CLOUD IMPLEMENTATION OF AGENT-BASED SIMULATION MODEL IN EVACUATION SCENARIOS

Andrzej Wilczyński
Cracow University of Technology
Warszawska 24, 31-155 Cracow, Poland
AGH University of Science and Technology
al. Mickiewicza 30, 30-059 Cracow, Poland
E-mail: and.wilczynski@gmail.com

Joanna Kołodziej
Cracow University of Technology
Warszawska 24, 31-155 Cracow, Poland
E-mail: jokolodziej@pk.edu.pl

KEYWORDS
Computational Cloud; Multi-Agent System; Evacuation System; Modelling and Simulation; MapReduce; Hadoop.

ABSTRACT

Over the years evacuation simulation has become increasingly important in the research on the wide class of problems related to the public security in emergency situations. In this paper we develop simulation platform fully integrated with the cloud systemwith using the MapReduce programing model and Hadoop framework.The environment illustrating evacuation scenarios and actors is modelled. by cell automata and interpreted as a potential field, in which technologies the generated agents are located with using multi-agent. The simulation is executed as a stream-based data-processing operation to enable environmental universality while taking advantage of the MapReduce model. Several test cases are provided to show the efficiency of the simulation platform.

INTRODUCTION

The management of emergency activities such as guiding people out of dangerous areas and coordinatingrescue teams is characterized by uncertainty regarding both the source of danger and the availability ofuseful resources. Depending upon the scale and nature of the incident, people involved in a crisis maysuffer from limited situational awareness (SA) [34]. SA involves being aware of what is happening around inorder to understand how information, events, and the crowd actions will impact the goals and objectives.

Lacking or inadequate SA in emergency situations has been identified as one of the primary factorsleading to human error, with potentially grave consequences. However, emergency situations are chaotic in nature and any incident management system typically encompasses multiple sources of information such as mobile devices from affected people, social network feeds and various on-site sensors. All of these sources are available in different formats, present indifferent locations and reliable at different confidence levels. As sources are dispersed throughout geographical areas, SA becomes a complex, distributed processing problem where innovative techniques need to be employed in order to efficiently use information sources in real-time. In many cases it is impossible to provide an effective crowd management in evacuation by extracting data generated in real-world evacuation [1, 2] and realistic experiments [3, 4], mainly because of large data volumes and incompleteness. In such cases, simulation can be the proper solution for such problems.

Modeling and simulation methods designed for evacuation systems can be divided into two main categories: (i) flow-based approaches and (ii) individual-based approaches. In flow-based approaches [5] the behavior of individuals in the crowd is ignored, while in the second category [6] the crowd is defined as a collection of individuals, which are either entity-based or agent-based with autonomous intelligent agents.

Agent-based approaches typically focus on defining the rule of an individual's behavior and then applies the rule to all individuals of the whole simulated crowd [7-11]. However, the complexity of MAS in big crowd simulations is high, which makes the whole model ineffective in the case of short-time decisions processes such as crowd management in emergency scenarios. Therefore, typical MAS used for crowd modeling has been supported additionally by using various multi-CPU systems [12], cluster [13] grid technologies [14-16], Graphic Processing Unit (GPUs) [17] and Field Programmable Gate Array (FPGAs) [18].

Cloud computing entails the exchange of computer and data resources across global networks; itconstitutes a new value-added paradigm for network computing, where higher efficiency, massive scalability and speed rely on effective software development [19-20]. Cloud computing is rapidly becoming a popular infrastructure of choice among all types of organisations. Despite some initial security concerns and technical issues, an increasing number of institutions are considering moving their applications and services into "The Cloud." Recently, cloud-based technologies have been successfully used as effective support tools in complex simulations in emergency situations [21-23]. MapReduce programing model [24-26] is very popular tool for implementation of the crowd simulation in evacuation situations as two-stage *mapper-reducer* process. In *mapper* phase, the input data is extended

Proceedings 30th European Conference on Modelling and Simulation ©ECMS Thorsten Claus, Frank Herrmann, Michael Manitz, Oliver Rose (Editors)
ISBN: 978-0-9932440-2-5 / ISBN: 978-0-9932440-3-2 (CD)

into a large intermediate table, while in the *reducer* phase table is reduced in order to generate the output data. The MapReduce procedure can be iterated many times until specified stopping criteria (usually the expected output data).

