Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/070605

Cellular Automata for the Flow Simulations on the Earth Surface, Optimization Computation Process

Juraj Cirbus[1,*] *and Michal Podhoranyi* [2]

[1] VSB - Technical University of Ostrava, Institute of Geoinformatics, 17. listopadu 15/2172, 708 33, Ostrava, Czech Republic
[2] IT4Innovations, VSB - Technical University of Ostrava, 17. listopadu 15/2172, 708 33 Ostrava, Czech Republic

Received: 10 Mar. 2013, Revised: 13 Jul. 2013, Accepted: 14 Jul. 2013
Published online: 1 Nov. 2013

Abstract: The spreading of water across terrain is not a simple and isolated process. The spread of water is a part of the more general water cycle. Many commercial software exist for water modelling, but their correct use requires a considerable knowledge of water modelling, and about the study area. Using the relatively simple principle of cellular automata (CA) it is possible to obtain results which compare well with real measurements. This article describes using a CA for simulating the spreading of liquid, using comparatively simple rules and conditions which include several factors affecting the spreading of water such as slope, roughness and infiltration. A disadvantage of CA is the slower calculation process, which is strongly dependent on the size of the study area. We address this issue by using three optimization methods to reduce the computation time.

Keywords: Cellular automata, spreading of liquid, hydrology, optimization, computing time, flow simulation.

1 Introduction

The spread of water across terrain is not an isolated process, nor one which can be expressed through the equations describing current hydrological models. It is a part of the complex rainfall-runoff process. The rainfall-runoff process is one of the fundamental aspects of hydrology and is a component of the hydrological cycle, influencing water resource management, landscape ecology, and the functioning of geosystems. Precipitation in various states is added to the water basin and initiates a series of processes, whereby water migrates via inanimate and living components of the hydrological system. The process is therefore a part of the river dynamics, and is dependent on the variability of water balance components such as precipitation and evapotranspiration. A general description of the rainfall-runoff process can be found e.g. in [1] or [9].

Over recent decades, attempts have been made to establish computerised representations of this complex system, and advances have been made in the understanding, and subsequent representation of, the laws and patterns associated with the aquatic component in the digital environment. In the last ten years considerable progress has been made in hydrological modelling, which tries to predict the behaviour of bodies of water. This contribution does not deal directly with existing hydrological models, but will outline some of the existing issues of cellular automata when used as a substitute for hydrological models.

Cellular automata (CA) can be understood as a mathematical idealization of a physical system, in which space and time are discrete [17]. Cellular automata have a relatively long history even though their application is not widespread. The original use of cellular automata was in biology as an idealization of the biological system [15, 16]. Other applications where cellular automata have been used are in non-linear chemical systems [5], model design for the development of spiral galaxies [4, 13], and in the modelling of intensification of dendritic crystals [7]. The cellular automaton developed for this paper will cover an application that affects the environment around us, namely the above-mentioned simulation of the water mass in the form of runoff water across the terrain. There are few scientific papers addressing the application of cellular automata to hydrology.

The primary aim of this work was to develop and implement CA able to predict the spread of water mass in the shortest possible simulation time. The computation

* Corresponding author e-mail: jcirbus@gmail.com

and simulation time in cellular automata is influenced by the size of raster, and on this basis the authors have developed three optimization techniques for decreasing the computation time. The cellular automata can take into account factors such as the spread velocity of the liquid, the roughness component of the terrain (Manning coefficient), and the slope ratios. By developing such a model we demonstrate that a CA approach can be a viable alternative to existing hydrological models.

2 Cellular Automata

2.1 The method

Cellular Automata were first introduced by J. Von Neumann and Stanislav Ulam in the 1940s, and they were widely used to simulate complex systems in 1970s [6]. Cellular automata are dynamical computational systems that are discrete in space and time, continuous state and whose behavior is specified completely by rules governing local relationships [2, 3, 17]. They are an attempt to simplify the often numerically intractable dynamic simulations into a set of simple rules that mirror intuition and that are easy to compute. As an approach to the modelling of emergent properties of complex systems, it has the great benefit of being visually informative of the progress of dynamic events. Since the early development by von Neumann (1966), a variety of applications ranging from gas phenomena to biological applications have been reported [18].

CA models are based on the use of several primary components. These components are: the cells arranged in a regular mosaic pattern (raster, grid); transition rules determining the changes in cell properties; the immediate neighbourhood of the cell; and boundary conditions. These components affect the status of each individual cell in a network in a given time step.

