International Journal of Modern Physics C
Vol. 18, No. 5 (2007) 833–848

World Scientific
www.worldscientific.com

MODELING LINEAR DYNAMICAL SYSTEMS BY CONTINUOUS-VALUED CELLULAR AUTOMATA

JUAN CARLOS SECK TUOH MORA*, MANUEL GONZALEZ HERNANDEZ†, NORBERTO HERNANDEZ ROMERO‡ and AARON RODRIGUEZ TREJO

Centro de Investigación Avanzada en Ingeniería Industrial
Universidad Autónoma del Estado de Hidalgo
Carr. Pachuca-Tulancingo Km 4.5, Pachuca Hidalgo 42080, México
**jseck@uaeh.reduaeh.mx*
†*mghdez@uaeh.reduaeh.mx*
‡*nhromero@uaeh.reduaeh.mx*

SERGIO V. CHAPA VERGARA

Departamento de Ingeniería Eléctrica, Sección Computación
Centro de Investigación y de Estudios Avanzados, Instituto Politécnico Nacional
Av. IPN 2508, Col. San Pedro Ticomán, México D.F. 05760, México
schapa@cs.cinvestav.mx

Received 14 February 2006
Accepted 26 July 2006

This paper exposes a procedure for modeling and solving linear systems using continuous-valued cellular automata. The original part of this work consists on showing how the cells in the automaton may contain both real values and operators for carrying out numerical calculations and solve a desired problem. In this sense the automaton acts as a program, where data and operators are mixed in the evolution space for obtaining the correct calculations. As an example, Euler's integration method is implemented in the configuration space in order to achieve an approximated solution for a dynamical system. Three examples showing linear behaviors are presented.

Keywords: Cellular automata; block diagrams; differential equations.

PACS Nos.: 87.18.Dh, 87.18.Sn, 07.05.Tp.

1. Introduction

The study of dynamical systems is relevant in every theoretical and applied science, where differential equations are the common framework used in this task. One problem is that their representation may have a complicated analytic solution in almost all the cases, even if serious simplifications are taken. Thus, the research in numerical analysis has developed a huge set of results widely used thanks to the availability of cheaper and more powerful computational equipment. The rising of these resources has been applied as well for investigating new paradigms for

studying dynamical systems, taking its atomic parts and characterizing their local interactions.

Cellular automata represent one of the simplest approaches using this idea; where space and time are discrete and local interactions are not complicated,[1] providing a model which can be smoothly executed in a computer and offering an easier analysis. One branch in this theory is to reproduce calculations mixing data and operators in the evolution space. This interaction has been inspected for effectuating unconventional computing;[2–5] a relevant result by Cook[6] and Wolfram,[7] use an automaton with two states for reproduce an universal cyclic tag system.

The previous references show that cellular automata are an active field for searching alternative computation prototypes. Nevertheless the existence of other tools with deeper developments for realizing efficient computations, as parallel paradigms based on shared or distributed memory;[8,9] the interest in using cellular automata is explained due to the simplicity of their elements. Although such a feature implies to employ a large number of cells, the same facilitates their implementation, where the current computers may manage millions of cells.

Based on the previous ideas, our goal is to show how numerical values and simple logic and arithmetic operators are used in the cells of a cellular automaton for performing procedures to solve dynamical systems. In this paper, Euler's integration method and classic linear systems are taken since they have well-known results, thus it can be easily understood how the numerical algorithm is programed and verify the correctness of the process; in this way the intention of the manuscript is more academic than practical. The relevance of the work is to show how a discrete system ruled by local interactions, is able to modeling a physical system without applying any control or global variables for achieving synchronization or data validation; the expected coordination is obtained by the set of local operations.

For achieving this objective, the differential equation representing the desired system is taken, then we shall use a graphic representation of their solution by block diagrams,[10] which are a common tool for defining the set of numerical operations to figure out the problem. The local interaction among their components is characterized and performed by means of cells, so block diagrams are the bridge between differential equations and cellular automata. Our definition of cellular automata has real continuous values for the cells, right-sided evolutions in the neighborhoods and non-uniform neighborhood size. While this interpretation is different from the classical one, their characteristics have been used before by other authors, for instance[11–13] apply real values in the states of the automaton for modeling dynamical systems;[14,15] take cellular automata with evolutions placed in one side of the neighborhoods for examining topological and dynamical properties and[16] treats cellular automata with non-uniform neighborhood size for simulating flexible manufacturing systems.

The paper is organized as follows: Sec. 2 presents the characterization of linear systems by differential equations and exposes block diagrams for showing a graphic representation of their solutions. Section 3 describes the construction of cellular

automata and depicts how the cells are able to model the elements of a block diagram. Section 4 gives examples of linear systems resolved by cellular automata and Sec. 5 provides the concluding remarks of the document.

2. Ordinary Differential Equations and Block Diagrams

To understand the dynamics of a real electrical or mechanical system, classical laws of physics are utilized to create a mathematical model. A good approximation of its behavior is achieved taking a system of concentrated parameters, conforming a model of ordinary differential equations. An ordinary, linear differential equation of order n with constant coefficients is described as:

$$a_0 y^{(n)} + a_1 y^{(n-1)} + a_2 y^{(n-2)} + \cdots + a_n y = g \tag{1}$$

where $\{a_0 \neq 0, a_1, a_2, \ldots, a_n\} \subset \mathbb{R}$, $g : I \to \mathbb{R}$ and $y : J \to \mathbb{R}$ for I, $J \subseteq \mathbb{R}$. On dividing Eq. (1) by a_0 we have that:

$$y^{(n)} + a_1 y^{(n-1)} + a_2 y^{(n-2)} + \cdots + a_n y = \frac{g}{a_0} = f \tag{2}$$

Let us take $L(y) = y^{(n)} + a_1 y^{(n-1)} + a_2 y^{(n-2)} + \cdots + a_n y$; so Eq. (2) becomes in $L(y) = f$. For all $x \in I$, if $f(x) = 0$ then $L(y) = 0$ is a homogeneous equation, in other case $L(y) \neq 0$ is non-homogeneous. A solution of $L(y) = b(x)$ is a function $\phi : J \to \mathbb{R}$ having n derivatives such that $L(\phi) = f$; if f is continuous on I, it is possible to derive all the solutions of $L(y) = f$.[17,18] Finding the solutions of the homogeneous equation is an algebraic problem consisting on calculating the roots of a polynomial; the solutions of the non-homogeneous equation are generated using those of the corresponding homogeneous case and integrating taking into account f. For $n = 2$, a well-known result[17,18] is that for constant values a_1 and a_2 and:

$$L(y) = y'' + a_1 y' + a_2 y = 0\,, \tag{3}$$

if r_1, r_2 are distinct roots of the characteristic polynomial $p(r) = r^2 + a_1 r + a_2$, then ϕ_1, ϕ_2 defined by:

$$\phi_1(x) = e^{r_1 x}\,, \qquad \phi_2 = e^{r_2 x} \tag{4}$$

are solutions of Eq. (3). If p has a repeated root r, then ϕ_1, ϕ_2 defined by:

$$\phi_1 = e^{rx}\,, \qquad \phi_2 = x e^{rx} \tag{5}$$

are solutions of Eq. (3). Furthermore, the linear combination with constants coefficients given by:

$$\phi = c_1 \phi_1 + c_2 \phi_2 \tag{6}$$

is a solution as well. For the non-homogeneous case:

$$L(y) = y'' + a_1 y' + a_2 y \neq 0 \tag{7}$$

another result is that if f is continuous in I, Ψ_p is a particular solution of Eq. (7), and Ψ is any other solution, then:

$$L(\Psi - \Psi_p) = L(\Psi) - L(\Psi_p) = 0 \tag{8}$$

showing that $\Psi - \Psi_p$ is a solution of Eq. (3).[17,18] Therefore if ϕ_1, ϕ_2 are linearly independent solutions of Eq. (3), there are unique constants c_1, c_2 such that:

$$\Psi - \Psi_p = c_1\phi_1 + c_2\phi_2\,. \tag{9}$$

Thus, every solution of Eq. (7) can be written as:

$$\Psi = \Psi_p + c_1\phi_1 + c_2\phi_2 \tag{10}$$

and the problem of finding all the solutions of Eq. (7) is solved by calculating a particular Ψ_p and two linearly independent solutions ϕ_1, ϕ_2 for Eq. (3). To yield a particular solution of Eq. (7) we use the variation of constants to get Ψ_p:

$$\Psi_p(x) = \phi_1 \int \frac{-\phi_2 f(x)}{W(\phi_1,\phi_2)}dx + \phi_2 \int \frac{\phi_1 f(x)}{W(\phi_1,\phi_2)}dx\,, \tag{11}$$

where $W(\phi_1,\phi_2)$ is the Wronskian of ϕ_1 and ϕ_2. Linear systems represent a common place for introducing the application of numerical algorithms, there exists a set of widely-used algorithms used for solving differential equations, one standard tool is Euler's method. Most of the commercial software implementing these procedures allows as well a graphic representation of their application by means of block diagrams. These diagrams are inspired on the work developed in 1930s for constructing an analog computer to resolve differential equations following the terms of an analog language. A block diagram is formed by simple elements, each one applying a numerical operation, these elements are connected by directed edges describing a data flow actualized as it crosses the different parts of the diagram. Eq. (2) can be reformulated as follows:

$$y^{(n)} = f - a_1 y^{(n-1)} - \cdots - a_{n-1}y' - a_n y\,. \tag{12}$$

Equation (12) is graphically represented by a symbol describing the sum of $n+1$ signals in Fig. 1. To obtain the solution of the differential equation it is necessary to solve the signals in function of $y^{(n)}$, these can be resolved integrating each one

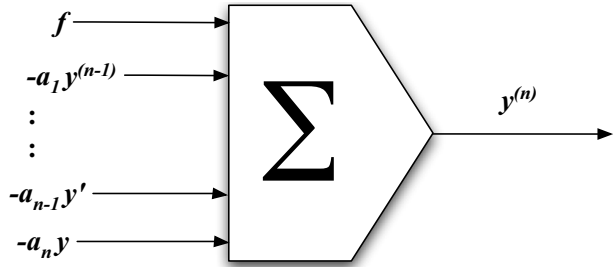


Fig. 1. Sum of the signals giving $y^{(n)}$.

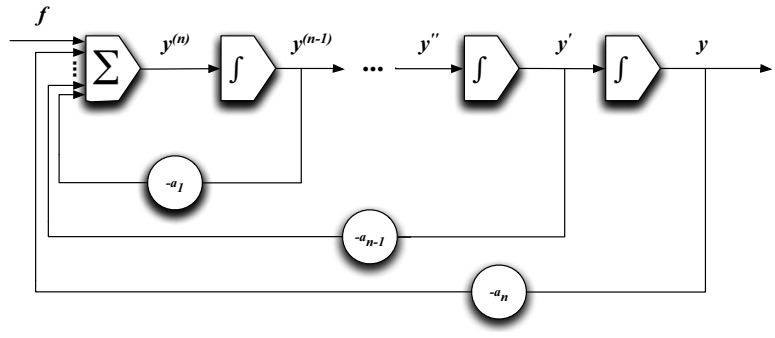


Fig. 2. Numerical solution of Eq. (12) in a block diagram.

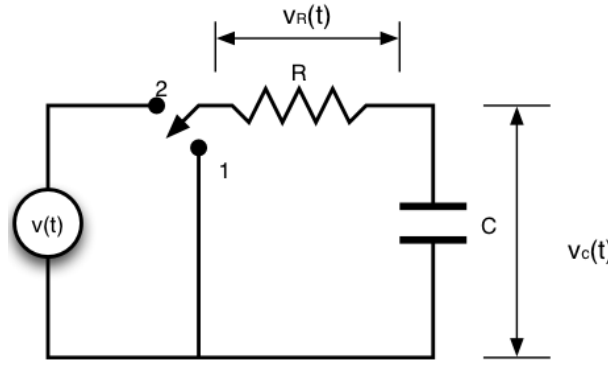


Fig. 3. Serial RC circuit.

of them, so we can add symbols to Fig. 2 presenting serial-connected integrators and multiplicative constants for calculating the solution function of Eq. (12), and returning the outputs of these blocks into the inputs of the sum block to get the complete solution in a representative period of time.

Thus every element of the diagram produces a small step of the computation; these blocks may be implemented in a computer language selecting a convenient method and integration step in the case of the integrator block. The interconnection of the blocks provides the corresponding flow of signals for getting the numerical solution. Physical systems numerically modeled and generally simulated are RC circuits, mass-spring and mass-spring-damper systems.

2.1. *RC Circuit*

A serial electric RC circuit is depicted in Fig. 3. $v(t)$ is the excitation voltage, $v_R(t)$ is the voltage lost in the resistance and $v_c(t)$ is the voltage lost in the capacitance. Initially, the interrupter is in position 1 yielding that all the other initial conditions are in 0. When the switch pass to position 2 in $t = 0$ the circuit is excited with

source $v(t)$. If the circuit has parameters $C = 1000\ \mu F$, $R = 250\Omega$ and $v(t)$ is a unitary step function, then the model to predict the behavior of the voltage $v_c(t)$ in the capacitor terminals is determined by:

$$\frac{d}{dt}v_c(t) + \frac{1}{RC}v_c(t) = \frac{1}{RC}v(t) \tag{13}$$

and the analytic solution is defined as:

$$v_c(t) = 1 - e^{-4t}\,. \tag{14}$$

The block diagram describing the solution of the circuit is showed in Fig. 4.