There are not many successful examples of integration of evacuation MAS with cloud environments. In most of such approaches, MAS-based simulation is interpreted as a simple data processing operation. In [27], in the loop-based simulation process, agents have been ordered and processed as a queue in the loop. The agent queues have been implemented as big tables of various sizes generated during the simulation. In this paper, we develop OpenStack [28] cloud-based simulation platform using the MapReduce programing modelto support a large-scale evacuation crowd simulation. Multi-agent technology was employed to provide the crowd simulation and a grid-based model was used to provide environmental information. The simulation results show benefits of using the cloud-support in evacuation in restricted indoor environments compared to traditional IT support used in many realistic scenarios.

The paper is organized as follows first we provide a short description of the environmental model and cloud-based simulation platform. Next, we define the MapReduce-based multi-agent simulation model and specify the output data requests and generation. Then we present the results of simple experiment. After that we define draft conclusions and plans for future work in this domain.

OPEN STACK CLOUD-BASED SIMULATION PLATFORM FOR CROWD EVACUATIONS

We designed and implemented our cloud-based simulation platform by using the cloud OpenStack technology [28] and MapReduce programming model [24-26] with Hadoop framework [29]. The platform model archtecture is presented in Fig. 1. The model is composed of three mainmodules, namely (i) a generic environment model, (ii) a multi-agent based simulation module and a data requestor.

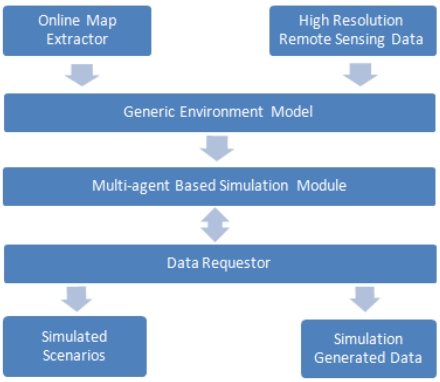

Figure 1. The Architecture of the Simulation Platform

The generic environment model is a formal model of the virtual environment used for simulation of the crowd evacuation scenarios. This model is based on the potential field paradigm and containts a vector field-based pathfinding mechanism. In the environmental model, various types of spatial data from multiple data sources (e.g., high-resolution remote sensing data or online map services data) are collected, merged and converted into streams and multi-streams for the further.

The multi-agent simulation module is used for simulations of the crowd behavior and decisions in emergency situations. It is based on the MapReduce architecture and runs on a Hadoop cluster. Most of the agent's operations are transformed into simple data access and processing operations. Hadoop Distributed File System (HDFS) is a standard technology used for data and information transfers.

GENERIC ENVIRONMENTAL MODEL

The generic environment model in this approach is based on environment models with cell automata developed in [30, 31].The main component of this model is *an environmental map* defined as a grid in which each cell represents a small square area in physical environment. We specify three types of cells, namely: (i) accessible cells, (ii) blocked cells and (iii) exit cells. An accessible cell represents an area which may be accessed by the individual (human, agent), where blocked cell is defined as an area with obstacles. In the simulation, individuals are able to pass through accessible cells but not blocked cells. The environmental map can be generated from satellite photos (sensing data) or online web maps (e.g., Google Map).

The second component of the environmental model is a position potential field [32] for the management of the agent's position on the environmental map. A potential field module calculates a distance to the nearest exit cell from the specified (current) agent's position. Then it is responsible for the generation of the optimal path for the agent from his current position to the exit.

The position potential field is usually defined as a discrete 3D function with values numerically represented as a matrix of the same size as environmental map. Each parameter in that matrix is interpreted as the cell's 'position potential'. In particular, the position potential of obstacle cells is -2, which means that the corresponding cells are inaccessible. Obviously, all exit cells have a position potential of 0, which is the lowest valid value.

With the potential field, the general environment model is able to provide path information for the simulation, which can eliminate the need for pathfinding algorithms. In fact, in our agent implementation, routing

is only a search operation in a small list, which can be much faster than ordinary pathfinding algorithms are.

MAP-REDUCE BASED SIMULATION

The crowd behavior in evacuation scenarios is modelled by the multi-agent system (MAS). MAS in our approach is fully integrated with the SaaS (Software-as-a-Service) cloud layer and MapReduce model. The architecture of the multi-agent simulation module is shown in Fig. 2.