The neighbourhood of the CA cell is a group of cells surrounding an individual central cell, which defines an area of influence, and the state of the cell and its neighbourhood at time t affects the state of the cell at time $t+1$. If x_{ij} is the cell at position (i, j), then $S_{(}x_{ij})^{t}$ is the state of cell x_{ij} at time t, and $S_{(}x_{ij})^{(t+1)}$ will be the state at time $t+1$, and will be given by (1).

$$S_{(x_{ij})}^{(t+1)} = f(S_{x_{ij}}^{t}, S_{\Omega_{x_{ij}}}^{t}) \tag{1}$$

Here $\Omega_{x_{ij}}$ represents the set of cells in the neighborhood of cell x_{ij}, $S_{\Omega_{x_{ij}}}^{t}$ is the set of states of the cells $\Omega_{x_{ij}}$ at time t, and f is a function which is represented by sets of transition rules. The neighbourhood can be defined in various ways: the Von Neumann neighbourhood for instance (Fig. 1, left) consists of four adjacent cells in the east, west, north and south directions. In the field of hydrology it is more appropriate to use the Moore neighbourhood (Fig. 1, centre), in which the diagonal directions are added. Overall, the Moore's neighbourhood consists of 8 cells. The disadvantage of using the Neumann and Moore neighbourhoods is the varying distance of the individual centres of cells [10]. A more appropriate solution would be to use a hexagonal grid (Fig. 1, right), where all secondary cells have the same distance. This type of grid is not supported in the GIS databases however, and so in real use it would be necessary to convert the transformation from a regular grid into a hexagonal shape.

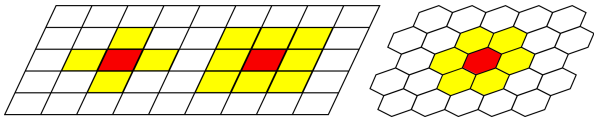


Fig. 1: Types of neighbourhood in the cellular automaton (CA) (Neumann neighbourhood on the left, Moore neighbourhood in the centre and hexagonal grid on the right)

This study combines a D8 algorithm (i.e. one using 8 neighbourhood cells) and cellular automata to model water flow on a DEM surface. Water from a given cell can flow in one of eight output directions relating to the eight adjacent cells. This approach is commonly referred to as the eight directions (D8) flow model and follows an approach presented in [19]. Each major cell, (i, j), has eight tessellated neighbours $(i-1, j-1)$, $(i-1, j)$, $(i-1, j+1)$, $(i, j-1)$, $(i, j+1)$, $(i+1, j-1)$, $(i+1, j)$, $(i+1, j+1)$ which constitute a Moore neighbourhood.

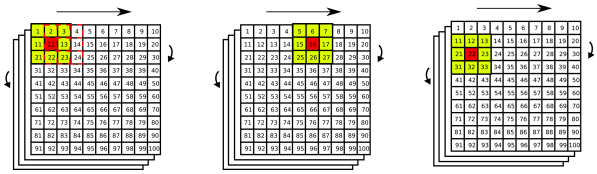


Fig. 2: Roving window moving on raster

The simulation process is divided into iterations, where one iteration is considered as the transition of a roving window through the entire grid (Fig. 2), and updates the status of each of the cell from the first to the last cell located in the grid. At each location of the roving window, the CA rules are used to deduce a new value for the central cell based on the cell values in the central cell and neighbouring cells The cell values include information related to the water-transport properties of the cell and are described later in section the rules and condition. The new value for the central cell is stored in a temporary grid, and at the end of each iteration all the

cells in the grid are updated with the contents of the temporary grid.

2.2 The rules and conditions

The key component of CA is the set of transition rules which define how the central cell changes state. The change of state is dependent on the immediate neighbourhood.

A set of hydrological rules and conditions has been developed here to simulate the spread of liquid. The rules are based on the physical processes of the spread of liquids and are appropriately modified to simulate the real world under CA conditions.

In developing the CA, a set of transition rules was created for simulating the spreading of liquid in the terrain. The rules are applied in each step of a roving window. The CA algorithm can be divided into three stages: **pre-processing input data; application of transition rules; post-processing and generation of output layers**.

In the pre-processing part, four grid layers are generated from the DEM - a water level grid, flooded cell grid, direction grid, and a slope grid. These layers are used in the transition rules stage. Each cell in the **water level grid** contains the height (m) of the water column in that cell. On initialization, only cells containing water sources have non-zero values of water column. The **flooded cell grid** contains a binary flag indicating whether or not a cell has met flooding conditions. If this grid value is true for a given cell, then the cell has been flooded and water can be transferred from this cell to neighbouring cells. Each cell in the **direction grid** stores the preferred flow direction from that cell, encoded as one of the values 1, 2, 4, 8, 16, 32, 64 and 128 (Fig. 3, upper left). The flow direction is determined by considering the DEM values in the eight neighbouring cells. The neighbouring cell with the lowest DEM value is taken as the flow direction, and the cell's location relative to the central cell is encoded and allocated to the central cell in the direction layer. In the case of more than one neighbouring cell sharing the lowest DEM value, the resulting multi-flow directions are encoded as a sum of directions (Fig. 3). The resulting direction layer is a grid layer where each cell holds an integer in the range 1 to 255.