2.2. *Mass-spring system*

A mass-spring system is reflected in Fig. 5. Where m is the mass, k is the elasticity constant of the spring, $f(t)$ is the force applied to the mass and $y(t)$ is the mass shift. Initially, the system is equilibrated and the initial conditions are 0, if the system is excited by a unitary step function with parameters $m = 1$ kg and $k = 1N/m$, then the model is determined by:

$$m\frac{d^2}{dt^2}y(t) + ky(t) = f(t) \tag{15}$$

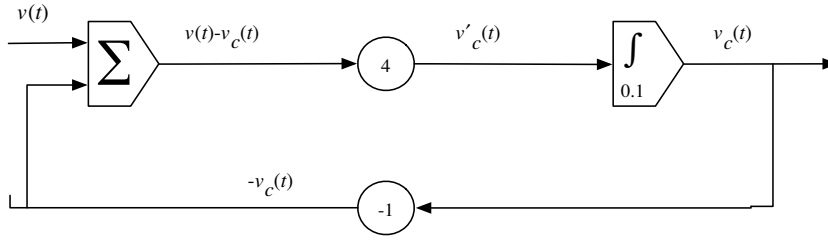


Fig. 4. Block diagram for a serial RC circuit.

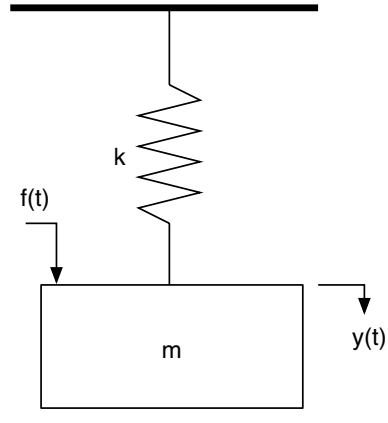


Fig. 5. Mass-spring system.

and the analytic solution showing a simple harmonic oscillation of the mass is:

$$y(t) = 1 - \cos(t)\,. \tag{16}$$

The block diagram describing the solution of the system is showed in Fig. 6, we have an integration step of 0.01 for the numerical calculation.

2.3. *Mass-spring-damper system*

A mass-spring-damper system is described in Fig. 7. The system is almost identical to the previous one, only there is an extra component b meaning the friction coefficient of the damp. For an equilibrated system all the initial conditions are 0, if the system is excited by a unitary step function with parameters $m = 0.01$ kg, $b = 0.02Ns/m$ and $k = 1N/m$, the model is defined in Eq. (17).

$$m\frac{d^2}{dt^2}y(t) + b\frac{d}{dt}y(t) + ky(t) = f(t) \tag{17}$$

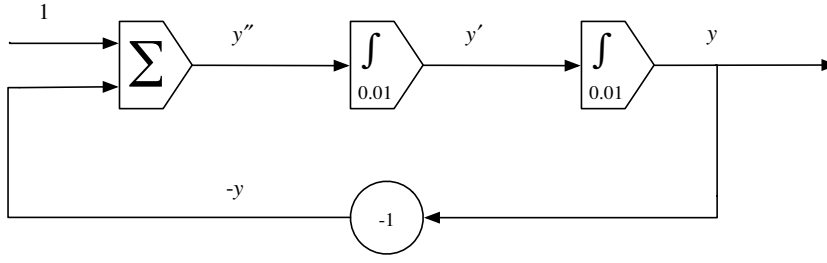


Fig. 6. Block diagram for a mass-spring system.

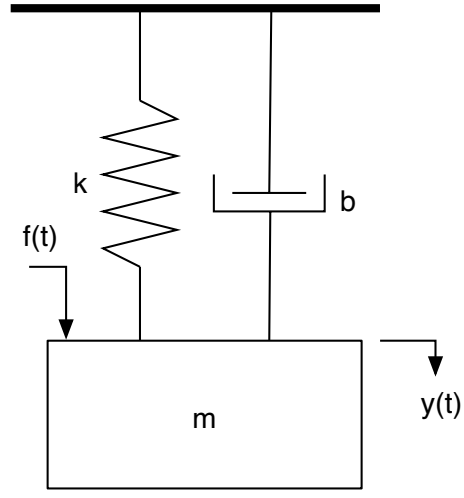


Fig. 7. Mass-spring-damper system.

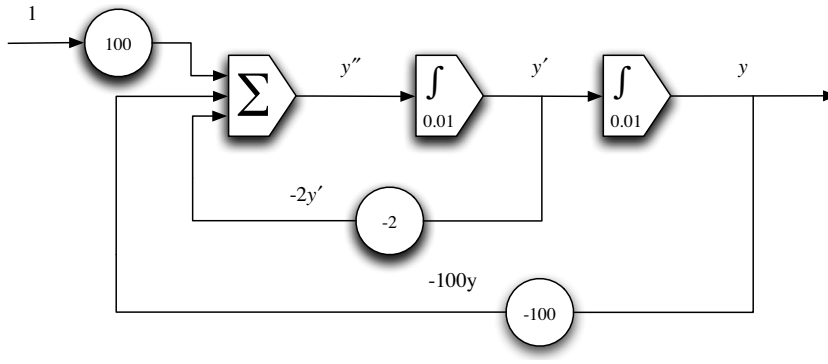


Fig. 8. Block diagram for a damped mass-spring system.

and the analytic solution showing an underdamped oscillation of the mass is:

$$y(t) = 1 - e^{-t}\cos(9.9498t) - \frac{1}{9.9498}e^{-t}\sin(9.9498t)\,. \tag{18}$$

The block diagram describing the dynamical solution of the system is in Fig. 8 with an integration step of 0.01.

For every block diagram, the operation of each element depends on the previous one, hence the interplay of the parts defines the global solution of the problem. This is analogous for a cellular automaton, where the local interactions defines the global behavior of the system. The following sections explains how to simulate the essential objects of a block diagram by means of states in a cellular automaton.

3. Cellular Automata Modeling Dynamical Systems

Normally, cellular automata have been used for representing dynamical systems finding an evolution rule which reflects the local behavior of the phenomenon.[11] Following another tendency,[2,19] we shall use one-dimensional cellular automata for establishing numerical solutions mixing data and operators in the configurations.

A cellular automaton consists of a set of states K; the set of finite sequences of states is described by K^*. For every $w \in K^*$, let n_w be the number of states in w. We shall index every state of w from left to right, starting from position 0, thus w_i is the state at position i mod n_w and $w_{[i\cdots j]}$ is the block of states from i to j. The automaton has an initial condition or configuration $c^0 \in K^*$; the superscript indicates the current time and will be omitted when it is understood. There is a set $\Phi \subset K^*$ of neighborhoods where for every $w \in \Phi$, there is a mapping or evolution rule $\varphi(w) = a \in K$ executing a set of logical and arithmetic operators when the neighborhood appears in the current configuration. If $c^t_{[i\cdots j]} = w \in \Phi$ then $c^{t+1}_j = \varphi(w)$; otherwise $c^{t+1}_j = c^t_j$.

