Physica 10D (1984) 128–134
North-Holland, Amsterdam

# MODELLING CELLULAR AUTOMATA WITH PARTIAL DIFFERENTIAL EQUATIONS

Stephen OMOHUNDRO*
*Lawrence Berkeley Laboratory, University of California, Berkeley, CA 94720, USA*

A system of 10 coupled nonlinear partial differential equations is exhibited which simulates an arbitrary two-dimensional, nine-neighbor, square-lattice cellular automata. A number of theoretical implications of the result and the techniques used in its construction as well as possible practical consequences are discussed.

## 1. Introduction

Relatively recently there has been a flowering of powerful mathematical tools for analyzing systems of differential equations and iterated smooth maps [3]. Many of the elementary concepts of this smooth dynamical systems theory like the notions of orbits, fixed points, periodic orbits, basins of attraction, and reversibility have exact analogs in a discrete dynamical context. Analogs of more sophisticated notions are being discovered [4] but it is clear that it is useful to understand the relationship between continua and discretum better in this context.

One of the original motivations for the computer and one of its major present uses is numerically modelling continuum systems that are too complex to deal with analytically. Various numerical approximation schemes represent partial differential equations (P.D.E.'s) on a discrete spatial lattice with a discrete time step and with a finite precision at each lattice point, i.e. as a cellular automata. These approximations have done remarkably well, probably because the evolution of physical P.D.E.'s is very insensitive to small space and time scales (after all, the physical modelling which produces P.D.E.'s almost always breaks down at some scale).

* Research partially carried out while visiting the Center for Nonlinear Studies, Los Alamos National Laboratory.

Trying to understand the relation between discrete and continuum dynamics naturally leads one to the inverse question: Can one rigorously construct a cellular automata from a smooth underlying dynamical system? In his studies von Neumann posed but never answered this question. In his compilation of von Neumann's Theory of Self-Reproducing Automata [5], Burks explains: "The fourth model of self-reproduction which von Neumann considered was a continuous model. He planned to base this on a system of nonlinear partial differential equations of the type which govern diffusion processes in a fluid..... Von Neumann recognized that a system of simultaneous non-linear partial differential equations adequate to account for self-reproduction would be much more complex than the systems usually studied".

The construction of such a system is actually much simpler than von Neumann envisaged. We will exhibit ten coupled, two space- and one time-dependent, nonlinear partial differential equations which reliably simulate any two-dimensional, nine-neighbor, square-lattice cellular automata. A cellular automata state is represented by a smooth function with bumps at lattice sites whose heights are near integers which represent the cellular automata state (fig. 2a). For theoretical and practical reasons we have added a number of "bells and whistles" to the basic cellular automata simulator. These make the dynamics insensitive to noise and

0167-2789/84/$03.00 

give a certain structural stability to the flow. In particular any state close enough to a canonical "bump" state quickly relaxes toward it and any value near enough to an integer is a valid representation of that integer. The spacetime dependence may be eliminated by adding extra variables, just as time-dependent Hamiltonian systems may be made time independent by going to extended phase space. The time quantity would be a local oscillator and the space quantities would play the role of "morphogens" in determining position. Other dimensions, numbers of states, types of lattice, or neighborhoods are trivial extensions of the given construction. Because there is lots of unused space in representing a state, it appears that with a more complex design one could fit all the dynamics into a single P.D.E..

Before delving into the construction we will indicate some theoretical and practical reasons why it is of interest. It shows that relatively simple P.D.E.'s can be computation universal and support self-reproducing configurations, because there exist cellular automata with these properties (e.g. LIFE [1]). Ed Fredkin [2] has suggested that the unsolvability of the halting problem could be used to demonstrate that there does not exist a solution in closed form to these P.D.E.'s. The local dynamics is contracting in the regions of interest: two sufficiently close initial conditions contract to the same "bump" state. Thus the usual local measures of chaos like Liapunov exponents fail to detect the fact that solutions evolve with arbitrary unpredictability globally. For example we may construct an initial condition whose asymptotic state is zero everywhere if Fermat's last theorem is true and has a single bump if it is false.

