death. Such processes are activated by widely different concentrations of Ca^{2+}, from fractions to hundreds of micromoles per litre (μM). Thus, living cells have developed powerful and sophisticated means to maintain very low basal calcium concentrations (about 0.1 μM, versus a more than 10^4 higher concentration in extracellular fluids) and to carefully regulate elevations in Ca^{2+} concentration in response to intracellular as well as extracellular stimuli.

Regulatory systems range from proteins which "pump" out calcium ions by coupling this process to energy consumption or other ion fluxes, to proteic pores or "channels" – finely regulated by several possible mechanisms – in the plasmamembrane surrounding the cell; proteins are present which can bind huge amounts of Ca^{2+}, and specific subcellular organelles are designed to actively store (by means of other "pumps") and release Ca^{2+} on demand (by means of other "channels") [9]. Such a multifarious and complex regulation, over more than 3 decades of Ca^{2+} concentration, raises some modeling problems:

- a classical representation of calcium ions as single particles would require up to about 10^6 particles per μm^3 of simulated cell volume (the size of an average cell is about 500 μm^3), and a lattice-gas-type automaton [7] would require about as many nodes (over 10 million for a two-dimension thin section of an average cell); furthermore, to obtain a reasonable diffusion speed for particles the simulation time step should be in the order of microseconds; on the other hand, molecular simulation of calcium ions is somehow required to reproduce stochastic fluctuations in calcium-regulated processes;

- the Ca^{2+} regulation system, like most biological systems, is comprized of multiple biochemical subsystems where many different elements interact in various ways; therefore, the model must be able to host several different species of "particles", and each such element must "play its role as an actor on stage" [3];
- for many sub-systems, rates and modulatory mechanisms are reasonably well characterized; therefore, although this is in contradiction with the need for low-level molecular simulation, it would be very convenient to directly introduce such knowledge as high-level rules in the simulation system.

Multilayered cellular automata offer an effective approach to the question. Once appropriate time and space scales are set, then molecules present in low concentration will be allowed to move around according to standard rules of simulated diffusion; all molecules – proteins, channels, pumps as well as electrolytes – located in the same cell of the automaton will appropriately react.

However, at least two problems call for specific solutions:

- species which are present at high concentrations (many particles per node) should be looked at in terms of local concentration, rather than single particles; this entails defining specific approaches to simulate diffusion of such species as well as their molecular interaction with other elements, preserving the stochastic aspects of such phenomena;
- several different compartments must be defined in the simulation area (cellular space); diffusion across boundaries must be prevented, or differentially regulated by *transport* mechanisms; this again requires a specific approach to diffusion; furthermore, in a bidimensional cellular space, where two compartments cross each other a discontinuity arises in at least one of them, which makes bidimensional mapping inadequate for multicompartmental models.

4.2. *The multilayered cellular automaton*

In order to tackle these challenges, a multilayered cellular automaton has been implemented by introducing three main features in addition to the properties described above for two-layered reaction–diffusion one.

a. Collective actors. The multilayered cellular automaton is based on a cellular space simulating the living cell; objects endowed with status attributes simulate chemical substances present in the living cell, and will be called "actors"; actors can move to neighboring cells according to diffusion rules and can react with other actors momentarily present in the same cell according to reaction rules (which change status attributes). Calcium ions, which may be present at high concentrations (up to >100 per cell), are not considered singularly. Rather, collective actors are introduced in the automaton; such collective actors have a fixed location, are present in all cells where Ca^{2+} can occur, and have a status attribute dedicated to represent the number of Ca ions momentarily present in the cell. Single Ca ions are represented as bits in a fixed-length string

[4, 6], the concentration attribute, which is randomly scrambled at each step in the life of the system; Ca^{2+} concentration will intervene in any reaction which is influenced by Ca^{2+}, by means of string-matching operations on the concentration string. This ensures that Ca^{2+}-dependence of any process is modeled even in its stochastic aspects. Thus, the function of collective actors is to increase the resolution of the model down to the molecular level, without concurrently requiring shorter time steps and a finer spatial grid; they constitute the "underground" portion of the model, working like lower-level automata.

b. Cellular space and compartments. The existence of different compartments is handled by assigning each element in the cellular space to a specific compartment (e.g. extracellular, cytoplasm, intracellular organelles, endoplasmic reticulum). Specific status fields will determine which compartments each *actor* can reside in. In general, barriers to diffusion will exist at boundaries between cells belonging to different compartments; however, specific status attributes may endow actors with the capability, or possibly the duty, of crossing such boundaries. Conversely, diffusion will generally be free among cells belonging to the same compartment.