Thus φ yields a new configuration c^{t+1}; periodic boundary conditions are applied to have complete neighborhoods for all the states in the configuration. We are using neighborhoods with right-sided evolutions for producing a shift of the information

from left to right during the dynamics of the automaton, holding the same behavior that in a block diagram. For linear systems, block diagrams have three operators: sums, integrators and products by constant values. They are executed using real values, so the automaton will have real states described by $v \in \mathbb{R}$ and the previous operations shall be implemented by action states which will have three properties:

val: The numerical value for realizing an operation; it can be real or integer.
min: The minimum value for controlling the periodic execution of an action.
max: The maximum value used with the minimum one for changing the numerical value of another state, mostly applied for passing information.

Table 1 depicts the action states completing the operations of a block diagram; when a state is not using a particular property, it will be indicated by NA (Non-applicable). With these states, Eq. (19) performs an integration using Euler's method.

$$v_1 P v_2 G v_3 \,. \tag{19}$$

Table 1. Action states for simulating a block diagram.

Operation	State	val	min	max	Neighborhood	Evolution
Constant	$k \in \mathbb{Z}$	$k \in \mathbb{Z}$	NA	NA	Several forms	The state preserves its value during the whole evolution
Addition	A	NA	NA	NA	$v_1 v_2 \mathrm{A} v_3$	$v_3 = v_1 + v_2$
Copy	C	NA	NA	NA	$v_1 \underbrace{\cdots\cdots\cdots}_{n \text{ states}} \mathrm{C} v_2$	$v_2 = v_1$
Sum	S	NA	NA	NA	$v_1 \underbrace{\cdots}_{n_1} v_2 \cdots v_j \underbrace{\cdots}_{n_j} \mathrm{S} v_{j+1}$	$v_{j+1} = \sum_{i=1}^{j} v_i$
Product	P	$a \in \mathbb{R}$	NA	NA	$v_1 \mathrm{P} v_2$	$v_2 = v_1 * a$

Sum gate	G	$0-1$	$i \in \mathbb{N}$	$j \in \mathbb{N}$	$v_1 \mathrm{G} v_2$	if (val = 1)
						$v_2 = v_2 + v_1$
						val = 0
						min = min +1
						else
						if(min < max)
						min = min +1
						else
						min = 1,val = 1

Equal gate	E	$0-1$	$i \in \mathbb{N}$	$j \in \mathbb{N}$	$v_1 \mathrm{E} v_2$	if(val = 1)
						$v_2 = v_1$
						val = 0
						min = min +1
						else
						if(min < max)
						min = min +1
						else
						min = 1,val = 1

In Eq. (19), state v_2 keeps the product of v_1 by an integration step P, this result is accumulated in v_3 by G after a predefined number of iterations; accumulating the area of a small rectangle. With the action states, the next section utilizes distinct cellular automata for solving the linear systems exposed in Sec. 2.

4. Examples

4.1. *RC circuit*

In order to modeling a RC circuit, we shall use a cellular automaton with $K = \mathbb{R} \cup \{A, P1, P2, G\}$; the neighborhoods are defined in Table 2 using instances of the states in Table 1 with $v = 0$ and $k = 1$.

With the neighborhoods in Table 2 we define the initial configuration c^0 in Fig. 9, it also shows how the parts of the block diagram in Fig. 4 are put into effect by the states of c^0.

Figure 10 presents 21 iterations of the automaton, in particular cell c_6 contains the numerical solution of the differential equation. The evolution shows how the information goes from left to right and the feedback of the system is given after four time steps. In the initial configuration, cell c_5 is a sum-gate state; for obtaining a right computation, the initial parameters in c_5^0 are: val $= 0$, min $= 3$ and max $= 4$; thus c_5^1 holds that min $= 4$ and c_5^2 has min $= 1$ and val $= 1$, letting pass the information in the following time step. The same behavior is repeated every four steps.

We can display the evolution of the system each fourth iteration to present a more extended calculation, this process is in Fig. 11 which also graphs the dynamical

Table 2. Neighborhoods for modeling a RC circuit.

Φ	φ
$v_1 k \mathrm{A} v_2$	$v_2 = k + v_1$
$v_1 \mathrm{P1} v_2$	$v_2 = 0.4 * v_1$
$v_1 \mathrm{P2} v_2$	$v_2 = -1 * v_1$
$v_1 \mathrm{G} v_2$	$v2 = v2 + v1$ iff $G = 1$

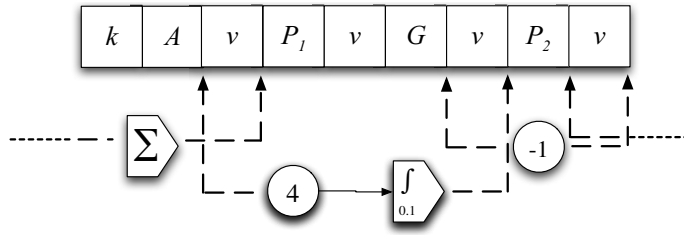
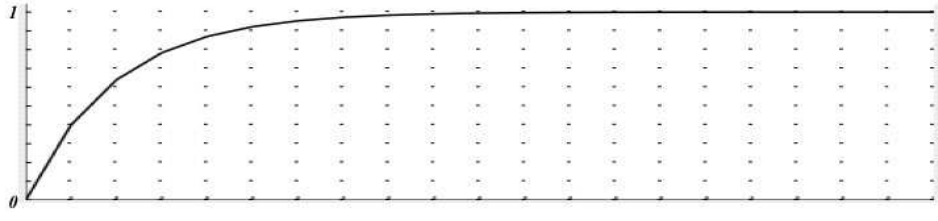


Fig. 9. Initial configuration for the automaton modeling a RC circuit.