The construction itself gives us insight into the mechanisms for creating separation of levels. Firstly, reliability and insensitivity to noise require that the flow be attracted to legal states. This dissipative or irreversible aspect of the underlying dynamics is completely independent of the reversibility or irreversibility of the simulated automata. Secondly, insensitivity of the output to the exact input requires nonlinearity (weak dependence of output levels on input levels in some regions, strong in others, fig. 1.). This nonlinearity is used in several places in our construction and also occurs in any digital circuits made from analog components. Thirdly, we needed to employ multiple time scales. The relaxation mechanisms had to be fast compared to the state changing mechanisms. This same point arose in Otto Rössler's discussion of the evolution of a biochemical switch [6]. Lastly, these elements are combined to give pattern formation in a continuum system in the manner that catastrophe theory suggests discrete biological differentiation and pattern formation occur over an underlying cellular continua.

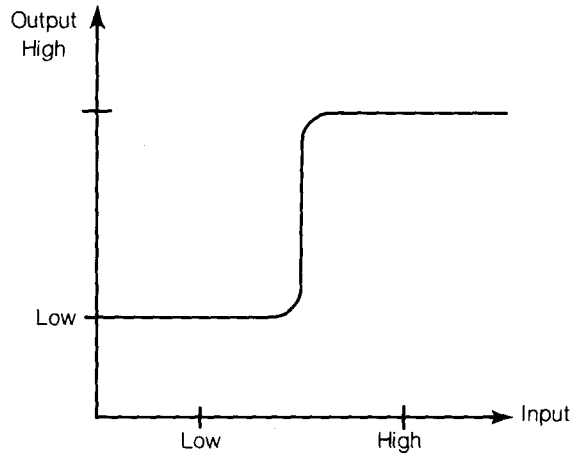


Fig. 1. The typical nonlinearity for an analog implementation of a digital design.

There are also some potentially practical aspects to this construction. Toffoli [7] has given a nice discussion of the engineering and cost advantages of constructing computers as large arrays of identical components, likening their construction to printing large sheets of postage stamps. There is no reason to stop there, one can build very inexpensive devices if one has no local structure at all. This structureless approach to circuits is actually being implemented currently to build inexpensive high-density silicon memories and displays [8], using the strong dopant dependence of the frequency of excitation of an electron to the conduction band in semiconductors. One utilizes continuous dopant

gradients and selects a particular region of a chip by the frequency of a signal instead of selecting particular access lines which have been defined via photolithography. Such an approach is very insensitive to nonuniformities in the dopant gradient and is inexpensive to implement.

A similar philosophy could conceivably be used to construct a structureless computer based on these P.D.E.'s. If one could find chemicals whose reaction–diffusion equations were these P.D.E.'s, or a set with the same capabilities, then pouring them together into a petri-dish would yield an "instant computer".

We will now describe the P.D.E.'s, often leaving constants to be chosen large enough. Choosing explicit values for these and rigorously proving the stated properties is easily done but not very enlightening. We define the 10 P.D.E. *variables* $N(x, y, t)$, $F(x, y, t)$, $S_1(x, y, t), \ldots, S_8(x, y, t)$, the 8 *bump functions* used in the construction $b_1(x), \ldots, b_8(x)$, the 4 *time steps* and their effects, the *hysteresis function* $H(x, c)$, the method of *forcing relaxation* to a bump, the *transition function* $T(N, S_1, \ldots, S_8)$, the method of *shifting bumps* around and finally exhibit *the P.D.E.*

## 2. The variables

The variable which holds the current state of the automata is $N(x, y, t)$ (for "Now"). As a function of $x$ and $y$ it is a bunch of $C^\infty$ bumps centered on the integer lattice $Z \times Z$ with approximately integral height at the center which defines which of the finite number of cell states is represented (fig. 2a). For convenience we take the bumps to be 0.2 wide and constant outside of 0.01 from the transition boundaries. The height error should be less than 0.1 (fig. 2b). The variables $S_1(x, y, t), \ldots, S_8(x, y, t)$ hold the same information as $N(x, y, t)$ in the same format but shifted in $(x, y)$ by the amounts: $(-1, 0)$, $(-1, 1)$, $(0, 1)$, $(1, 1)$, $(1, 0)$, $(1, -1)$, $(0, -1)$, $(-1, -1)$ (fig. 2c). The variable $F(x, y, t)$ (for "Future") will hold the next state of the automata in the same format as $N(x, y, t)$ but with bumps that are .4 across.

