2D P Colonies and Modelling of Liquid Flow Over the Earth's Surface

Luděk Cienciala, Lucie Ciencialová, and Miroslav Langer

Institute of Computer Science
and
Research Institute of the IT4Innovations Centre of Excellence,
Silesian University in Opava, Czech Republic
{ludek.cienciala, lucie.ciencialova, miroslav.langer}@fpf.slu.cz

Summary. We continue the investigation of 2D P colonies introduced in [1], a class of abstract computing devices composed of independent agents, acting and evolving in a shared 2D environment where the agents are located. Agents have limited information about the contents of the environment where they can move in four directions.

1 Introduction

P colonies were introduced in the paper [5] as formal models of computing devices belonging to membrane systems and similar to formal grammars called colonies. This model is inspired by the structure and the behaviour of communities of living organisms in a shared environment. The independent organisms living in a P colony are called agents. Each agent is represented by several objects embedded in a membrane. The number of objects inside each agent is the same and constant during computation. The environment is agents' communication channel and storage place for objects. At any moment all agents "know" about all the objects in the environment and they can access any object immediately. More information about P colonies the reader can find in [4, 2]. P colonies are one of the types of P systems. They were introduced in 2000 in [6] by Gheorghe Păun as a formal model inspired by the structure and the behaviour of cells.

With each agent a set of programs is associated. The program, which determines the activity of an agent, is very simple and depends on the contents of agents and on types and number of objects placed in the environment. An agent can change the contents of the environment through programs and it can affect the behaviour of other agents through the environment. This influence between agents is the key factor in the functioning of the P colony. At any moment each object inside every agent is affected by the execution of the program.

For more information about P systems see [8, 7] or [11].

In addition 2D P colony has the environment in a form of a 2D grid of square cells. The agents are located in this grid and their view is limited to the cells that immediately surround them. Based on the contents of these cells, the agents decide their future locations.

Behaviour of each agent is based on its set of programs. The programs are formed from two rules of type rewriting, communication and movement. By using the rewriting rule one object within the agent is changed (evolved) to another object. When the communication rule is applied one object from the environment is consumed by the agent and one object from content of the agent is placed to the environment. The last type of rules is the movement rule. The condition for the movement of an agent is to find specific objects in specific locations in the environment. This is specified by a matrix with elements - objects. The agent is looking for at most one object in every surrounding cell. If the condition is fulfilled then the agent moves one cell up, down, left or right.

The program can contain one movement rule at most. To achieve the greatest simplicity in agent behaviour, we set another condition. If the agent moves, it cannot communicate with the environment. So if the program contains a movement rule, then the second rule is the rewriting rule.

Although the colony is a theoretical computing model through 2D, it is a suitable tool for modelling the behaviour of natural multi-agent systems - colonies of bacteria or ants, spreading substances in homogeneous and inhomogeneous medium.

In this paper we present hydrological modelling flow of liquid over the Earth's surface using 2D P colonies. Based on the entered data - the slope surface, a source of fluid and quantity - we simulate the fluid distribution in the environment.

The first part of the paper is devoted to 2D P colonies. The rest is organised as follows: The issue of the flow of liquid over the surface, problem solution - maps preparation, definition of the agent, process simulation, comparison with cellular automaton and future expansion.

2 Definitions

Throughout the paper we assume that the reader is familiar with the basics of the formal language theory.

We use NRE to denote the family of the recursively enumerable sets of natural numbers. Let Σ be the alphabet. Let Σ^* be the set of all words over Σ (including the empty word ε). We denote the length of the word $w \in \Sigma^*$ by $|w|$ and the number of occurrences of the symbol $a \in \Sigma$ in w by $|w|_a$.

A multiset of objects M is a pair $M = (V, f)$, where V is an arbitrary (not necessarily finite) set of objects and f is a mapping $f : V \to N$; f assigns to each object in V its multiplicity in M. The set of all multisets with the set of objects V is denoted by V°. The set V' is called the support of M and is denoted by $supp(M)$ if for all $x \in V'$ $f(x) \neq 0$ holds. The cardinality of M, denoted by

$|M|$, is defined by $|M| = \sum_{a \in V} f(a)$. Each multiset of objects M with the set of objects $V' = \{a_1, \ldots a_n\}$ can be represented as a string w over alphabet V', where $|w|_{a_i} = f(a_i);\ 1 \leq i \leq n$. Obviously, all words obtained from w by permuting the letters represent the same multiset M. The ε represents the empty multiset.