						c_6		
1	A	0.00000	P1	0.00000	0	0.00000	P2	0.00000
1	A	1.00000	P1	0.00000	0	0.00000	P2	−0.00000
1	A	1.00000	P1	0.40000	1	0.00000	P2	−0.00000
1	A	1.00000	P1	0.40000	0	0.40000	P2	−0.00000
1	A	1.00000	P1	0.40000	0	0.40000	P2	−0.40000
1	A	0.60000	P1	0.40000	0	0.40000	P2	−0.40000
1	A	0.60000	P1	0.24000	1	0.40000	P2	−0.40000
1	A	0.60000	P1	0.24000	0	0.64000	P2	−0.40000
1	A	0.60000	P1	0.24000	0	0.64000	P2	−0.64000
1	A	0.36000	P1	0.24000	0	0.64000	P2	−0.64000
1	A	0.36000	P1	0.14400	1	0.64000	P2	−0.64000
1	A	0.36000	P1	0.14400	0	0.78400	P2	−0.64000
1	A	0.36000	P1	0.14400	0	0.78400	P2	−0.78400
1	A	0.21600	P1	0.14400	0	0.78400	P2	−0.78400
1	A	0.21600	P1	0.08640	1	0.78400	P2	−0.78400
1	A	0.21600	P1	0.08640	0	0.87040	P2	−0.78400
1	A	0.21600	P1	0.08640	0	0.87040	P2	−0.87040
1	A	0.12960	P1	0.08640	0	0.87040	P2	−0.87040
1	A	0.12960	P1	0.05184	1	0.87040	P2	−0.87040
1	A	0.12960	P1	0.05184	0	0.92224	P2	−0.87040
1	A	0.12960	P1	0.05184	0	0.92224	P2	−0.92224

Fig. 10. Evolution of the automaton modeling a RC circuit.

						c_6		
1	A	0.00000	P1	0.00000	0	0.00000	P2	0.00000
1	A	1.00000	P1	0.40000	0	0.40000	P2	−0.40000
1	A	0.60000	P1	0.24000	0	0.64000	P2	−0.64000
1	A	0.36000	P1	0.14400	0	0.78400	P2	−0.78400
1	A	0.21600	P1	0.08640	0	0.87040	P2	−0.87040
1	A	0.12960	P1	0.05184	0	0.92224	P2	−0.92224
1	A	0.07776	P1	0.03110	0	0.95334	P2	−0.95334
1	A	0.04666	P1	0.01866	0	0.97201	P2	−0.97201
1	A	0.02799	P1	0.01120	0	0.98320	P2	−0.98320
1	A	0.01680	P1	0.00672	0	0.98992	P2	−0.98992
1	A	0.01008	P1	0.00403	0	0.99395	P2	−0.99395
1	A	0.00605	P1	0.00242	0	0.99637	P2	−0.99637
1	A	0.00363	P1	0.00145	0	0.99782	P2	−0.99782
1	A	0.00218	P1	0.00087	0	0.99869	P2	−0.99869
1	A	0.00131	P1	0.00052	0	0.99922	P2	−0.99922
1	A	0.00078	P1	0.00031	0	0.99953	P2	−0.99953
1	A	0.00047	P1	0.00019	0	0.99972	P2	−0.99972
1	A	0.00028	P1	0.00011	0	0.99983	P2	−0.99983
1	A	0.00017	P1	0.00007	0	0.99990	P2	−0.99990
1	A	0.00010	P1	0.00004	0	0.99994	P2	−0.99994
1	A	0.00006	P1	0.00002	0	0.99996	P2	−0.99996



Numerical values of cell c_6 in time

Fig. 11. Numerical solution of a RC circuit.

behavior of c_6. This one presents an exponential behavior to 1, describing adequately the solution of Eq. (14).

4.2. *Mass-spring system*

For this system we shall use almost the same cellular automaton, only the initial configuration c^0 is different (Fig. 12).

Figure 13 presents 21 iterations of the automaton, cell c_{10} contains the numerical solution of the differential equation in every step. The automaton shows a feedback of the system in seven time steps, where the initial parameters in c_5 and c_9 are min = 6 and min = 4, respectively, both states have val = 0 and max = 7. Finally, state $P1$ holds that val = 0.01.

Figure 14 depicts the system every seven iterations; due to the small integration step in $P1$, it is not practical to display the evolution showing a representative behavior over a large period of time. Thereby we shall graph c_{10} after 21 000 iterations, presenting only its value in every seven steps in Fig. 15, effectuating 3000 cycles for the numerical calculation of the mass-spring system.

Figure 15 presents the harmonic oscillation between 0 and 2 of c_{10}, therefore the automaton correctly describes the solution in Eq. (16).

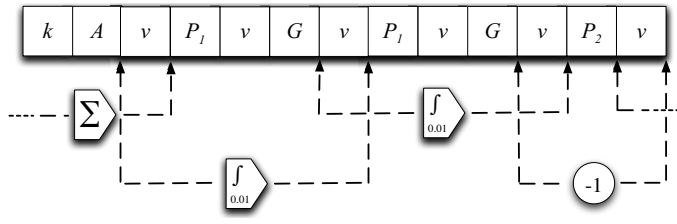


Fig. 12. Initial configuration for the automaton modeling a mass-spring system.

													c_{10}
1	A	0.00000	P1	0.00000	0	0.00000	P1	0.00000	0	0.00000	P2	0.00000
1	A	1.00000	P1	0.00000	0	0.00000	P1	0.00000	0	0.00000	P2	−0.00000
1	A	1.00000	P1	0.01000	1	0.00000	P1	0.00000	0	0.00000	P2	−0.00000
1	A	1.00000	P1	0.01000	0	0.01000	P1	0.00000	0	0.00000	P2	−0.00000
1	A	1.00000	P1	0.01000	0	0.01000	P1	0.00010	1	0.00000	P2	−0.00000
1	A	1.00000	P1	0.01000	0	0.01000	P1	0.00010	0	0.00010	P2	−0.00000
1	A	1.00000	P1	0.01000	0	0.01000	P1	0.00010	0	0.00010	P2	−0.00010
1	A	0.99990	P1	0.01000	0	0.01000	P1	0.00010	0	0.00010	P2	−0.00010
1	A	0.99990	P1	0.01000	0	0.01000	P1	0.00010	0	0.00010	P2	−0.00010
1	A	0.99990	P1	0.01000	1	0.01000	P1	0.00010	0	0.00010	P2	−0.00010
1	A	0.99990	P1	0.01000	0	0.02000	P1	0.00010	0	0.00010	P2	−0.00010
1	A	0.99990	P1	0.01000	0	0.02000	P1	0.00020	1	0.00010	P2	−0.00010
1	A	0.99990	P1	0.01000	0	0.02000	P1	0.00020	0	0.00030	P2	−0.00010
1	A	0.99990	P1	0.01000	0	0.02000	P1	0.00020	0	0.00030	P2	−0.00030
1	A	0.99970	P1	0.01000	0	0.02000	P1	0.00020	0	0.00030	P2	−0.00030
1	A	0.99970	P1	0.01000	0	0.02000	P1	0.00020	0	0.00030	P2	−0.00030
1	A	0.99970	P1	0.01000	1	0.02000	P1	0.00020	0	0.00030	P2	−0.00030
1	A	0.99970	P1	0.01000	0	0.03000	P1	0.00020	0	0.00030	P2	−0.00030
1	A	0.99970	P1	0.01000	0	0.03000	P1	0.00030	1	0.00030	P2	−0.00030
1	A	0.99970	P1	0.01000	0	0.03000	P1	0.00030	0	0.00060	P2	−0.00030
1	A	0.99970	P1	0.01000	0	0.03000	P1	0.00030	0	0.00060	P2	−0.00060