Crowd simulation is performed continuously as a phase loops (steps). Each loop corresponds to one time-step and contains one map operation and one reduce operation. After each loop, a position (location) of each agent is updated. It means that a serial cell selection operation is performed and the simulated scene of this time-step is produced.

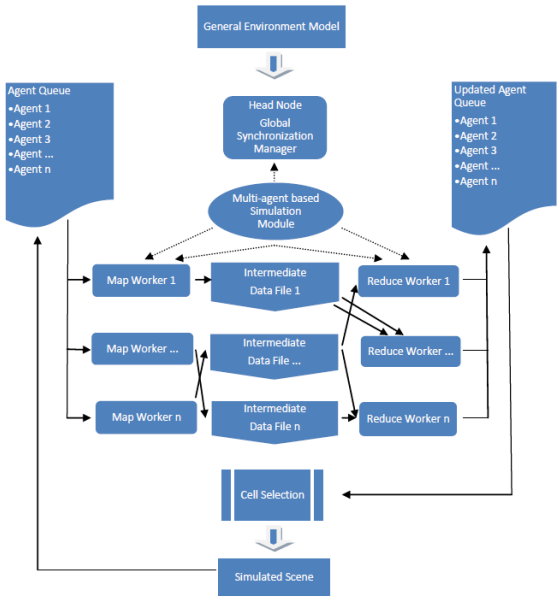

Figure 2. MapReduce-based Simulation Module Architecture

Agent Modelling

An agent contains the following information about its location, speed of movement, health, recent moves and a set of weights as well as a candidate cell queue. Cell queue consists of eight cells from the neighbourhood specified by the current agent's position in the environment. This situation is illustrated in Fig. 3. Each agent decides about possible movements in a given time step. If the agent decides to move, it can only move to one of the cells in the candidate list. More information about the agent model and its uses in the clouds is discussed in [27, 33].

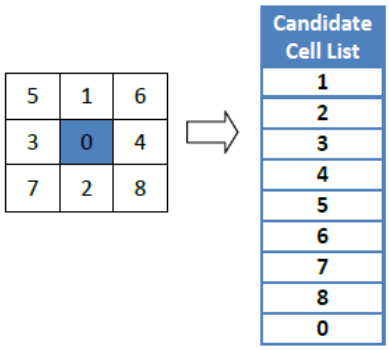

Figure 3. The Candidate Cell List

The crowd simulation process begins with the transformation of the agents in the agent items that contain key-value pairs, as shown in Fig. 4. A key is used to determine the position of the agent. It is unique due to the fact that in one cell it can be only one agent and it is static because the position does not change during the processsing stage of the loop. An important element which deserves attention is that Hadoop sorts the agent items depending on their state of health. Then, agent items are transferred to the mapper which is described below.

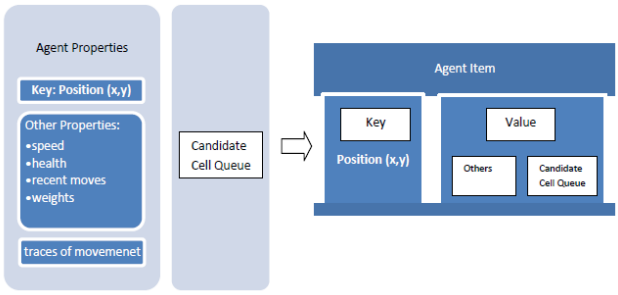

Figure 4. The Transition From Agent Model to Agent Item

The Mapper

The input of the agent is transmitted to the mapper that creates a set of intermediate key/value pairs, each of these pairs determines the attraction value of each item in the queue of agent's candidate cells. Identifier of each of these pairs is *iKey*, which means that the items in the queued candidate cells of the same agent have the same *iKey*. This phenomenon is characteristic of the reducer by the Hadoop framework. Environment data, which requires the mapper can be presented in the form of a general environment model. For small and simple problems, they can be transferred from the standard input or to speed up the simulation built into the mapper.

The mapper calculates the cell attraction value of all eight surrounding cells and produces an intermediate table, as shown in Fig. 5. The intermediate table contains information of the agent itself and a candidate cell queue with each cell's attraction value. The first

item corresponds to the current cell itself, which has the lowest attraction value since the agent on current cell will choose to stand still only if it cannot move to any of the 8 surrounding cells. Next, the intermediate table is sorted by Hadoop framework in descending order according to cell attraction value of each pair.