3 2D P colonies

We briefly summarize the notion of 2D P colonies. A P colony consists of agents and an environment. Both the agents and the environment contain objects. With each agent a set of programs is associated. There are three types of rules in the programs.

The first rule type, called the evolution rule, is of the form $a \ \rightarrow b$. It means that the object a inside the agent is rewritten (evolved) to the object b. The second rule type, called the communication rule, is of the form $c \leftrightarrow d$. When the communication rule is performed, the object c inside the agent and the object d outside the agent swap their places. Thus, after the execution of the rule, the object d appears inside the agent and the object c is placed outside the agent.

The third rule type, called the motion rule, is of the form matrix $3 \times 3 \rightarrow$ move direction. Based on the contents of the neighbouring cells, an agent can move one step to the left, right, up or down.

A program can contain maximum one motion rule. When there is a motion rule inside a program, there cannot be a communication rule inside the same program.

Definition 1. *The 2D P colony is a construct*
$$\Pi = (A, e, Env, B_1, \ldots, B_k, f), k \geq 1, \textit{ where}$$

- *A is an alphabet of the colony, its elements are called objects,*
- *$e \in A$ is the basic environmental object of the colony,*
- *Env is a pair $(m \times n, w_E)$, where $m \times n, m, n \in N$ is the size of the environment and w_E is the initial contents of environment, it is a matrix of size $m \times n$ of multisets of objects over $A - \{e\}$.*
- *$B_i,\ 1 \leq i \leq k$, are agents, each agent is a construct $B_i = (O_i, P_i, [o, p])$, $0 \leq o \leq m,\ 0 \leq p \leq n$, where*
  - *O_i is a multiset over A, it determines the initial state (contents) of the agent, $|O_i| = 2$,*
  - *$P_i = \{p_{i,1}, \ldots, p_{i,l_i}\}, l \geq 1, 1 \leq i \leq k$ is a finite set of programs, where each program contains exactly 2 rules, which are in one of the following forms each:*
    - *$a \rightarrow b$, called the evolution rule,*
    - *$c \leftrightarrow d$, called the communication rule,*
    - *$[a_{q,r}] \rightarrow s, 0 \leq q, r \leq 2, s \in \{\Leftarrow, \Rightarrow, \Uparrow, \Downarrow\}$, called the motion rule;*
- *$f \in A$ is the final object of the colony.*

The configuration of the 2D P colony is given by the state of the environment - matrix of type $m \times n$ with multisets of objects over $A - \{e\}$ as its elements, and

by the state of all agents - pairs of objects from alphabet A and the coordinates of the agents. An initial configuration is given by the definition of the 2D P colony.

The computational step consists of three parts. The first part lies in determining the applicable set of programs according to the actual configuration of the P colony. There are programs belonging to all agents in this set of programs. In the second part we have to choose one program corresponding to each agent from the set of applicable programs. There is no collision between the communication rules belonging to different programs. The third part is the execution of the chosen programs.

A change of the configuration is triggered by the execution of programs and it involves changing the state of the environment, contents and placement of the agents.

The computation is nondeterministic and maximally parallel. The computation ends by halting when no agent has an applicable program.

The result of the computation is the number of copies of the final object placed in the environment at the end of the computation.

Another way to determine the result of the computation is to take into account not only the number of objects but also their location. The result could then be a grayscale image, a character string or a number that is dependent on both the number and pposition of the objects (for example, $g = \sum_{j=0}^{n-1} \left(\sum_{i=0}^{m-1} f(i,j) \right) \cdot n^i$, where $f(i,j)$ is the number of copies of object f in the $[i,j]$-cell).

The reason for the introduction of 2D P colonies is not the study of their computational power but monitoring their behaviour during the computation. We can define certain measures to assess the dynamics of the computation:

- the number of moves of agents
- the number of visited cells (or not visited cells)
- the number of copies of a certain object in the home cell or throughout the environment.