All this profoundly affects the fundamental properties of the cellular space, which becomes a graph where different kinds of vertices and arcs are allowed (i.e. different compartments and direct connection or different boundaries between cells).

A further generalization with respect to classical cellular space is introduced in the automaton to cope with the limitations arising from a bidimensional simulation area: the cellular space is created as a graph where each vertex (cell) is connected to *at least* four other vertices, by means of arcs of possibly different kinds (depending on the compartments the two connected cells belong to). A subpopulation of the vertices and arcs map, respectively, to cells in a bidimensional simulation area and to the neighborhood relations among such cells; this makes it possible to display a representation of the simulation area. The other vertices map to cells outside the bidimensional simulation area and provide – together with the arcs originating from them – (i) the continuity when two compartments cross each other and (ii) a virtual prolongation of the bidimensional simulation area where this has to be arbitrarily truncated.

c. The rules. In addition to the main sets of rules mentioned above, i.e. reaction and diffusion rules, which we may more appropriately call *molecular* reaction/diffusion rules, the automaton contains two further sets of rules, which govern *collective* reaction and diffusion.

- *Molecular diffusion rules* apply to all *mobile* actors; they govern the movement from a site to one of the neighboring sites (be they inside or outside the displayable bidimensional area). For each actor in each location, the applicable rules are directly determined by the status attributes of the actors, by the type of the cell and by its neighborhood relations (i.e. by the type of the corresponding vertex and of the arcs originating from it): in other words, the diffusion rule A–C–C' determines how an actor with diffusion attribute A which is located in a cell of a specific compartment C can move to a neighboring cell, belonging to compartment C'.

- *Reaction rules* apply to all actors, simple as well as *collective* ones; reactions occur according to a totally-connected-graph scheme, i.e. they involve, for each cell, all actors momentarily present in that particular cell. However, reaction fields in the structure of each actor determine, by means of string-matching rules, the other actors they can significantly interact with. Object oriented programming makes it easy to create *methods* for each actor which produce the desired changes in status attributes following recognition of a matching actor. Self reaction is also allowed to reproduce delays (countdown in a specific field) and complex responses (stepping through a series of states).
- *Collective reaction rules* are just a subclass of reaction rules, with the exception that they do not apply to the totally connected graph of actors simultaneously present in the same cell, but rather on a reduced graph where each *collective actor* is connected to all other actors in the cell which possess the appropriate matching string. These rules are generally more complex than *molecular* reaction rules in that string matching of the *concentration attribute* to appropriate strings presented by the counterpart can determine graded changes in specific status attributes, thereby simulating graded activation/inactivation of proteins and mechanisms as functions of Ca^{2+} concentration, preserving in the mean time the stochastic aspects of molecular interactions. Reactions between collective actors are allowed to simulate, for example, the binding/unbinding of calcium-ions to Ca^{2+}-buffering proteins, through the comparison of the concentration strings of collective actors which respectively represent the populations of free and bound calcium ions in the cell. This simulates slowing down of Ca^{2+} diffusion by the buffering proteins.
- *Collective diffusion rules* apply to collective actors only, and in particular to collective actors located in adjacent cells. Adjacency between two cells for collective diffusion is determined by the existence of permissive arcs between the two vertices that represent the two cells, in the graph which defines the simulation space. These rules consist in the swapping of variable length substrings between the *concentration attributes* of the two involved collective actors, and simulate the movement (diffusion) of particles present at high concentration, by means of changes in status attributes, rather than location, of specific actors. Thus, as we shall shortly see, these rules must be formally distinguished from both reaction and molecular diffusion rules.

Let us indicate by D_M and D_C the procedures to update the system by applying the *molecular diffusion* rules and the *collective diffusion* rules, respectively, and by R_M and R_C the procedure to update the system by applying the *molecular* and *collective reaction* rules, respectively. Let us also indicate by G_1 the graph where each vertex corresponds to a cell in the simulation space and the arcs define the neighborhood relation, and by G_i $(i=1,N)$ – where N is the total number of vertices of G_1 – the totally connected graph dynamically generated at each step by the set of actors currently present in the cell mapped by vertex i of G_1.