Fig. 13. Evolution of the automaton modeling a mass-spring system.

										c_{10}		
1	A	0.00000	P1	0.00000	0	0.00000	P1	0.00000	0	0.00000	P2	0.00000
1	A	0.99990	P1	0.01000	0	0.01000	P1	0.00010	0	0.00010	P2	-0.00010
1	A	0.99970	P1	0.01000	0	0.02000	P1	0.00020	0	0.00030	P2	-0.00030
1	A	0.99940	P1	0.01000	0	0.03000	P1	0.00030	0	0.00060	P2	-0.00060
1	A	0.99900	P1	0.00999	0	0.03999	P1	0.00040	0	0.00100	P2	-0.00100
1	A	0.99850	P1	0.00999	0	0.04998	P1	0.00050	0	0.00150	P2	-0.00150
1	A	0.99790	P1	0.00999	0	0.05997	P1	0.00060	0	0.00210	P2	-0.00210
1	A	0.99720	P1	0.00998	0	0.06994	P1	0.00070	0	0.00280	P2	-0.00280
1	A	0.99640	P1	0.00997	0	0.07992	P1	0.00080	0	0.00360	P2	-0.00360
1	A	0.99550	P1	0.00996	0	0.08988	P1	0.00090	0	0.00450	P2	-0.00450
1	A	0.99450	P1	0.00996	0	0.09984	P1	0.00100	0	0.00550	P2	-0.00550
1	A	0.99341	P1	0.00995	0	0.10978	P1	0.00110	0	0.00659	P2	-0.00659
1	A	0.99221	P1	0.00993	0	0.11971	P1	0.00120	0	0.00779	P2	-0.00779
1	A	0.99091	P1	0.00992	0	0.12964	P1	0.00130	0	0.00909	P2	-0.00909
1	A	0.98952	P1	0.00991	0	0.13955	P1	0.00140	0	0.01048	P2	-0.01048
1	A	0.98802	P1	0.00990	0	0.14944	P1	0.00149	0	0.01198	P2	-0.01198
1	A	0.98643	P1	0.00988	0	0.15932	P1	0.00159	0	0.01357	P2	-0.01357
1	A	0.98474	P1	0.00986	0	0.16919	P1	0.00169	0	0.01526	P2	-0.01526
1	A	0.98295	P1	0.00985	0	0.17903	P1	0.00179	0	0.01705	P2	-0.01705
1	A	0.98106	P1	0.00983	0	0.18886	P1	0.00189	0	0.01894	P2	-0.01894
1	A	0.97907	P1	0.00981	0	0.19867	P1	0.00199	0	0.02093	P2	-0.02093

Fig. 14. Extended evolution of the automaton modeling a mass-spring system.

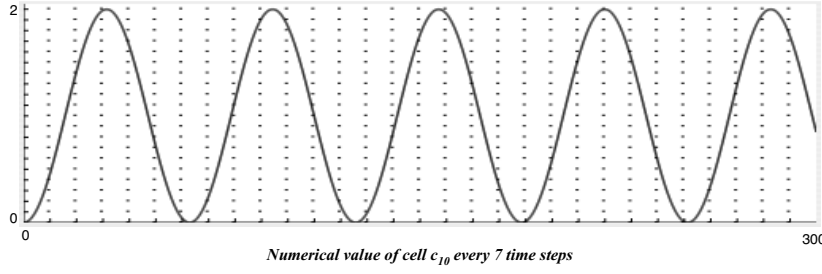


Fig. 15. Dynamical behavior of c_{10}.

Table 3. Neighborhoods modeling a mass-spring-damper system.

Φ	φ	Φ	φ
$v_1 k \mathrm{A} v_2$	$v_2 = k + v_1$	$v_1 \mathrm{P1} v_2$	$v_2 = 0.01 * v_1$
$v_1 \mathrm{G} v_2$	$v2 = v2 + v1$ iff $G = 1$	$v_1 \mathrm{P2} v_2$	$v_2 = -2 * v_1$
$v_1 \mathrm{E} v_2$	$v2 = v1$ iff $E = 1$	$v_1 \mathrm{P3} v_2$	$v_2 = -100 * v_1$
$v_1 \underbrace{\cdots\cdots}_{4\,\text{cells}} \mathrm{C} v_2$	$v_2 = v_1$	$v_1 \underbrace{\cdots\cdots}_{3\,\text{cells}} v_2 \mathrm{S} v_3$	$v_3 = v_1 + v_2$

4.3. *Mass-spring-damper system*

We shall use a cellular automaton with $K = \mathbb{R} \cup \{A, P1, P2, P3, G, C, S, E\}$ and the neighborhoods showed in Table 3; taking initially $v = 0$ and $k = 100$.

The initial configuration for this system is described in Fig. 16. We use state C for retrieving the original result of each integrator, state S combines adequately both results when it is needed and state E allows to pass information from left to right for a correct feedback of the system.

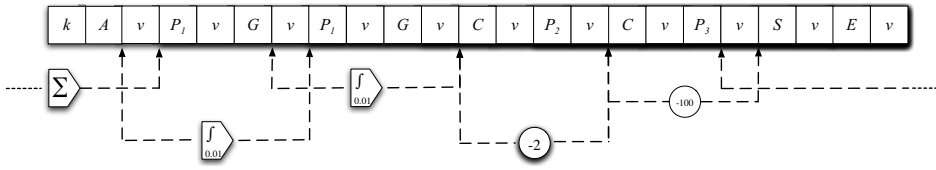


Fig. 16. Initial configuration for the automaton modeling a mass-spring-damper system.