These measures can be observed both for the individual steps of the computation and the computation as a whole.

4 The issue of the flow of liquid over the surface

The issue of the flow of liquid over the Earth's surface is studied by experts from two areas - hydrology and geoinformatics. Both of these disciplines work closely together on the issue of the so-called "surface runoff". Surface runoff is the water flow that occurs when the soil is infiltrated to full capacity and excess water from rain, meltwater, or other sources flows over the land.

Surface runoff can be generated in four reasons: infiltration excess overland flow, saturation excess overland flow, antecedent soil moisture, subsurface return flow. Infiltration excess overland flow occurs when the rate of rainfall on a surface exceeds the rate at which water can infiltrate the ground, and any depression

storage has already been filled. When the soil is saturated and the depression storage filled, and rain continues to fall, the rainfall will immediately produce surface runoff - saturation excess overland flow. Soil retains a degree of moisture after a rainfall. This residual water moisture (antecedent soil moisture) affects the soil's infiltration capacity. During the next rainfall event, the infiltration capacity will cause the soil to be saturated at a different rate. The higher the level of antecedent soil moisture, the more quickly the soil becomes saturated. Once the soil is saturated, runoff occurs. After water infiltrates the soil on an up-slope portion of a hill, the water may flow laterally through the soil, and exfiltrate (flow out of the soil) closer to a channel. This is called subsurface return flow or throughflow.

We can say that generation surface runoff depends on type of soil, temperature, humidity and rainfall. The task of our model is to determine which way the flow would run and which areas could be affected by flash floods.

5 Problem solution

We divide solution of the problem into two parts - (1) preparation of maps (2D P colony's environment) and (2) definition of agents. We assume that the soil is already saturated thus the main factor of overland flow is the slope of the field.

5.1 Preparation of maps

Map data is obtained from the geographic information system (GIS) and processing system ArcGIS. We use the map data for the Czech Republic called the digital model of the terrain in scale 1: 25 000 (DMÚ25).

Raster graphics images are probably the most appropriate format for modelling real-world phenomena in the field of GIS. To process this format, many tools were created and can be used for performing various analyses. A raster image is composed of a regular network of cells, usually in a square shape, to which values of displayed properties can be assigned independently. More information about GIS and image processing the reader can find in [3] and about geosimulation in [10].

The first step to simulate the flow of liquid over relief was the determination of its runoff from individual pixels (cells). Gradient with respect to an adjacent cell is defined as the ratio of the height difference to the horizontal distance. Gradient is positive due to the lower neighbours, or negative due to higher and zero in relation to the neighbours of the same height. Lowest neighbour is neighbour with the largest positive gradient.

Basic classification algorithms to calculate the runoff:

- Single flow direction (SFD) - each pixel of the liquid flows in one direction only (toward neighbour in the direction of the largest gradient). Each pixel belongs to only one basin.

- Multiple flow direction (MFD) - fluid can flow out of each pixel in multiple directions, maximum of eight. In the case of MFD a unit volume flow is fairly distributed among all lower neighbours. The MFD may include the pixel to multiple basins.

There is implemented a tool for calculating the flow direction in ArcGIS software, called simply Flow direction. Flow Direction tool works as a simple flow direction (SFD). After its execution integer raster file is created that specifies the flow direction for each cell. Every cell can reach value ranging from 1 to 255.

Eight basic directions of the flow are represented by the numbers $1, 2, 4, 8, 16$, 32, 64 and 128 (see Table 1). Other directions are generated as sums of values of the basic directions.

32	64	128
16	↖ ↑ ↗ ← →	1
	↙ ↓ ↘	
8	4	2

Table 1. The numbers of eight basic directions

When creating the model, we used the test data to propose group of programs. The final visualization is based on data from DMÚ 25.

What we obtain from ArcGIS is a raster file with natural number in each cell corresponding to the runoff from this cell. Because 2D P colony works with discrete symbols and not with numbers, it needed to transcode numbers to symbols. A coding table is shown on Table 2

direction	→	←	↑	↓	↘	↙	↗	↖
symbol	a	E	i	m	q	u	y	2

Table 2. The coding table

The first processed map is map without drainless area and its size is 20×12 and it is shown on the Table 3. Transcoded symbols are shown on the Table 4.