The Reducer

Agent items with the same *iKey* are combined to reproduce the agent information, as shown in Fig. 5. It is worth noting that the output of the reducer is the same as the input of the mapper, as shown in Fig. 6. Thus, the role of the reducer is to collect information from the sorted intermediate table and generate agent information, which is then transmitted to the Hadoop framework to create a list of agents.

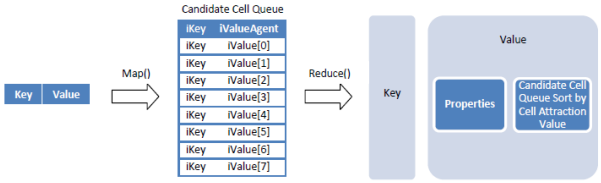

Figure 5. The Map-and-Reduce Process used in the Simulation

Cell Selection and Agent Movement

Every moving of an agent can affect other agents and attempt to parallel shift can cause errors. To prevent this happening, the parallel map and reduce operations only generates a sorted candidate cell queue for each agent but do not move agents. These movements are performed in a processing stage called a cell selection. Here, the agent queue is processed one by one. For each agent one cell is selected as a target, but it must be a cell that has not been previously taken by another agent. Then the agent is moved and the information about its location is updated.

After the cell selection stage, the simulated scene of the current time-step is fully ready. A new agent queue is also produced, which is the input for the next time-step.

Simulation Process Control and Data Generation

The simulation process is a process that takes place in an infinite loop, each iteration represents exactly one time-step, as shown in Fig. 6. It is an asynchronous process which is run in the cloud. However, if a user sends a request, the data requestor will send an imperative synchronization signal to the simulation module. At the time of request processing a global lock is set, which means that all nodes in the cloud are stop for a specified period of time. Then, the entire static (frozen) simulation scenario is generated by all the nodes and then returned to the user. After this process is completed, the global lock is canceled and nodes return to their previous jobs.

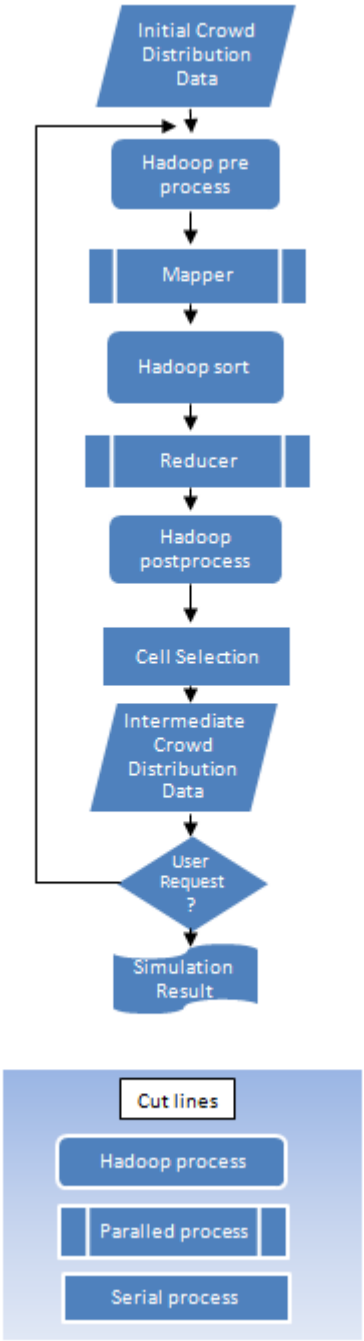

Figure 6. Simulation Process

CASE STUDY

In this Section we demonstrate results of simple experiments provided for verification of the correctness of platform implementation and cloud system motivation, and agent-based crowd management in the evacuation scenarios.