The **slope grid** is generated from the DEM, and contains a calculation of the slope across the neighbourhood of each central cell. The slope of the terrain is calculated by using the algorithm described by [14]. The rate of change (delta) of the surface in the horizontal (dz/dx) and vertical (dz/dy) directions from the central cell determines the slope. The algorithm calculates the slope as (2),

$$tg(slope) = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \qquad (2)$$

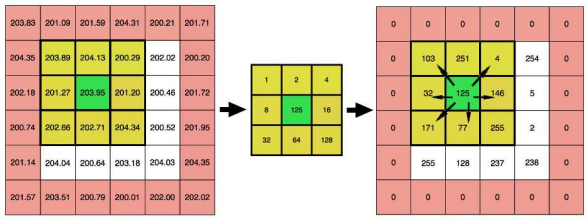


Fig. 3: Dividing flow direction

where $\partial f/\partial x$, $\partial f/\partial y$ are partial derivatives of a function of two variables $f(x,y)$. With 8 neighbourhood cells, the derivatives are given in equations (3) and (4).

$$\frac{\partial f}{\partial x} = \frac{(h_{1,3} + h_{2,3} + h_{3,3} - h_{1,1} - h_{2,1} - h_{3,1})}{6d} \qquad (3)$$

$$\frac{\partial f}{\partial y} = \frac{(h_{3,1} + h_{3,2} + h_{3,3} - h_{1,1} - h_{1,2} - h_{1,3})}{6d} \qquad (4)$$

Here h are the contents of the DEM in each of the neighbouring cells. The result is a layer in which each cell contains a floating point representation of the value of slope in degrees. The roving window (Fig. 2) moves cell by cell over the input **DEM** layer, and uses information from the flooded cell grid to check if the cell(s) around the middle cell is/are flooded. If the CA finds flooded cell(s) in neighborhood, the water direction is checked from the direction grid, and the flooded cell grid is checked to determine if water from the neighbouring cell can be distributed to the central cell. If these conditions are met, the CA use the information from the slope layer and flooded cell grid to calculate an amount of water, which will be transferred to the middle cell. The information about the amount of water which will be moved from neighborhood is stored in a **temporary water-level layer**. If the roving window detects water in the central cell, the CA calculates in the same manner an amount of water which will be distributed to the cell(s) in neighborhood. The information about water loss is stored in a **temporary drain layer**. At the end of each iteration is information from the temporary layers is summed and added to the flooded cell grid, and the temporary layers are reset. This process and application of the transition rules can be described as the set of if-then conditions. The algorithm is (Fig. 4).

The CA rules indicate how a certain amount of liquid is transferred, and define the amount of liquid loss as a result of the infiltration process. The rules for the spreading of liquids are based on the use of the Manning coefficient, where the fluid velocity v in each cell in which the liquid is present is calculated according to the following equation (5).