5.2 Definition of the agent

Agents in 2D P colonies have capacity 2. It follows that the agent contains two objects, and each program is composed by two rules.

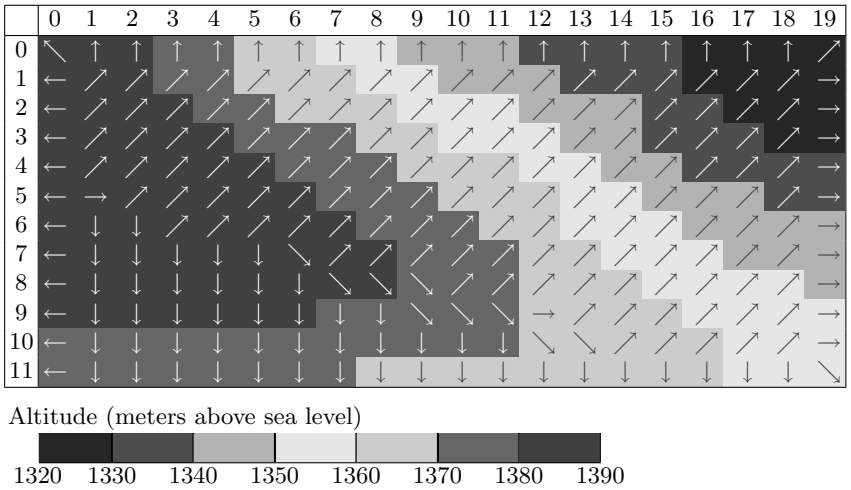

Table 3. Processed map

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	2	i	i	i	i	i	i	i	i	i	i	i	i	i	i	i	i	i	i	y
1	E	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	a
2	E	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	a
3	E	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	a
4	E	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	a
5	E	a	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	a
6	E	m	m	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	y	a
7	E	m	m	m	m	m	q	y	y	y	y	y	y	y	y	y	y	y	y	a
8	E	m	m	m	m	m	m	q	q	q	y	y	y	y	y	y	y	y	y	a
9	E	m	m	m	m	m	m	m	m	q	q	q	a	y	y	y	y	y	y	a
10	E	m	m	m	m	m	m	m	m	m	m	m	q	q	y	y	y	y	y	a
11	E	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	m	t

Table 4. Transcoded symbols

Each of the objects carries the information about the state of the agent. The first object has information about the activity of the agent. At this stage of the simulation it is the information that the agent "flows" down the terrain or it is still inactive (belonging to the rainfall that have not fall). The second object stores information about the previous direction of flow. This information can further modify the way of the agent as inertia.

Objects and their association to the flow directions are given in the following table.

direction	$\rightarrow$	$\leftarrow$	$\uparrow$	$\downarrow$	$\searrow$	$\swarrow$	$\nearrow$	$\nwarrow$
symbol	9	8	6	7	D	D	U	U
symbol	L	K	H	I	I	I	H	H

The inertia of crossing directions is modified because of longer distance between the centres of the cells.

The first subset of programs with priority 0 is defined for the first step of computation. The initial configuration of each "working" agent is Xe.

$$(1)\left\langle \begin{bmatrix} * & * & * \\ * & a & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ e \rightarrow 9 \right\rangle;\ (2)\left\langle \begin{bmatrix} * & * & * \\ * & E & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ e \rightarrow 8 \right\rangle;$$

$$(3)\left\langle \begin{bmatrix} * & * & * \\ * & i & * \\ * & * & * \end{bmatrix} \rightarrow \Uparrow;\ e \rightarrow 6 \right\rangle;\ (4)\left\langle \begin{bmatrix} * & * & * \\ * & m & * \\ * & * & * \end{bmatrix} \rightarrow \Downarrow;\ e \rightarrow 7 \right\rangle;$$

$$(5)\left\langle \begin{bmatrix} * & * & * \\ * & q & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ e \rightarrow D \right\rangle;\ (6)\left\langle \begin{bmatrix} * & * & * \\ * & u & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ e \rightarrow D \right\rangle;$$

$$(7)\left\langle \begin{bmatrix} * & * & * \\ * & y & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ e \rightarrow U \right\rangle;\ (8)\left\langle \begin{bmatrix} * & * & * \\ * & 2 & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ e \rightarrow U \right\rangle;$$

In the case of programs (5) and (6) (resp. (7) and (8)) it is neccessary to take one step down (resp. up).

$$(9)\left\langle \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \rightarrow \Downarrow;\ D \rightarrow I \right\rangle;\ (10)\left\langle \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix} \rightarrow \Uparrow;\ U \rightarrow H \right\rangle;$$

While agents apply programs with priority 1 (9) and (10), agents, that do not move in a cross direction, must stand. Therefore, they use a program composed of two rewriting rules. The programs have priority 2.