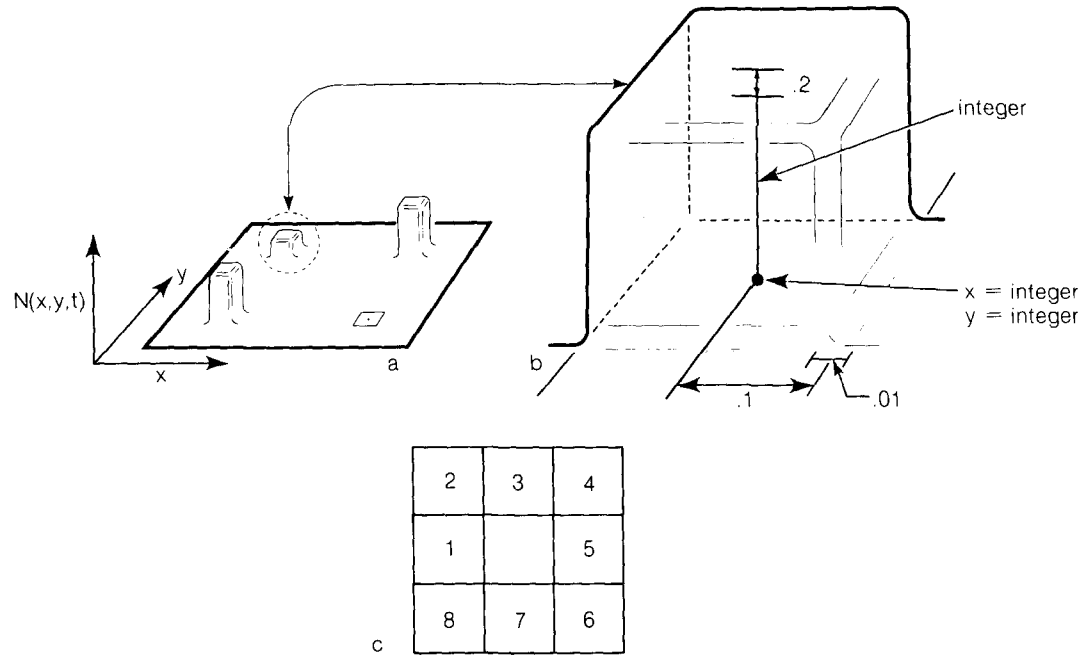


Fig. 2. a) the graph of a bump state; b) the specifications of a single bump; c) the eight nearest neighbors of a cell.

## 3. Bump functions

The basic smooth construction tools are $C^\infty$ bump functions (sometimes called mollifiers) which can be constant in some regions and yet smoothly vary in others. The bumps we shall need are defined in fig. 3. $b_1(x)$ is the basic bump: zero for $|x| \geq 0.01$, nonzero for $|x| < 0.01$. $b_2(x)$ is 0.4 wide and has a peak value of 1. $b_3(x)$ is a smooth step that is zero for $x < -0.01$ and 1 for $x > 0.01$. $b_4(x)$ is a 0.2 wide bump that is 1 for $|x| < 0.03$ and 0 for $|x| > 0.11$. $b_5(x)$ is the same but 0.4 wide. $b_6(x)$ is a bunch of 0.2 wide $b_4$'s over integers. $b_7(x)$ is 0.4 wide $b_5$'s over integers. $b_8(x)$ is a bunch of $b_2$'s with no flat top over integers.

## 4. Time steps

The P.D.E. cycles once every unit of time. This unit is broken into four intervals of length 0.2 as in fig. 4 during which different operations take