$$v = \frac{\sqrt[3]{depth^2}.\sqrt{s}}{n} \qquad (5)$$

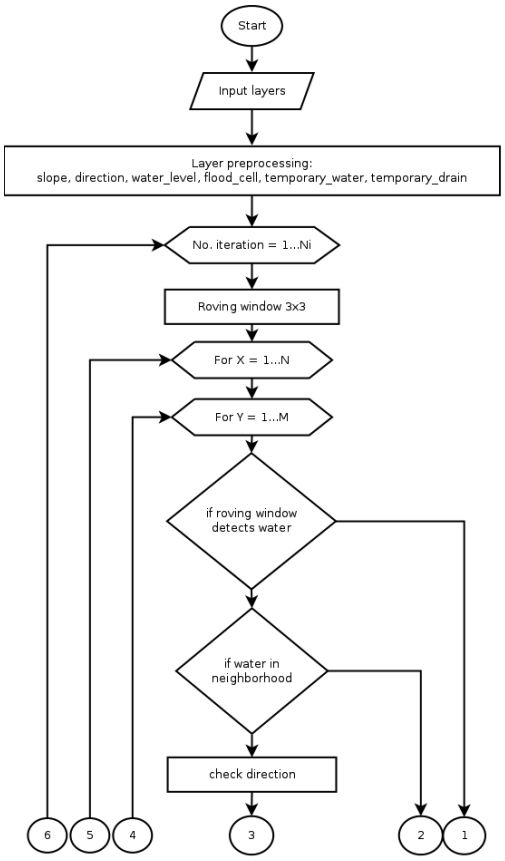


Fig. 4: (a) CA algorithm, chart diagram

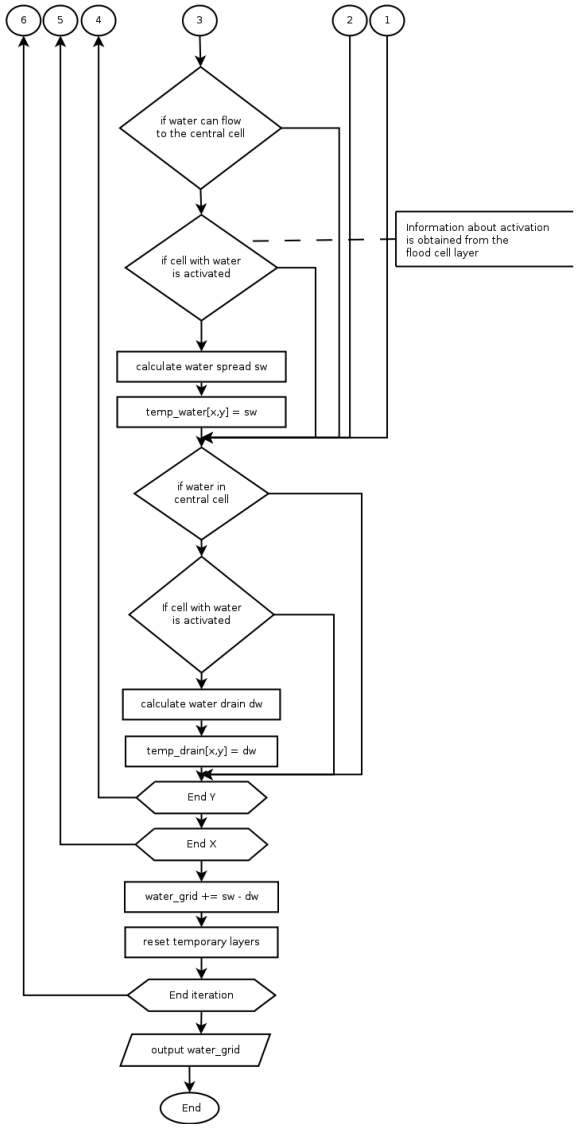


Fig. 4: (b) CA algorithm, chart diagram

Here $depth[m]$ is the height of the liquid column in the cell, $s[-]$ is the slope of the terrain derived from the digital elevation model, and $n[s.m^{1/3}]$ is the Manning roughness coefficient. The Manning roughness coefficient n accounts for the effect of bed roughness on the flow field, and its determination is essential to the accuracy of the calculated flow, sediment transport, and bed change. [12] The velocity is calculated using relation (5) and is used to determine the time (6) needed to move the liquid through the cell.

$$T = \frac{width}{v} \tag{6}$$

Here $width[m]$ is the cell width and $v[m.s-1]$ is the velocity of the liquid. After meeting the conditions for flooding the cell (liquid transfer), the liquid can spread to cells in its neighbourhood. A cell is considered flooded when the time required for water to move through the cell (7) is more than the iteration time t $(T > t)$. The relation between space and time discretization must respect this condition, if is not happened the CA reduce the time discretization. The time step for the one iteration is initially set to 1s. If the $(T > t)$ in the cell is violated, the CA iteration is stopped, the iteration time is reduced by one half, and the iteration process continues with the reduced time-step.The volume of liquid entering to the cell is calculated as (7).

$$W = depth.width.v.t \tag{7}$$

Here W is volume of water, $depth$ is water column, v is velocity of water and t is time interval of one iteration in CA. The time interval of the one iteration is usually 1s. One iteration can be understood as moving of roving window from the first cell on the grid to the last one on the grid.

The rule for water loss (water moving from the middle cell to the next cell) depends on the surrounding cells. The velocity of infiltration f (8) is calculate using the Green-and-Ampt equation. Green-and-Ampt's model applies Darcys law and the principle of conservation of

mass, but in a finite-difference formulation that allows a more holistic view of the infiltration process [8].

$$f = K_s \left(1 + \frac{H_f M_d}{F}\right) \tag{8}$$

Here $K_s[mm.h^{-1}]$ represented hydraulic conductivity, $H_f[mm]$ is suction lift, $M_d[mm]$ is a soil moisture deficit and $F[mm]$ is total depth of infiltration.

The last key components of a CA are the boundary conditions. Boundary conditions describe the behaviour of liquid at outermost cells of the grid. Boundary conditions can be divided into three categories:

- **Periodic:** Water leaving the grid at one side will re-enter the grid on the opposite side.
- **Reflective:** A wall is created around the grid. Water cannot leave the grid and is instead accumulated in the modelled area.
- **Absorbing:** Water leaving the grid at any edge is removed from the model.

Here we use absorbing boundary conditions, and water is removed from the modelled area.

The last part of algorithm involves post-processing of the results. The results grid indicates the location of water in the terrain, with each cell containing the water column level in that cell. The results grid is converted into a TIFF image indicating the flooded cells in the terrain.

3 Simulation water flow

3.1 Study Area

The study area is part of the river Ostravica. Digital elevation model data have been used for the simulation including images created by photogrammetry in 2007. The result of photogrammetrical processing was a DEM raster with resolution 10 m. The image resolution is 10 m and the study area is 1500 by 1500 m. The watershed of Ostravica (Fig. 5 left) includes several measurement stations for detecting flow rate and water level. This selected study area is shown in Fig. 5 (right) and was chosen because of the occurrence of two measurement stations (indicated by red dots) close together in a relatively small area. The first measurement station is located 23 km from the confluence and the second on is located at 24.2 km distance.

3.2 Input data

Input to the model can be divided into two categories: grid layers, and model parameters. The primary input is a grid layer in form of digital elevation model (DEM) of the area of interest. The individual cells forming the DEM contain elevation information and are used to determine the slope and the general direction of the spread of the liquid.