$(11)\,\langle X \rightarrow\ X;\ 6 \rightarrow H\rangle;\ (12)\,\langle X \rightarrow\ X;\ 7 \rightarrow I\rangle;\ (13)\,\langle X \rightarrow\ X;\ 8 \rightarrow K\rangle;$
$(14)\,\langle X \rightarrow\ X;\ 9 \rightarrow L\rangle;$

The following programs with priority 0 are used to guide the agent in the next steps, the agent may hold information about the movement in the previous step.

$$(15)\left\langle \begin{bmatrix} * & * & * \\ * & a & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ H \rightarrow 9 \right\rangle;\ (16)\left\langle \begin{bmatrix} * & * & * \\ * & E & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ H \rightarrow 8 \right\rangle;$$

$$(17)\left\langle \begin{bmatrix} * & * & * \\ * & i & * \\ * & * & * \end{bmatrix} \rightarrow \Uparrow;\ H \rightarrow \mathbf{U} \right\rangle;\ (18)\left\langle \begin{bmatrix} * & * & * \\ * & m & * \\ * & * & * \end{bmatrix} \rightarrow \Downarrow;\ H \rightarrow 7 \right\rangle;$$

$$(19)\left\langle \begin{bmatrix} * & * & * \\ * & q & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ H \rightarrow D \right\rangle;\ (20)\left\langle \begin{bmatrix} * & * & * \\ * & u & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ H \rightarrow D \right\rangle;$$

$$(21)\left\langle \begin{bmatrix} * & * & * \\ * & y & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ H \rightarrow U \right\rangle;\ (22)\left\langle \begin{bmatrix} * & * & * \\ * & 2 & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ H \rightarrow U \right\rangle;$$

$$(23)\left\langle \begin{bmatrix} * & * & * \\ * & a & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ I \rightarrow 9 \right\rangle;\ (24)\left\langle \begin{bmatrix} * & * & * \\ * & E & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ I \rightarrow 8 \right\rangle;$$

$$(25)\left\langle \begin{bmatrix} * & * & * \\ * & i & * \\ * & * & * \end{bmatrix} \rightarrow \Uparrow;\ I \rightarrow 6 \right\rangle;\ (26)\left\langle \begin{bmatrix} * & * & * \\ * & m & * \\ * & * & * \end{bmatrix} \rightarrow \Downarrow;\ I \rightarrow 7 \right\rangle;$$

$$(27)\left\langle \begin{bmatrix} * & * & * \\ * & q & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ I \rightarrow D \right\rangle;\ (28)\left\langle \begin{bmatrix} * & * & * \\ * & u & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ I \rightarrow D \right\rangle;$$

$$(29)\left\langle \begin{bmatrix} * & * & * \\ * & y & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ I \rightarrow U \right\rangle;\ (30)\left\langle \begin{bmatrix} * & * & * \\ * & 2 & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ I \rightarrow U \right\rangle;$$

$$(31)\left\langle \begin{bmatrix} * & * & * \\ * & a & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ J \rightarrow 9 \right\rangle;\ (32)\left\langle \begin{bmatrix} * & * & * \\ * & E & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ J \rightarrow L \right\rangle;$$

$$(33)\left\langle \begin{bmatrix} * & * & * \\ * & i & * \\ * & * & * \end{bmatrix} \rightarrow \Uparrow;\ J \rightarrow 6 \right\rangle;\ (34)\left\langle \begin{bmatrix} * & * & * \\ * & m & * \\ * & * & * \end{bmatrix} \rightarrow \Downarrow;\ J \rightarrow 7 \right\rangle;$$