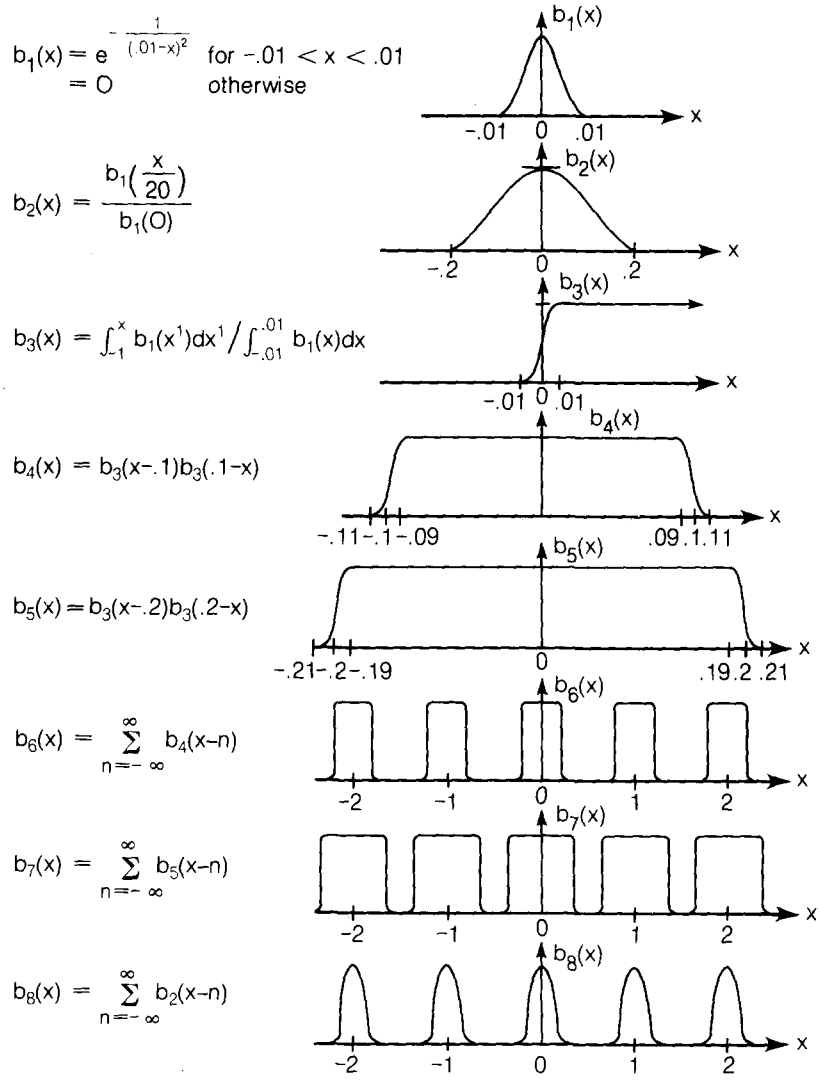


Fig. 3. The definitions and graphs of the 8 basic bump functions used in the construction.

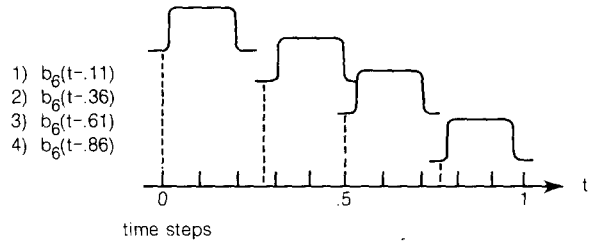


Fig. 4. The four time steps.

place. In the first time step $b_6(t-0.11)$, the now variable $N$ and the shifted variables $S$ are cleared and the state in $F$ is smoothed out. In the second time step $b_6(t-0.36)$, $F$ is loaded into $N$ and the $S$'s truncating with a 0.2 wide bump. In the third time step $b_6(t-0.61)$, $F$ is cleared and the state of the $S$'s is shifted in each of the eight directions. In the fourth time step $b_6(t-0.86)$, we load $F$ with the new value computed from $N$ and the $S$'s with hysteresis.

## 5. Hysteresis

We now define the mechanism which pushes near integer values closer to being integral. For a four state automata we could define a function $H(x, c)$ where $x$ is thought of as the state and $c$ as a control parameter. We build $H(x, c)$ out of bumps to be a $C^\infty$ function with critical points given in fig. 5a: solid lines are minima, dotted are maxima. As a function of $x$, $H$ will look like fig. 5b. With the dynamics

$$\dot{x} = c_1 \frac{\partial}{\partial x} H(x, c)$$

and large enough $c_1$ we have a device with control knob $c$ that behaves as follows: if $c$ is close to $-1$ (i.e. $-1.1 < c < -0.9$) then $x$ is "reset" to 0 (i.e. $-0.1 < x < 0.1$) within 0.01 time units, if $c$ is set close to $n = 0, 1, 2, 3$ when $x$ is near zero then $x$ is "set" to $n$, and otherwise $x$ remembers its state.