c_{10}

100	R	0.00000	P1	0.00000	0	0.00000	P1	0.00000	0	0.00000	C	0.00000	P2	0.00000	C	0.00000	P3	0.00000	S	0.00000	0	0.00000
100	R	100.00000	P1	0.00000	0	0.00000	P1	0.00000	0	0.00000	C	0.00000	P2	-0.00000	C	0.00000	P3	-0.00000	S	0.00000	0	0.00000
100	R	100.00000	P1	1.00000	1	0.00000	P1	0.00000	0	0.00000	C	0.00000	P2	-0.00000	C	0.00000	P3	-0.00000	S	0.00000	0	0.00000
100	R	100.00000	P1	1.00000	0	1.00000	P1	0.00000	0	0.00000	C	0.00000	P2	-0.00000	C	0.00000	P3	-0.00000	S	0.00000	0	0.00000
100	R	100.00000	P1	1.00000	0	1.00000	P1	0.01000	1	0.00000	C	1.00000	P2	-0.00000	C	0.00000	P3	-0.00000	S	0.00000	0	0.00000
100	R	100.00000	P1	1.00000	0	1.00000	P1	0.01000	0	0.01000	C	1.00000	P2	-2.00000	C	0.00000	P3	-0.00000	S	0.00000	0	0.00000
100	R	100.00000	P1	1.00000	0	1.00000	P1	0.01000	0	0.01000	C	1.00000	P2	-2.00000	C	0.01000	P3	-0.00000	S	-2.00000	1	0.00000
100	R	100.00000	P1	1.00000	0	1.00000	P1	0.01000	0	0.01000	C	1.00000	P2	-2.00000	C	0.01000	P3	-1.00000	S	-2.00000	0	-2.00000
100	R	98.00000	P1	1.00000	0	1.00000	P1	0.01000	0	0.01000	C	1.00000	P2	-2.00000	C	0.01000	P3	-1.00000	S	-3.00000	0	-2.00000
100	R	98.00000	P1	0.98000	1	1.00000	P1	0.01000	0	0.01000	C	1.00000	P2	-2.00000	C	0.01000	P3	-1.00000	S	-3.00000	0	-2.00000
100	R	98.00000	P1	0.98000	0	1.98000	P1	0.01000	0	0.01000	C	1.00000	P2	-2.00000	C	0.01000	P3	-1.00000	S	-3.00000	0	-2.00000
100	R	98.00000	P1	0.98000	0	1.98000	P1	0.01980	1	0.01000	C	1.98000	P2	-2.00000	C	0.01000	P3	-1.00000	S	-3.00000	0	-2.00000
100	R	98.00000	P1	0.98000	0	1.98000	P1	0.01980	0	0.02980	C	1.98000	P2	-3.96000	C	0.01000	P3	-1.00000	S	-3.00000	0	-2.00000
100	R	98.00000	P1	0.98000	0	1.98000	P1	0.01980	0	0.02980	C	1.98000	P2	-3.96000	C	0.02980	P3	-1.00000	S	-4.96000	1	-2.00000
100	R	98.00000	P1	0.98000	0	1.98000	P1	0.01980	0	0.02980	C	1.98000	P2	-3.96000	C	0.02980	P3	-2.98000	S	-4.96000	0	-4.96000
100	R	95.04000	P1	0.98000	0	1.98000	P1	0.01980	0	0.02980	C	1.98000	P2	-3.96000	C	0.02980	P3	-2.98000	S	-6.94000	0	-4.96000
100	R	95.04000	P1	0.95040	1	1.98000	P1	0.01980	0	0.02980	C	1.98000	P2	-3.96000	C	0.02980	P3	-2.98000	S	-6.94000	0	-4.96000
100	R	95.04000	P1	0.95040	0	2.93040	P1	0.01980	0	0.02980	C	1.98000	P2	-3.96000	C	0.02980	P3	-2.98000	S	-6.94000	0	-4.96000
100	R	95.04000	P1	0.95040	0	2.93040	P1	0.02930	1	0.02980	C	2.93040	P2	-3.96000	C	0.02980	P3	-2.98000	S	-6.94000	0	-4.96000
100	R	95.04000	P1	0.95040	0	2.93040	P1	0.02930	0	0.05910	C	2.93040	P2	-5.86080	C	0.02980	P3	-2.98000	S	-6.94000	0	-4.96000
100	R	95.04000	P1	0.95040	0	2.93040	P1	0.02930	0	0.05910	C	2.93040	P2	-5.86080	C	0.05910	P3	-2.98000	S	-8.84080	1	-4.96000
100	R	95.04000	P1	0.95040	0	2.93040	P1	0.02930	0	0.05910	C	2.93040	P2	-5.86080	C	0.05910	P3	-5.91040	S	-8.84080	0	-8.84080

Fig. 17. Evolution of the automaton modeling a mass-spring-damper system.

Figure 17 presents 21 iterations of the automaton, c_{10} contains the numerical solution of the differential equation. This figure shows a complete cycle of the system in seven steps; initially the sum-gate state in c_5 has min = 6, c_9 has min = 4 and finally c_{21} has min = 2, all the three states have val = 0 and max = 7.

Figure 18 depicts the evolution of the system every seven time steps, as before, it is not practical to display a representative evolution over a large period of time by the small integration step in $P1$. We graph the value of c_{10} in 2100 iterations of the system, presenting only the value in every seven steps in Fig. 19 for 300 cycles of the numerical solution. The graph presents an underdamped oscillation between 0 and 2 of the values in c_10, therefore the automaton gives the expected approximation for the dynamics of Eq. (18) using Euler's method.

5. Concluding Remarks

We have modeling linear dynamical systems using one-dimensional cellular automata. Taken results from preceding works, the original part of the constructions presented in the paper is that real values can be combined with operations in the evolution space in order to achieve the required numerical calculations. This approach has been easily implemented in a computer system, few cells are required for simulating each one of the previous examples, hence the execution time needed for obtaining numerical solutions is the same that the one used by a traditional computer program using Euler's method. The automata described in this manuscript