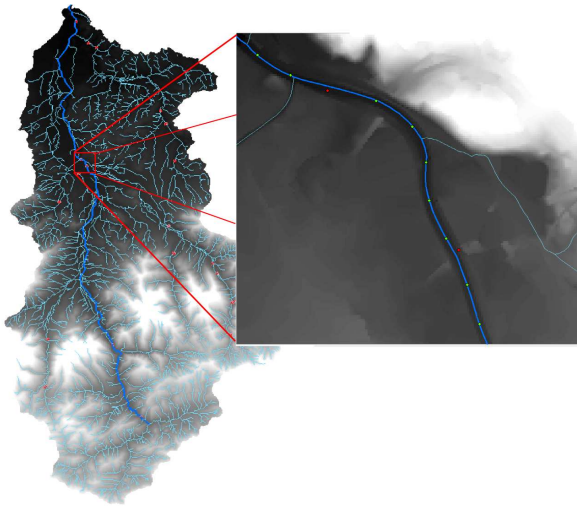


Fig. 5: Study area

The area represented by the DEM should be optimized by minimizing the number of cells, hence reducing the number of necessary calculations and the resultant computation time.

The use of layer of DEM is sufficient to run the calculation process, but results may deviate considerably from real measurements. Using additional layers can increase the accuracy. Secondary layers which can be included in CA water simulations include land cover, distribution of the types of soil, or the infiltration.

Land cover, expressed in the form of surface roughness and infiltration, influences the rate of spread of liquid. In a CA model the components of infiltration and roughness of the terrain can be represented in the form of a grid, where each cell can have its own infiltration and roughness values, or can be replaced with a constant value over the entire area of interest. Areas of interest can be subdivided into smaller areas where the roughness of each area is represented as a single value reflecting the prevailing land cover in each area.

A hydraulic roughness coefficient is necessary to calculate the liquid spread velocities. One option here is to use a single Manning coefficient for the entire area of interest. This approach is suitable for areas with fairly homogeneous land cover. Alternatively, it is possible to include representation of all types of cover in the layer of surface roughness, as captured in the area of interest. Typical values of Manning coefficient are 0.011 for rocky soil and 0.2 for dense vegetation.

Numerical values form an additional component of the CA. They determine the duration of the entire simulation, and a time step controlling the creation of the output layers. Creating such output layers allows a continuous monitoring of the model during a simulation. This is useful in cases where a long simulation period is set, which might last several tens of minutes.

3.3 Configuration of the simulation

The basic functionality of the designed CA is modelling runoff of liquid across the land surface. The CA is trying to simulate in the most realistic manner the spread of the liquid through a set of simple mathematical and physical rules. The total amount of liquid water living the region can be called the hydrologic circulation, or more commonly, the runoff. The amount of liquid water actually produced in the region is called hydrologic production [8]. For the purpose of CA it is necessary to generalize these factors.

The input DEM layer used here is that representing part of river Ostravice. The source of water is inserted into the riverbed, where the water column was calculated from measured discharge value in the terrain. The value of water column is set to 0.7 m with a width of 30 m, represented in the raster layer by 3 cells (each cell of size 10x10 m). A single value of Manning roughness coefficient is used for the whole area, and is set to $n = 0.04$. The value 0.04 is often used for water flow in a riverbed. The infiltration rate is set to zero, and rainfall is not considered here. The CA simulated 6 hours of water flow, and output rasters were produced at one-minute intervals.

3.4 Comparison of simulation results

The results of the simulation are shown in Fig. 6. Here it is possible to see gauging stations (red dots in Fig. 6c). Measurements from these stations are compared with the results of the CA simulation. Between the measurement stations a series of manual measurements were made every 200 m along the river (orange dots in Fig 6c) using a surveying rod. The measurement data correspond to a discharge $Q = 154m^3/s$ and are shown in Tab. 1. The CA simulated the same situation using a DEM and a source layer to allocate cells with initial water values. After each time step the cells in the raster contain information about the water level. The calculation time for the selected area was 2 h and 50 minutes. The simulated water levels in the final raster output layer are compared with the measured values and are shown in Tab. 1.

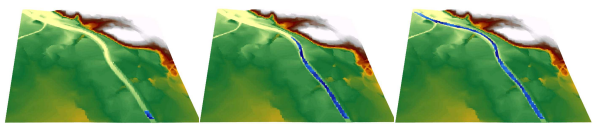


Fig. 6: Study area

The same trend can be seen in the simulated and measured values (Fig. 7), and the correlation coefficient here is 0.556. The most widely used correlation

Table 1: Results of simulations of the liquid runoff for defined DEMs

No. measurement	Station [km]	Real Value [m]	Simulate Value [m]
1	22.60	0.79	0.62
2	22.80	0.77	0.60
3	23.00	0.70	0.54
4	23.20	0.75	0.67
5	23.40	0.87	0.83
6	23.60	0.77	0.74
7	23.80	0.65	0.64
8	24.00	0.75	0.73
9	24.07	0.67	0.67
10	24.20	0.65	0.64
11	24.40	0.67	0.66