$$(35)\left\langle \begin{bmatrix} * & * & * \\ * & q & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ J \rightarrow 7 \right\rangle;\ (36)\left\langle \begin{bmatrix} * & * & * \\ * & u & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ J \rightarrow D \right\rangle;$$

$$(37)\left\langle \begin{bmatrix} * & * & * \\ * & y & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ J \rightarrow 6 \right\rangle;\ (38)\left\langle \begin{bmatrix} * & * & * \\ * & 2 & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ J \rightarrow U \right\rangle;$$

$$(39)\left\langle \begin{bmatrix} * & * & * \\ * & a & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ K \rightarrow N \right\rangle;\ (40)\left\langle \begin{bmatrix} * & * & * \\ * & E & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ K \rightarrow 8 \right\rangle;$$

$$(41)\left\langle \begin{bmatrix} * & * & * \\ * & i & * \\ * & * & * \end{bmatrix} \rightarrow \Uparrow;\ K \rightarrow 6 \right\rangle;\ (42)\left\langle \begin{bmatrix} * & * & * \\ * & m & * \\ * & * & * \end{bmatrix} \rightarrow \Downarrow;\ K \rightarrow 7 \right\rangle;$$

$$(43)\left\langle \begin{bmatrix} * & * & * \\ * & q & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ K \rightarrow D \right\rangle;\ (44)\left\langle \begin{bmatrix} * & * & * \\ * & u & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ K \rightarrow 7 \right\rangle;$$

$$(45)\left\langle \begin{bmatrix} * & * & * \\ * & y & * \\ * & * & * \end{bmatrix} \rightarrow \Rightarrow;\ K \rightarrow U \right\rangle;\ (46)\left\langle \begin{bmatrix} * & * & * \\ * & 2 & * \\ * & * & * \end{bmatrix} \rightarrow \Leftarrow;\ K \rightarrow 6 \right\rangle;$$

We need one more program for "resetting" inertia. This is for the case when the slope of the terrain changes extremely. $(47)\ \langle X \rightarrow\ X;\ N \rightarrow e\rangle$;

If we run the obtained 2D P colony in the simulator, agents, which represent a unit volume of water, will begin to move around the environment. The number of agents located in one cell at one moment corresponds to the amount of water that at once flowed through the territory in one unit of time.

A source of water is placed into cells $[5,6],[5,7],[6,6],[6,7]$. In every source cell there are four agents. To simulate rain all agents are not active in the initial configuration. Only one agent has the configuration of Xe in each cell. The next three become active always in two computational steps. The numbers of active agents in the environment are shown in the Tables 5 - 13.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 5. The numbers of active agents in the initial configuration

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6. The numbers of active agents after 2 step of computation

The obtained results were compared with data that are listed in the master thesis [9]. This work is devoted to the simulation flow of liquid over the Earth's surface using cellular automata.

In work [9], the author devotes a great deal of time preparing data for cellular automaton, calculates not only the direction of flow but also the number and direction of outflows and inflows into the cell. After then cellular automaton starts

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7. The numbers of active agents after 4 step of computation

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8. The numbers of active agents after 6 step of computation

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Table 9. The numbers of active agents after 8 step of computation

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Table 10. The numbers of active agents after 10 step of computation

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Table 11. The numbers of active agents after 12 step of computation

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Table 12. The numbers of active agents after 14 step of computation

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 13. The numbers of active agents after 16 step of computation

working. The results are shown in Figure 1. Cells shown white are cells that contain water.