We can observe that R_M and R_C are formally similar, in that they apply to G_i $(i=1,N)$ and work independently for each cell, i.e. the execution of these procedures on the whole system is exactly equivalent to the parallel execution of the same rules on each single cell of the automaton. Furthermore, the execution of R_M and R_C do not produce any displacement of actors in the simulation space but only changes in the status attributes.

Conversely, execution of the D_M procedure is exactly equivalent to the parallel application of molecular diffusion rules to each single mobile actor. This procedure brings about changes in the location of actors in the simulation space but does not affect any status attribute.

This reproduces the two-step updating procedure of a reaction–diffusion hyper cellular automaton, where the two steps apply to two different domains – i.e. the cohort of actors for the diffusion step and the set of cells for the reaction step.

Execution of the D_C procedure, finally, applies to a third, different domain, because it is equivalent to the parallel application of the corresponding rules to each *arc* in G_I. Furthermore, it simulates the diffusion of elements, present at high concentration in the system, by affecting the status of collective actors, i.e. it simulates diffusion but is formally equivalent to a reaction step (albeit the reacting actors are present in adjacent cells rather than in the same one).

5. Notes on the implementation

5.1. The user interface

The model is set up by the user who defines G_I starting from a graphical representation of the bidimensional displayable space (possibly mapped on a digitized micrograph obtained experimentally from a living cell) and adding "external" cells where needed to grant the continuity of intersecting compartments.

The initial population of actors is generated and their initial location and set of status attributes are defined. This procedure can be performed, at will, either on the whole simulation area (independently for each compartment) or on selected regions or single cells. The parameters of most reactions can be set to default values or tuned at will.

When the simulation starts, a window displays the simulation area: a color code represents local calcium concentrations and the location of selected actors can be displayed. A second window yields full information on the actors momentarily present in a chosen cell, and lets the user modify their attributes. A control window lets the user suspend/restart the execution, select the monitored area and cell, change the number of actors and define the mode for data saving: time courses of relevant status attributes of specific actors, time courses of average Ca^{2+} concentration and transcompartmental fluxes, in the selected area, can be saved; images of local Ca^{2+} concentration can be saved at the desired time intervals.

5.2. The actors

Actors are present in the model to represent each relevant biological aspect. An actor does not necessarily correspond to a biological entity (see, below, collective actors and *shuttles*).

Each actor is a software object containing (i) a series of status attributes, (ii) a binary string which determines, by means of boolean operations, which other actors it can interact with, and (iii) all the procedures aimed at changing its status attributes as a consequence of the interaction with other actors (*reaction rules*). Most status fields regard the "functional state" of the actor, its and its behavior in *reaction* procedures. Diffusive behavior is affected by an *access* field which defines the compartments the actor can reside in and move to, and by a *direction* fields which determines whether the actor is fixed or, in case it is mobile, whether diffusion is random or along an assigned direction (this allows deterministic translocation of actors).

The main classes of actors are collective actors, carrier actors and mobile actors.

Collective actors – called *Free* and *Bound* – are fixed and represent calcium concentrations in the cell they are located in, by means of a binary string – the *concentration string* – where each set bit represents a fixed number of calcium atoms (1 in the cytoplasm, about 100 in compartments where calcium concentrations are higher). Ca^{2+} diffusion within each cell is simulated by shuffling *Free*'s binary string, Ca^{2+} binding/unbinding is simulated by transferring bits from suitable length substrings in *Free* to *Bound* and viceversa, and diffusion to neighboring cells is simulated by swapping fixed length substrings with *Free* actors in the neighboring cells. The occupancy of Ca^{2+} receptors and the degree of activation of Ca^{2+}-dependent processes are easily defined by reaction rules which look up the number of set bits in substrings from the concentration string of either *Free* or *Bound*.

Carrier actors simulate Ca^{2+} channels and pumps. They are located in cytoplasmic cells confining with other compartments, and they are in particular associated with a single arc in G_1 which connects two vertices of different kind. These actors simulate the opening of a pore or the active transport of calcium ions in a graded fashion, depending on their status. This allows for any kind of complex regulation of such mechanisms. The actual translocation of calcium ions is performed by *shuttle* actors, which are generated with the appropriate capacity and released into the cell; *shuttles* are loaded with Ca^{2+} by reacting with *Free* and mandatorily migrate along the appropriate arc of G_1 for a defined number of steps (usually 1 or 2, i.e. back and forth).