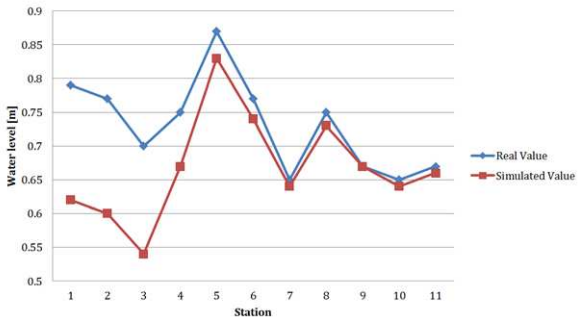


Fig. 7: Comparison of measurements and simulated values

coefficient is the Pearson Product-Moment Correlation, often called the Pearson r, which was used for comparing two values (real and simulated). The Pearson r coefficient is calculated as (9),

$$r = \frac{N(\sum XY) - (\sum X)(\sum Y)}{\sqrt{(N(\sum X^2) - (\sum X)^2)(N(\sum Y^2) - (\sum Y)^2)}} \qquad (9)$$

where X and Y are sets of input data. The simulated water levels at the downstream stations, however, are considerably lower than measured values. This discrepancy is due to two tributaries which flow into the river Ostravica. Omission of these two tributaries has resulted in the reduction of the water level in the downstream part of river in the study area.

The river Ostravica has 2 tributaries in our area of interest(Fig. 8). The first tributary coming from the right side is a stream with lower water level and discharge and the second tributary from the left side is a stream with higher values of water level and discharge. The simulation was repeated with the addition of two tributaries as new sources of water. The first tributary from the right side was modelled with an input water level of 0.20 m and width 10 m (1 cell), and the second tributary was modelled with water level 0.30 m and width

Fig. 8: Part of river Ostravica with 2 tributaries

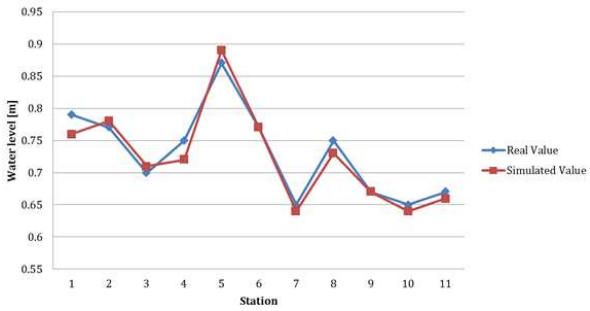


Fig. 10: Comparison of measurements and simulated values with extra tributaries

20 m (2 cells). The results are shown in Fig. 9 and values can be found in Tab. 2 and in Fig. 10.

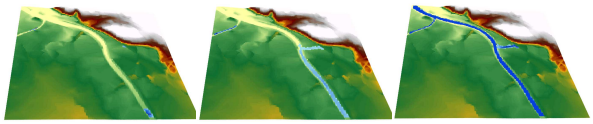


Fig. 9: Results from CA with 2 tributaries

The addition of the two tributaries results in a better match between simulated and measured values, with a correlation coefficient of 0.976.

Table 2: Results of simulations of the liquid runoff with extra tributaries for defined DEMs

No. measurement	Station [km]	Real Value [m]	Simulate Value [m]
1	22.60	0.79	0.76
2	22.80	0.77	0.78
3	23.00	0.70	0.71
4	23.20	0.75	0.72
5	23.40	0.87	0.89
6	23.60	0.77	0.77
7	23.80	0.65	0.64
8	24.00	0.75	0.73
9	24.07	0.67	0.67
10	24.20	0.65	0.64
11	24.40	0.67	0.66

4 Optimization calculation process

The CA described here was developed in the Python language and was run on a high-end computer machine with 2.2 quad core i7 processor and 4 GB RAM. 2 hours of real time flow was simulated in 2 hours and 50 minutes machine time. Implementation in other high level languages such as C++ could result in improved performance at the expense of increased development time. Here we focus on improving the simulation time by implementing several optimisations. The optimisations described here are applied to the same data-sets with identical CA rules, and are run on the same PC hardware, and result in decreased calculation times.

In case of using a 100x100 raster (100 rows and 100 columns), the roving window during one iteration inspects each cell, resulting in 10 000 applications of the CA rules. The computing time can be improved by reducing the number of cells which are inspected, and by applying multi-processing techniques. Three optimisation techniques were developed and investigated here:

- Optimization of the elevation stage
- Optimization of the watershed stage
- Raster caching and multiprocessing.

4.1 Elevation optimization

The CA rules described above check each cell in the input layer and examine its neighbouring cells. This process is very lengthy, and is applied even to cells where it is known that no water can exist because the elevation of the cell is higher several times higher than that of the source water cell. This results in unnecessarily large computation times. To avoid this, the elevation of each cell is compared with the elevation of the source cell(s) in a pre-processing step.

In this step, a mask is created in which cells with an elevation greater than the source elevation level are marked as NoData. This mask is applied to the DEM layer before the CA simulation begins. In the subsequent CA algorithm, the roving window ignores any cells marked as ”NoData”, effectively disabling any cells with too high an elevation. Fig. 11 shows a digital elevation layer after the application of the elevation mask, with white areas indicating NoData. For this simulation it is necessary to have a digital elevation layer without sinks.