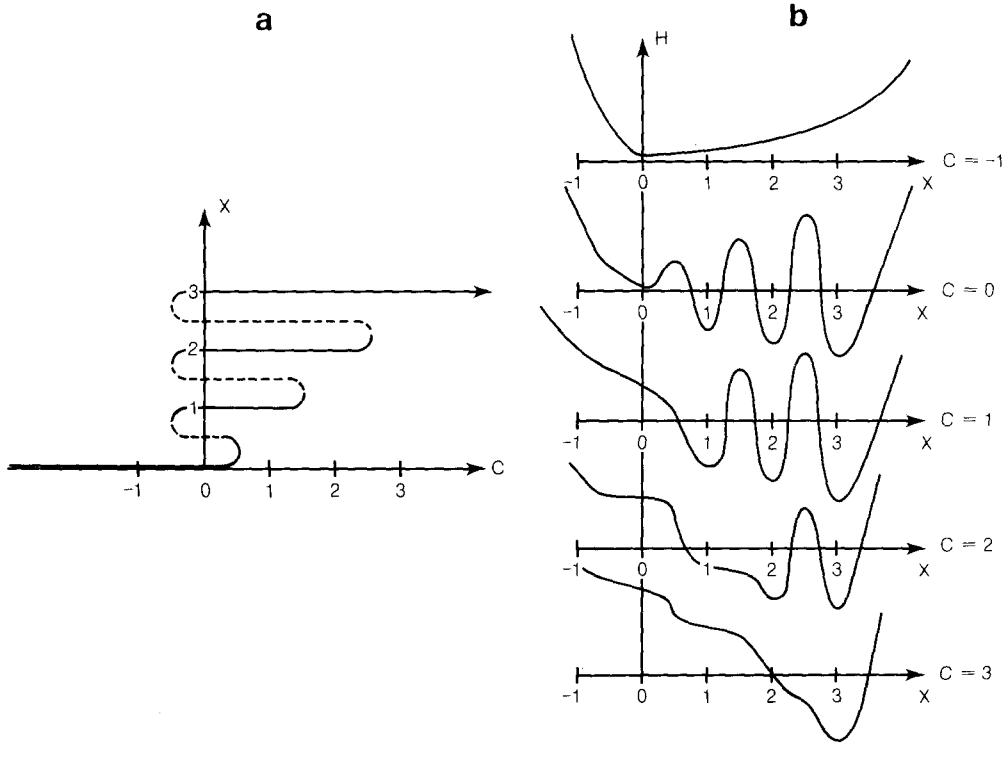


Fig. 5. a) The critical points of $H(x, c)$, solid lines are minima, dotted are maxima; b) Graphs of $H$ versus $x$ at various values of $c$.

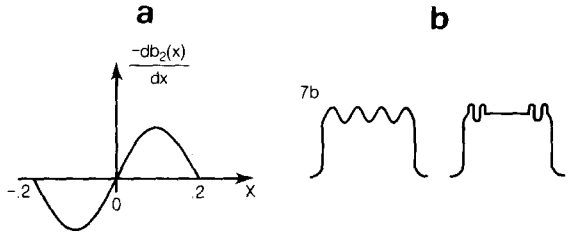


Fig. 6. a) The graph of $-db_2(x)/dx$; b) the effect of smoothing on a bump.

## 6. Forcing relaxation to a bump

As part of our design we would like the P.D.E. to accept initial states which are noisy and do not follow bump protocol and quickly convert them to cleaner states. There are many ways to do this but most suffer from lack to centralized control. Our scheme eventually slaves the whole bump to the value at the center. The technique is to push the bad information out to the wings of the bump in $F$ (which uses the wider 0.4 bumps) and to truncate it away when loading $F$'s state into $N$ and the $S$'s. The function $-db_2(x)/dx$ is shown in fig. 6a. The effect of the evolution: $\dot{F}(x) = -c_2(db_2/dx)\partial F/\partial x$ is to push $F$'s value near zero outward while keeping the nonzero portion within 0.2 of zero (see fig. 6b). In two dimensions this evolution is given by

$$\dot{F}(x, y, t) = -c_2\left(\frac{db_2(x)}{dx}\frac{\partial F}{\partial x} + \frac{db_2(y)}{dy}\frac{\partial F}{\partial y}\right).$$

We choose $c_2$ large enough that this blows up a small neighborhood of zero to the whole of a 0.2 bump in the allotted time.

## 7. Transition function

The function $T$ of nine variables holds the transition rules for the automata. We construct it from bump functions so that $T(N, S_1, \ldots, S_8)$ is the integer representing the successor state to the state represented by the integer $N$ and the eight neighbors $S_1, \ldots, S_8$.

## 8. Shifting the bumps around

We now describe the means for getting the shifted versions $S_1, \ldots, S_8$ of the state $N$. The idea is to turn on a bunch of first order wave equations for the right amount of time to shift by the right amount. The equation

$$\frac{\partial f}{\partial t} = b_4(t)\frac{\partial f}{\partial x}$$

has the solution: $f(x, t) = f(x + a(t))$ where $a(t) = \int_{-0.11}^{t} b_4(t)\,dt$. After $t = 0.11$ the effect is to just shift $f$ by a distance $a(0.11)$. Define $a_1 = \int_{-0.11}^{0.11} b_4(t)\,dt$. Then $\partial f/\partial t = (b_4(t)/a_1)(\partial f/\partial x)$ shifts $f$ one space to the left. The equations for the other 7 directions are similar.