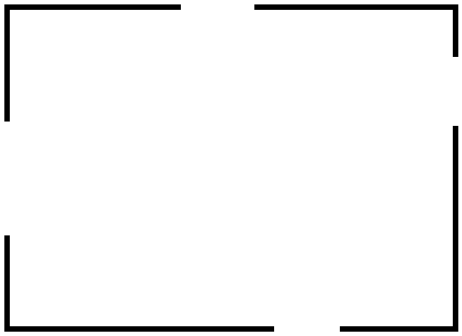

Figure 7. Environmental map model

The environmental model includes a simple map of 9000 m^2 indoor area with 4 exits is presented in Fig. 7. The tests were run on a small physical Hadoop cluster with one head node and 20 slave nodes with identical hardware and software configuration. This small physical cluster is connected with the wide public OpenStack cloud environment to provide proper services for crowd management. The mapper and the reducer module were written in C++ and implemented by using the Hadoop framework (streaming module). The environment information is generated by a map editor and compiled in both mapper and reducer modules. A platform evaluation process is monitored by the internal shell mechanisms in the cloud. The monitoring results are stored in the shellsripts. Those scripts are used also for activating the Hadoop framework and running the mapper and the reducer.

We compared the results achieved by the cloud-based Hadoop cluster with the result of similar tests performed for a single powerful workstation with the same and 10 times smaller crowd, where all operations were sequential without the cloud support. This scenario isvery typical for conventional evacuation systems and most of realistic evacuation scenarios so far. The main idea of such a comparative analysis was to demonstrate benefits of using Hadoop clusters, even if the synchronization of parallel processes in the system may delay the whole crowd management. Table 1 shows hardware and software settings for Hadoop cluster and the sequential server.

TABLE I. CONFIGURATION OF HADOOP NODES AND THE POWERFUL SEQUENTIAL SERVER USED IN EXPERIMENTS

Hardware for the Server and a Hadoop nodes	***CPU***	Intel Core i7 3770 4x 3.4GHz
	RAM	8GB
	Network	1000Mbit Ethernet
Software	***OS for Server***	Windows Server 2010 64-bit
	OS Hadoop	Ubuntu 14.04 LTS 64-bit, hadoop 2.7.1

In our experiments, we first compare the simulation time of the crowd. We assume that in both systems – cloud with Hadoop and sequential server – we have 20000 agents and the size of the crowd increases in equal time intervals by 20% of the maximal amount of agents – it means 4000 in each time interval. Another criterion was a simulation cost measured for in the sense of total memory usage and memory working set size recorded after 50 time intervals. The length of the one time interval in simulation was set to 3 sec. For this experiment we have generated 2 environmental scenarios: (i) for the Hadoop cluster, we generated a large number (20,000) of individuals (agents); (ii) for powerful sequential server we generated a smaller number (2,000) of individuals (agents) trying to evacuate themselves from the same building. The number of agents for servers represented 1/10 of the number of agents for Hadoop cluster with 20 nodes, which makes the tests comparable: the avarage number of agents which can be managed by a Hadoop node is 2000.

Crowd evacuation times

Table II gives the execution times (in seconds) of the two types of crowd simulation systems with increasing crowd sizes.

TABLE II SIMULATION TIME

Nb of agents	Server	Hadoop
4000	899	1429
8000	1990	1599
12000	2976	2009
16000	N/A	2198
20000	N/A	2244

In the case of 4000 agents, the execution time in cloud simulation is longer (almost 2 times) than in the case of sequential server. The reason can be the communication overhead of the cloud services. Hadoop becomes to work better for big crowds (over 10000 agents). In those scenarios, the communication overhead is covered by the benefits gained by distributing computation load of agents to multiple compute nodes. The achieved results indicate that (i) the cloud architecture can significantly improve the runtime performance of executing complex

simulation scenarios and (ii) it scales well with the scenario's complexity.

Simulation cost

Table III shows the calculated simulation costs after 50 time intervals including total memory usage, memory working set size and average memory usage for one individual for both Hadoop cluster and the sequential server.

TABLE III SIMULATION COST

	Hadoop	Server
Total memory usage (KB)	1056597	122632
Memory working set size (KB)	52113	18872
Memory usage per individual (KB)	65.660	61.316
Memory working set size per individual (KB)	3.211	9.436

It could be expected that the total memory utilization and the memory working set size are larger for the Hadoop cluster. This is mainly because of the initial population of the agents, which was 10 times bigger than that generated for the single server. However, memory distribution per individual is almost the same, which shows good memory management results in the cloud system. But the most significant results are for the fourth parameter in Table III: the memory working set size for each individual. In the case of Hadoop, this parameter is only 34% of the value achieved for the single server. The main reason is that in the cloud environment, data processing operations are processes successfully executed for different data sets. In the case of the single server, each data mining procedure is executed as a new process. This comparison shows significant potential benefits of using the cloud system for supporting the crowd evacuation in critical scenarios. Such benefits can be observed in the case of a big disproportion in the crowd size (10 times larger in the cloud-support case).