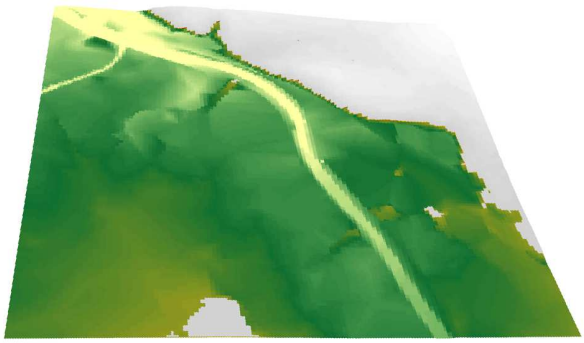


Fig. 11: Elevation optimization

In the table below (Tab. 3), results time obtained with and without the elevation optimization are compared.

Table 3: Comparison of results with and without elevation optimization

	Number of input cells	Computing time for 1 iteration [s]
Without opt.	22 500	1.25
With opt.	17 249	0.96

4.2 Watershed optimization

A watershed (also called drainage basin, river basin or catchment) is defined as the area that topographically appears to contribute all the water that passes through a given cross section of a stream. The horizontal projection of the watershed is called the drainage area of the stream at the cross section [8].

The watershed optimization involves restricting the CA simulation to use only cells falling within a given drainage area. The definition of the drainage area is based on Conrad's algorithm in which the DEM layer is separated into different regions of interest along boundaries of equal elevation. Each region has a different color and a different identity number (ID). The identity number is from the Pfafstetter Coding System which assigns IDs based on the topology of the land surface [11]. Each watershed is assigned a specific Pfafstetter Code based on its location within the overall drainage system (Fig. 12a), and on the total drainage area upstream of the watersheds outlet. The drainage region in which the source cell is located is selected, and all cells in this region with an elevation lower than the source elevation are used to create a mask. Cells not meeting these criteria are marked as "NoData" and are omitted from the CA simulation (Fig. 12b).

Applying the watershed optimization not only reduces the computation time (Tab. 4) but also restricts the simulation to an area which better reflects the real area of interest.

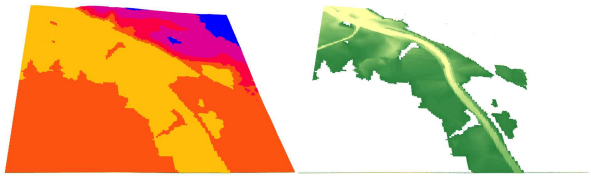


Fig. 12: Watershed optimization (12a: watershed, 12b: terrain with watershed mask)

Table 4: Comparison results with and without watershed optimization

	Number of input cells	Computing time for 1 iteration [s]
Without opt.	22 500	1.25
With opt.	4 838	0.27

4.3 Multiprocessing optimization

Multiprocessing is a method of computation in which different parts of a task are distributed between two or more similar central processing units, allowing the computer to complete operations more quickly and to handle larger, more complex procedures.

The multiprocessing optimization applied here to CA involves dividing the raster layer into several clusters, where the number of clusters can be changing dynamically according to the number of CPU cores available in the computer. Fig. 13 illustrates a raster layer divided into two and four separate clusters. In each iteration of the CA, the raster layers are divided into clusters, and the computations for each cluster are allocated to one core. At the end of the each iteration the clusters are joined back together, and this forms the basis for the input of another iteration.

In the programming implementation, the calculations for each cluster are performed in a separate thread. Fig. 13 also shows the initial position of the roving windows. Orange coloured cells indicate areas of overlap. Using this multiprocessing optimisation resulted in an increase in speed of 30% using two threads and 56% using four threads (Tab. 5).

5 Conclusion and discussion

The primary purpose of this study was to demonstrate the feasibility of using cellular automata for liquid runoff simulations in terrain. The study showed that even such a

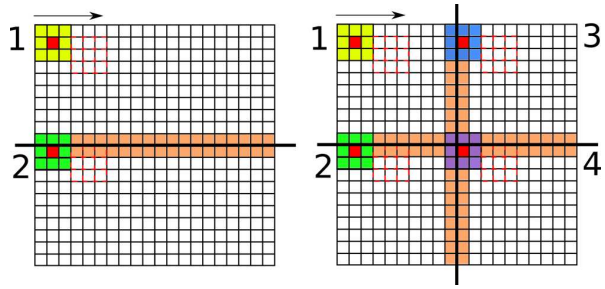


Fig. 13: Multiprocessing, start position for rowing windows

Table 5: Results with multiprocessing optimization

	Number of input cells	Computing time for 60 iterations [s]
1 thread	22 500	125.0
2 threads	22 500	87.5
3 threads	22 500	55.0

complex natural process as water runoff can be, to some extent, imitated only with the assistance of a digital elevation model and related raster layers, and has demonstrated that the grid cell can be a carrier of information which can identify some of the natural processes.

The cellular automaton in this article is the initial stage of the design of a more complex cellular automaton which could be used to model different kinds of liquid e.g. oil. The currently designed cellular automaton takes into account, in particular, the slope of the terrain, the roughness of the surface, the velocity of the flowing liquid, and the confluence of two liquids. A comparison of model output with real measurements showed good agreement, demonstrating that, despite the simplicity of the model, a CA approach can provide realistic results for a complex natural process like liquid runoff.