## 9. The P.D.E.

Using the pieces we have described in the previous sections we now exhibit the full P.D.E.

$$\dot{F}(x, y, t) = \Big\{ -b_6(t - 0.11)c_2\Big(\frac{db_8(x)}{dx}\frac{\partial F}{\partial x} + \frac{db_8(y)}{dy}\frac{\partial F}{\partial y}\Big) - b_6(t - 0.61)c_3 F + b_6(t - 0.86)H(F, T(N, S_1, \ldots, S_8))\Big\},$$

$$\dot{N}(x, y, t) = -b_6(t - 0.11)c_3 N + b_6(t - 0.36)c_3[b_6(x)b_6(y)F - N],$$

$$\dot{S}_1(x, y, t) = \Big\{ -b_6(t - 0.11)c_3 S_1 + b_6(t - 0.36)c_3[b_6(x)b_6(y)F - S_1] + b_6(t - 0.61)\frac{1}{a_1}\frac{\partial S_1}{\partial x}\Big\},$$

and similarly for $S_2, \ldots, S_8$.

The first term in $\dot{F}$ smooths $F$ in the first time step, the second term clears $F$ in the third time step

and the last term sets $F$ to the new state with hysteresis in the fourth time step. The first term in $\dot{N}$ clears $N$ in the first time step and the second term sets $N$ to the values in $F$ with truncation to 0.2 bumps in the second time step. The first term in $\dot{S}_1$ clears $S_1$ in the first time step, the second term sets $S_1$ to $F$ with truncation in the second time step and the last term shifts $S_1$ leftward in the third time step. It is the last term in $\dot{S}_1$ that is altered in $S_2, \ldots, S_8$ to cause shifting in the different directions.

## 10. Conclusions

We have demonstrated the existence of a set of P.D.E.'s which simulates cellular automata and thus may exhibit computation universality and self-reproduction. In some ways these P.D.E.'s are somewhat artificial in that they use rather intricate $C^\infty$ functions in a manner quite analogous to the design of a digital simulator. One would really like to use the continuum tools of functional analysis to analyze the behavior of cellular automata like one uses Koopmanism [9] to analyze finite dimensional nonlinear problems via equivalent infinite dimensional linear ones. For this type of program one would prefer a much more "natural" system of P.D.E.'s defined by very simple analytic functions. One wonders if some of the standard nonlinear P.D.E.'s with regimes exhibiting very complex "turbulent" behavior like the Navier–Stokes equations might be in some sense computation universal [10]. Other work relating continuous and discrete realizations may be found in [11] and [12].

## Acknowledgements

I would like to thank Jim Crutchfield, Norm Packard and Doyne Farmer for many stimulating conversations and the Center for Nonlinear Studies at Los Alamos and Lawrence Berkeley Laboratory for facilities and support.

## References

[1] E. Berlekamp, J. Conway and R. Guy, Winning Ways (Academic Press, London, 1982), pp. 817–849.
[2] E. Fredkin, private communication.
[3] V.I. Arnold, Geometrical Methods in the Theory of Ordinary Differential Equations (Springer, New York, 1983).
[4] D. Lind, Physica 10D (1984) 36 (these proceedings).
[5] J. von Neumann, Theory of Self-Reproducing Automata, edited and completed by A.W. Burks, Urbana, 1966, p. 97.
[6] O. Rössler, Chemical automata in homogeneous and reaction-diffusion kinetics, in: Springer Lecture Notes in Biomathematics 4 (1974) 399–418.
[7] T. Toffoli, Physica 10D (1984) 195 (these proceedings).
[8] D. McGoveran, private communication.
[9] I.P. Cornfeld, S.V. Fomin and Ya.G. Sinai, Ergodic Theory (Springer, New York, 1982).
[10] M. Conrad and O. Rössler, Example of a system which is computation universal but not effectively programmable, Bull. Math. Biol. 44 (1982).
[11] P. Channell, preprint.
[12] T. Toffoli, Bicontinuous Extensions of Invertible Combinatorial Functions, Math. Systems Theory 14 (1981) 13–23.