CONCLUSIONS AND FUTURE WORK

In this paper, we presented an early-stage development results on OpenStack cloud-based multi-agent simulation platform for evacuation of the crowd from the indoor environment with the limited number of evacuation exits and evacuation path size. Environment in this model is represented by cell automata and interpreted as a potential field, in which generated agents are located. The crowd management in the cloud is supported by the MapReduce programing model with the classical Hadoop framework used for its implementation. Simple experiments were performed on a small Hadoop cluster with ten nodes and separately for a single powerful server in order to demonstrate potential benefits of using the cloud system. The results of the experiments show that cloud-based systems can reduce significantly the complexity of the management of individuals in the crowd. Moreover, there is no need to initiate the large number of new processes on the same work station cause some data processing operations can be performed by using the software frameworks shared inside the public cloud.

The research presented in this paper is just the first step in the long period research plans. In the future work, we would like to improve the mechanisms used for control of the simulation process and the cell selection operation in order to reduce the number of sequential operations in the system as much as possible. We plan to apply KVDB-based data system, such as Apache HBase to improve the organization of agent data tables. Additionally, the generic environmental model should be improved to illustrate both individuals and environment activities, which allow us to provide our simulations in fully dynamic environments. Further experiments with larger crowd sizes and highly complex environments should be performed to improve the stability and efficiency of the developed simulation platform.

ACKNOWLEDGMENT

The inspiration for the research presented in this paper is the result of work in the IC1406 COST Action Horizon 2020 project cHiPSet "High-Performance Modelling and Simulation for Big Data Applications".

REFERENCES

[1] T. J. Shields, K. E. Boyce and N. McConnell, "The behaviour and evacuation experiences of WTC 9/11 evacuees with self-designated mobility impairments," *Fire Safety Journal,* vol. 44, pp. 881-893, 2009.

[2] P. F. Johnson, C. E. Johnson and C. Sutherland, "Stay or Go? Human Behavior and Decision Making in Bushfires and Other Emergencies," *Fire Technology,* vol. 48, pp. 137-153, 2012.

[3] C. M. Henein and T. White, "Macroscopic effects of microscopic forces between agents in crowd models," *Physica A-Statistical Mechanics And Its Applications,* vol. 373, pp. 694-712, 2007.

[4] D. Lee, J. H. Park and H. Kim, "A study on experiment of human behavior for evacuation simulation," *Ocean Engineering,* vol. 31, pp. 931-941, 2004.

[5] R. L. Hughes, "The flow of human crowds", *Annual Review of Fluid Mechanics*, vol. 35, pp. 169-182, 2003.

[6] L. Y. Jiao and Q. Y. Jiang, "Study on Emergency Management in Large-Social Activities Based on Behavior Modification Theory", *Progress in Safety Science and Technology, Vol. 8, Pts A And B*, Part a, pp. 451-454, 2010.

[7] J. Was and K. Kulakowski, "Agent-Based Approach in Evacuation Modeling," *Agent and Multi-Agent Systems: Technologies and Application*, pp. 325-330, 2010.

[8] Y. Q. Lin, I. Fedchenia, B. LaBarre, and R. Tomastik, "Agent-Based Simulation of Evacuation: An Office Building Case Study," *Pedestrian And Evacuation Dynamics 2008,* pp. 347-357, 2008.

[9] M. H. Zaharia, F. Leon, C. Pal, and G. Pagu, "Agent-Based Simulation of Crowd Evacuation Behavior," in *Proceedings Of The 11th Wseas International Conference on Automatic Control, Modelling And Simulation,* pp. 529-533, 2009.

[10] S. Sharma, H. Singh and A. Prakash, "Multi-Agent Modeling and Simulation of Human Behavior in Aircraft Evacuations," *IEEE Transactions On Aerospace And Electronic Systems,* vol. 44, pp. 1477-1488, 2008.

[11] N. Zarboutis and N. Marmaras, "Design of formative evacuation plans using agent-based simulation," *Safety Science,* vol. 45, pp. 920-940, 2007.