In the future it is intended to test the model further by obtaining data describing a longer runoff episode, and including precipitation. It is also desirable to study the effects of the infiltration process, which in this study has been neglected.

The authors designed three optimization strategies for the CA, each of which resulted in improved computation times. Each of the optimizations were applied individually here, although in principle all three methods could be applied at the same time. This would help to reduce computation times further, or to allow more detailed studies using reduced pixel sizes.

A further topic for development will be the development of CA rules for the distribution of water in flat areas. Here the prediction of transport direction is more problematic and needs a more careful approach than that which is described in this article.

6 Acknowledgement

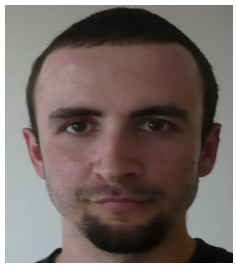
This article has been elaborated in the framework of the IT4Innovations Centre of Excel-lence project, reg. no. *CZ*.1.05/1.1.00/02.0070 supported by the Operational Programme 'Research and Development for Innovations' funded by the Structural Funds of the European Union and the state budget of the Czech Republic. The authors are grateful to all the collaborators which participated in the development of the model, P. Rapant coordinated the project and special thanks for R. Watson, which criticisms improve the original manuscript.
The authors are grateful to the anonymous referee for a careful checking of the details and for helpful comments that improved this paper.

References

[1] P. B. Bedient, W. C. Huber, B. C. Vieux, Hydrology and Floodplain Analysis, fourth ed., Prentice Hall, London, (2007).
[2] C. R. Calidonna, A. Naddeo, G. A. Trunfio, S. Di Gregorio, From classical infinite space-time CA to a hybrid CA model for natural sciences modeling, Appl. Math. Comput., **218**, 8137-8150 (2012).
[3] G. Cattaneo, M. Comito, D. Bianucci, Sand piles: From physics to cellular automata models, Theor. Comput. Sci., **436**, 35-53 (2012).
[4] H. Gerola, P. E. Seiden, Stochastic star formation and spiral structure of galaxies, Astrophys. J., **223**, 129-139 (1978).
[5] J. M. Greenberg, B. D. Hassard, S. P. Hastings, Pattern formation and periodic structures in systems modelled by refraction-diffusion equations, Bull. Am. Math. Soc., **84**, 1296-1327 (1978).
[6] J. Kari, Universal pattern generation by cellular automata, Theor. Comput. Sci., **429**, 180-184 (2012).
[7] J. S. Langer, Instabilities and pattern formation in crystal growth, Rev. Mod. Phys., **52**, 1-28 (1980).
[8] S. Dingman, Physical Hydrology, first ed., Englewood Cliffs, New Jersey, (1994).
[9] D. R. Maidment, Handbook of Hydrology, first ed., McGraw-Hill Professional, London, (1993).
[10] J. Parson, M. Fonstad, A cellular automata model of surface water flow, Hydrol. Processes, **21**, 2189-2195 (2007).
[11] O. Pfafstetter, Classification of hydrographic basin: coding methodology, Dep. Nac. De Obras e Saneamento, Rio de Janeiro, (1989).
[12] W. Wu, Computational River Dynamics, Taylor and Francis, London, (2008).
[13] P. F. Schewe, Galaxies the Game of Life, and Percolation, in Physics News, Amer. Inst. Phys. Pub., **R-302**, 61 (1981).
[14] P. A. Burrough, R. A. McDonnell, Principles of Geographical Information System, Oxford University Press, Oxford, (1998).
[15] J. Von Neumann, The general and logical theory of automata, in: J. Von Neumann, A.H.Taub (Eds.), Von Neumann Collected Works, Pergamon, 288 (1963).
[16] J. Von Neumann, Theory of Self-Reproducing Automata, in: A.W.Burks (Eds.), University of Illinois Press, Urbana London, (1966).

[17] S. Wolfram, Cellular automata as models of complexity, Nature, **311**, 419-424 (1984).
[18] G. B. Ermentrout, L. Edlestein-Keshet, Cellular automata approaches to biological modelling, J. Theor. Biol., **160**, 97-133 (1993).
[19] S. K. Jenson, J. O. Domingue, Extracting Topographic Structure form Digital Elevation Data for Geographic Information system Analysis, Photogramm. Eng. Rem. S., **54**, 1593-1600 (1988).

Juraj Cirbus
is a PhD candidate in Geoinformatics at the Faculty of Mining and Geology at the VSB-Technical University of Ostrava, where he also received his BSc and MSc in Geoinformatics. His research is currently focused on developing software for simulation spreading of liquid on the terrain based on the principle cellular automat.



Michal Podhoranyi
is a PhD student in Geoinformatics at the Faculty of Mining and Geology at the VSB-Technical University of Ostrava, where he also received his MSc in Geoinformatics in 2008. He also worked for the IT4Innovations. His research is currently focused on hydraulic modelling (mainly HEC-RAS, MIKE 11), floods, evaluation of digital elevation data, flow simulations and GIS analysis.