Mobile actors simulate any diffusible biological entity of interest – mostly regulatory molecules – which can be generated by specific reactions and in turn react with specific actors. They usually have a *life* field in their status which limits in time their persistence in the automaton. They may exhibit *access* restrictions to the different compartments and may diffuse either at random or along obligatory directions (see the description of the *shuttles* above).

5.3. *Algorithms and boolean operations on status strings*

A series of algorithms have been developed to appropriately simulate diffusion and complex dependence on Ca^{2+} concentration. Based on experimentally determined chemical–physical parameters and on theoretical computations, the probability of a particle leaving a fixed-volume cell during a fixed-duration time step in any direction is computed, thereby fixing the probability for a mobile actor to be displaced from its current location and the extent of the substring in *Free*'s concentration attribute to be swapped with neighboring *Frees*. Based on the knowledge of the affinity for Ca^{2+} of the binding proteins and of other Ca^{2+} receptors, the momentary occupancy of the latter is determined by boolean operations on substrings of *Bound*'s concentration attribute. Boolean operations on substrings of *Free*'s concentration attribute also reproduce polynomial approximations to high-order functions that describe the dependence of specific functions on Ca^{2+} concentration.

6. Conclusions

In this paper a formal definition of multilayered automata network and a derived definition of multilayered cellular space is given. Moreover, an example of modeling a complex biological process and simulating the dynamic regulation of Ca^{2+} distribution in the subcompartments of a living cell by means of multilayered cellular automata is described.

The multilayered cellular automaton, where diffusion rules operate on the vertices of lower-level graph (cells) and reaction rules operate on the corresponding set of second-level graphs, is further extended by introducing (i) the existence of different kinds of cells and restricted diffusion in the first-level graph, (ii) a set of updating rules that operate on the arcs of the first-level graph and (iii) particles that are *de facto* automata themselves. The latter makes it possible to reproduce underlying molecular processes whose simulation would require a finer spatial/temporal resolution.

The general features of this model permit to tackle the simulation of the particularly complex regulation of Ca^{2+} distribution in the living cell. Most problems here discussed also occur in their general features in other complex biological systems, where many different agents are present, several compartments must be considered, different degrees of detail are required in the simulation of different aspects, and high order functions describe the dependence of certain functions on regulatory factors. It appears therefore that the composite automaton here presented offers a general scheme which should make many apparently untractable biological processes amenable to computer simulation.

References

[1] S. Bandini, Hyper-cellular automata for the simulation of complex biological systems: a model for the immune system, in: D. Kirschner (Ed.), Special issue on "Advances in Mathematical Modeling of Biological Processes", Internat. J. Appl. Sci. Comput. 3 (1) (1996).

[2] S. Bandini, R. Fesce, G. Malagutti, G. Mauri, Multilayered cellular automata in modeling biological systems, Proc. 3rd European Congress on Systems Science, Rome, 1996, pp. 1121–1125.

[3] F. Celada, La Logica della Risposta Immunitaria, in: F. Celada (Ed.), La Nuova Immunologia, Le Scienze Editore, Milano, 1992.

[4] F. Celada, P.E. Seiden, A computer model of cellular interaction in the immune system, Immunology Today 13 (2) (1992).

[5] D. Dab, J.P. Boon, Approach to reaction diffusion problems, in: P. Manneville, N. Boccara, G.Y. Vichniac, R. Bidaux (Eds.), Cellular Automata and Modeling of Complex Physical Systems, Springer, Berlin, 1990.
[6] J.D. Farmer, N.H. Packard, The immune system, adaptation, and machine learning, Physica D 22 (1986).
[7] U. Frish, B. Hasslacher, Y. Pomeau, Lattice-gas automata for the Navier–Stokes equation, Phys. Rev. Lett. 56 (14) (1986).
[8] E. Goles, S. Martínez, Neural and Automata Networks: Dynamical Behavior and Applications, Kluwer Academic Publishers, Boston, 1990.
[9] T. Pozzan, R. Rizzuto, P. Volpe, J. Meldolesi, Molecular and cellular physiology of intracellular calcium stores, Physiol. Rev. 74 (1994).
[10] H.B. Sieburg, J.A. McCutchan, O.K. Clay, L. Caballero, J.J. Ostlund, Simulation of HIV infection in artificial immune systems, Physica D 45 (1990).
[11] S. Wolfram, Theory and Applications of Cellular Automata, World Scientific, Singapore, 1986.