c_{10}

100	R	0.00000	P1	0.00000	0	0.00000	P1	0.00000	0	0.00000	C	0.00000	P2	0.00000	C	0.00000	P3	0.00000	S	0.00000	0	0.00000
100	R	100.00000	P1	1.00000	0	1.00000	P1	0.01000	0	0.01000	C	1.00000	P2	-2.00000	C	0.01000	P3	-1.00000	S	-2.00000	0	-2.00000
100	R	98.00000	P1	0.98000	0	1.98000	P1	0.01980	0	0.02980	C	1.96000	P2	-3.96000	C	0.02980	P3	-2.98000	S	-4.96000	0	-4.96000
100	R	95.04000	P1	0.95040	0	2.93040	P1	0.02930	0	0.05910	C	2.93040	P2	-5.86080	C	0.05910	P3	-5.91040	S	-8.84080	0	-8.84080
100	R	91.15920	P1	0.91159	0	3.84199	P1	0.03842	0	0.09752	C	3.84199	P2	-7.68398	C	0.09752	P3	-9.75239	S	-13.59438	0	-13.59438
100	R	86.40562	P1	0.86406	0	4.70605	P1	0.04706	0	0.14458	C	4.70605	P2	-9.41210	C	0.14458	P3	-14.45844	S	-19.16449	0	-19.16449
100	R	80.83551	P1	0.80836	0	5.51440	P1	0.05514	0	0.19973	C	5.51440	P2	-11.02881	C	0.19973	P3	-19.97284	S	-25.48725	0	-25.48725
100	R	74.51276	P1	0.74513	0	6.25953	P1	0.06260	0	0.26232	C	6.25953	P2	-12.51906	C	0.26232	P3	-26.23237	S	-32.49191	0	-32.49191
100	R	67.50809	P1	0.67508	0	6.93461	P1	0.06935	0	0.33167	C	6.93461	P2	-13.86922	C	0.33167	P3	-33.16699	S	-40.10160	0	-40.10160
100	R	59.89840	P1	0.59898	0	7.53360	P1	0.07534	0	0.40701	C	7.53360	P2	-15.06719	C	0.40701	P3	-40.70058	S	-48.23418	0	-48.23418
100	R	51.76582	P1	0.51766	0	8.05125	P1	0.08051	0	0.48752	C	8.05125	P2	-16.10251	C	0.48752	P3	-48.75184	S	-56.80309	0	-56.80309
100	R	43.19691	P1	0.43197	0	8.48322	P1	0.08483	0	0.57235	C	8.48322	P2	-16.96645	C	0.57235	P3	-57.23506	S	-65.71828	0	-65.71828
100	R	34.28172	P1	0.34282	0	8.82604	P1	0.08826	0	0.66061	C	8.82604	P2	-17.65208	C	0.66061	P3	-66.06110	S	-74.88715	0	-74.88715
100	R	25.11285	P1	0.25113	0	9.07717	P1	0.09077	0	0.75138	C	9.07717	P2	-18.15434	C	0.75138	P3	-75.13827	S	-84.21544	0	-84.21544
100	R	15.78456	P1	0.15785	0	9.23501	P1	0.09235	0	0.84373	C	9.23501	P2	-18.47003	C	0.84373	P3	-84.37528	S	-93.60829	0	-93.60829
100	R	6.39171	P1	0.06392	0	9.29893	P1	0.09299	0	0.93672	C	9.29893	P2	-18.59786	C	0.93672	P3	-93.67222	S	-102.97115	0	-102.97115
100	R	-2.97115	P1	-0.02971	0	9.26922	P1	0.09269	0	1.02941	C	9.26922	P2	-18.53844	C	1.02941	P3	-102.94143	S	-112.21066	0	-112.21066
100	R	-12.21066	P1	-0.12211	0	9.14711	P1	0.09147	0	1.12089	C	9.14711	P2	-18.29423	C	1.12089	P3	-112.08854	S	-121.23566	0	-121.23566
100	R	-21.23566	P1	-0.21236	0	8.93476	P1	0.08935	0	1.21023	C	8.93476	P2	-17.86951	C	1.21023	P3	-121.02330	S	-129.95805	0	-129.95805
100	R	-29.95805	P1	-0.29958	0	8.63518	P1	0.08635	0	1.29658	C	8.63518	P2	-17.27035	C	1.29658	P3	-129.65848	S	-138.29366	0	-138.29366
100	R	-38.29366	P1	-0.38294	0	8.25224	P1	0.08252	0	1.37911	C	8.25224	P2	-16.50448	C	1.37911	P3	-137.91071	S	-146.16296	0	-146.16296
100	R	-46.16296	P1	-0.46163	0	7.79061	P1	0.07791	0	1.45701	C	7.79061	P2	-15.58122	C	1.45701	P3	-145.70132	S	-153.49193	0	-153.49193

Fig. 18. Extended evolution of the automaton modeling a mass-spring-damper system.

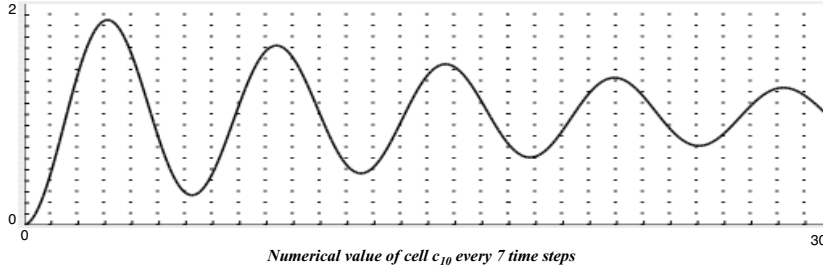


Fig. 19. Dynamical behavior of c_{10} modeling a mass-spring-damper system.

have resolved well-known linear systems by means of local interactions over a finite set of elements taking discrete time steps, proving that they can be used for realizing different kinds of calculations without any need of a global control. Looking for an enrichment of this technique, further improvements will be done to yield procedures which can be applied for non-linear dynamical systems and be employed not only in academic problems but in practical issues; for instance:

- Implementing more efficient integration methods as Heun or Runge-Kutta.
- Establishing a complete-structured methodology based on cellular automata for modeling dynamical systems, in this sense the classical use of block diagrams may be useful to formalize the basis of this paradigm.
- Investigate the relation between models based on cellular automata and other alternative tools used for the same task, for instance bond graphs.

References

1. J. von Neumann, *Theory of Self-Reproducing Automata*, ed. A. W. Burks (University of Illinois Press, 1966).
2. A. Adamatzky (ed.), *Collision-Based Computing* (Springer, 2002).
3. J. Park, K. Steiglitz and W. Thurston, *Physica D* **19**, 423 (1986).

4. G. Juárez, H. V. McIntosh and J. C. Seck Tuoh, Lecture Notes Comp. Sci., 2801, 2003, p. 175.
5. G. Juárez, H. V. McIntosh and J. C. Seck Tuoh, *Int. J. Unconven. Comp.* **2**, 1 (2006).
6. M. Cook, *Complex Systems* **15**, 140 (2004).
7. S. Wolfram (ed.), *A New Kind of Science* (Wolfram Media, 2002).
8. D. L. Eager and J. Zahorjan, *ACM Trans. Comp. Syst.* **11**(1), 1 (1993).
9. D. Mavriplis, R. Das, R. E. Vermeland and J. Saltz, *Proc. 1992 ACM/IEEE Conf. Supercomput.* (1992), pp. 132–141.
10. T. E. Marlin (eds.), *Process Control: Designing Processes and Control Systems for Dynamic Performance* (McGraw-Hill, 1995).
11. B. Chopard and M. Droz (eds.), *Cellular Automata Modeling of Physical Systems* (Cambridge University Press, 1998).
12. D. Ostrov and R. Rucker, *Complex Systems* **10**, 91 (1996).
13. R. Rucker, in *New Constructions in Cellular Automata*, eds. D. Griffeath and C. Moore (Oxford University Press, 2003), pp. 295–315.
14. G. A. Hedlund, *Mathem. Systems Theory* **3**, 320 (1969).
15. M. Boyle and A. Maass, *J. Math. Soc. Jpn.* **52**(4), 725 (2000).
16. H. C. Shen and W. P. Yan, *Int. J. Advanced Manufact. Technol.* **7**, 333 (1992).
17. E. A. Coddington, *An Introduction to Ordinary Differential Equations* (Dover Books on Advanced Mathematics, 1989).
18. W. E. Boyce and R. C. DiPrima, *Elementary Differential Equations and Boundary Value Problems* (John Wiley, 2005).
19. G. Juárez, A. Adamatzky and H. V. McIntosh, *Chaos, Fractals & Solitons* **28**, 100 (2006).