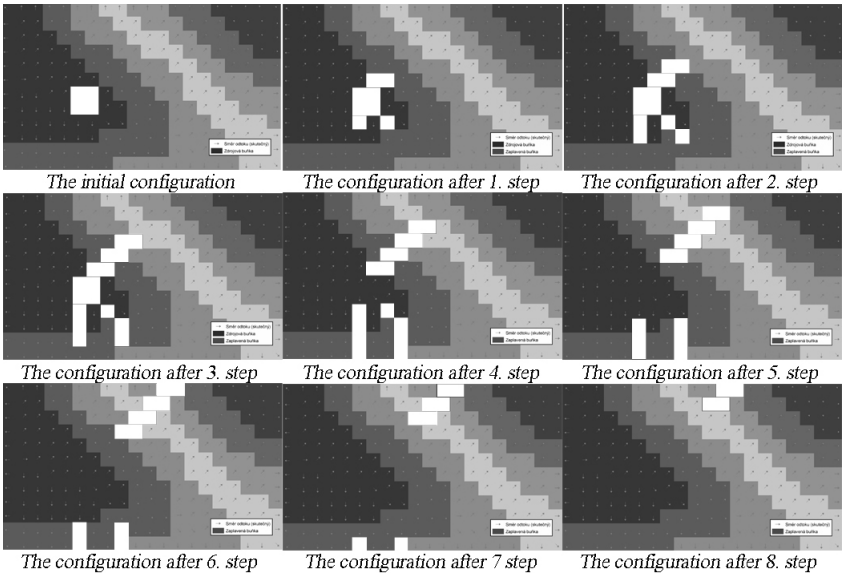

Fig. 1. Cellular automaton

If we compare the simulation process using 2D P colonies and using cellular automaton we obtain the following:

- Cells are flooded with water in the same time sequence.

- 2D P colony needs twice as many steps for the computation as cellular automaton.
- In 2D P colonies we do not need pre-treatment of data, output from the tool Flow direction is sufficient.

6 Conclusion and future work

The aim of this paper was to analyse the situation and to create a 2D model P colonies that would simulate the flow of liquid over the Earth's surface, a phenomenon called Surface runoff. This process is very common in nature and accumulation of water leads to flash flooding or floods in general. Flow down of water on the surface is influenced by many factors: the surface slope, soil saturation, temperature, humidity, size of source and lots of others. The first condition was partially met. Water flows down the surface in the right direction. In the case of places which are depressions, the possibility of overflow of the "tank" and the subsequent redistribution have not been implemented. The way of solving the problem is obvious and it is similar to absorption of water into the ground - certain amount of objects, which will represent the amount of water to be absorbed, needs to be added. Agents which have stopped, consumed the object and sets the direction of the flow using flow direction in neighbouring cells if it is possible.

Remark 1.
This work was partially supported by the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070), by SGS/7/2011 and by project OPVK no. CZ.1.07/2.2.00/28.0014.

References

1. Cienciala, L., Ciencialová, L., Perdek, M.: 2D P colonies. In Csuhaj-Varjú et al. (eds.). CMC 2012, Springer, LNCS 7762, 2013, pp. 161–172.
2. Csuhaj-Varjú, E., Kelemen, J., Kelemenová, A., Păun, Gh., Vaszil, G.: *Cells in environment: P colonies*, Multiple-valued Logic and Soft Computing, 12, 3-4, 2006, pp. 201–215.
3. Eastman, R. J.: *IDRISI Andes Guide to GIS and Image Processing*, Clerk Lab. Clerk University, Worcester, MA, USA, 2006.
4. Kelemen, J., Kelemenová, A.: *On P colonies, a biochemically inspired model of computation.* Proc. of the 6^{th} International Symposium of Hungarian Researchers on Computational Intelligence, Budapest TECH, Hungary, 2005, pp. 40–56.
5. Kelemen, J., Kelemenová, A., Păun, Gh.: *Preview of P colonies: A biochemically inspired computing model.* Workshop and Tutorial Proceedings, Ninth International Conference on the Simulation and Synthesis of Living Systems, ALIFE IX (M. Bedau at al., eds.) Boston, Mass., 2004, pp. 82–86.
6. Păun, Gh.: *Computing with membranes.* Journal of Computer and System Sciences 61, 2000, pp. 108–143.

7. Păun, Gh.: *Membrane computing: An introduction.* Springer-Verlag, Berlin, 2002.
8. Păun, Gh., Rozenberg, G., Salomaa, A.: *The Oxford Handbook of Membrane Computing*, Oxford University Press, 2009.
9. Pustějovská, L.: Implementation of a cellular automaton capable of simulating flow of liquid over terrestrial surface, master thesis (in czech), VŠB-Technical University of Ostrava, Czech Republic, 2008.
10. Torrents, B.: *Geosimulation.* John Wiley & Sons, 2004.
11. P systems web page: http://psystems.disco.unimib.it