[12] T. Mao, H. Jiang, J. Li, Y. Zhang, S. Xia, and Z. Wang, "Parallelizing continuum crowds," in *Proceedings of the 17th ACM*

Symposium on Virtual Reality Software and Technology (ACM VRST) 2010, 231-234, 2010.
[13] G. Vigueras, M. Lozano, C. Perez, and J. M. Orduna, "A scalable architecture for crowd simulation: Implementing a parallel action server," *International Conference on Parallel Processing*, pp. 430-437, 2008.
[14] D. Chen, L. Wang, C. Bian, and X. Zhang, "A grid infrastructure for hybrid simulations," *Computer Systems Science and Engineering,* vol. 26, pp. 197-206, 2011.
[15] D. Chen, L. Wang, X. Wu, J. Chen, S. U. Khan, J. Kolodziej, M. Tian, F. Huang, and W. Liu, "Hybrid modelling and simulation of huge crowd over a hierarchical Grid architecture," *Future Generation Computer Systems,* vol. 29, pp. 1309-1317, 2013.
[16] Y. Wang, M. Lees and W. Cai, "Grid-based partitioning for large-scale distributed agent-based crowd simulation," in *Proceedings of the 2012 Winter Simulation Conference*, Berlin, Germany, pp. 2727-2738, 2012.
[17] E. Yilmaz, V. Isler and Y. Y. Cetin, "The virtual marathon: Parallel computing supports crowd simulations," *IEEE Computer Graphics and Applications,* vol. 29, pp. 26-33, 2009-01-01 2009.
[18] I. G. Georgoudas, P. Kyriakos, G. C. Sirakoulis, and I. T. Andreadis, "An FPGA implemented cellular automaton crowd evacuation model inspired by the electrostatic-induced potential fields," *Microprocessors and Microsystems,* vol. 34, pp. 285-300, 2010-01-01 2010.
[19] H. C. Yang, Z. B. Wang and J. S. Peng, "Production simulation using a distributed node-aware system," *IEEE International Conference on Industrial Engineering and Engineering Management* , pp. 2057-2061, 2010.
[20] G. Pratx and L. Xing, "Monte Carlo simulation of photon migration in a cloud computing environment with MapReduce," *Journal of Biomedical Optics,* vol. 16, 2011-01-01 2011.
[21] A. Ciobanu and F. Ipate, "P system testing with parallel simulators - A survey," *Scalable Computing,* vol. 14, pp. 169-179, 2013-01-01 2013.
[22] J. Niu, S. Bai, E. Khosravi, and S. Park, "A Hadoop approach to advanced sampling algorithms in molecular dynamics simulation on cloud computing," *IEEE International Conference on Bioinformatics and Biomedicine* , pp. 452-455, 2013.
[23] O. Scekic, C. Dorn and S. Dustdar, "Simulation-based modeling and evaluation of incentive schemes in crowdsourcing environments,"*Confederated International Conferences on On the Move to Meaningful Internet Systems*, pp. 167-184, 2013.
[24] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Communications Of The Acm,* vol. 51, pp. 107-113, 2008.
[25] R. Laemmel, "Google's MapReduce programming model - Revisited," *Science Of Computer Programming,* vol. 70, pp. 1-30, 2008.
[26] J. Dean and S. Ghemawat, "MapReduce: A Flexible Data Processing Tool," *Communications Of The Acm,* vol. 53, pp. 72-77, 2010.
[27] M. Dou, J. Chen, D. Chen, X. Chen, Z. Deng, X. Zhang, K. Xu, and J. Wang, "Modeling and simulation for natural disaster contingency planning driven by high-resolution remote sensing images," *Future Generation Computer Systems,* 2013-01-01 2013.
[28] D. Grzonka, "The Analysis of OpenStack Cloud Computing Platfrom: Features and Performance", *Journal of Telecommunications and Information Technology*, 3/2015, pp. 52-57.
[29] Taylor, R. C Taylor, "An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics", *BMC Bioinformatics*, 2010 Supplement 12, Vol. 11, p1, 2010.
[30] S. Wolfram, "Theory and applications of cellular automata," *Advanced Series on Complex Systems, Singapore: World Scientific Publication*, 1986.
[31] S. Wolfram, *Cellular automata and complexity: collected papers* vol. 1: ISBN 0201626640, Addison-Wesley Reading, 1994.
[32] R. Guo, H. Huang and S. C. Wong, "A potential field approach to the modeling of route choice in pedestrian evacuation," *Journal of Statistical Mechanics: Theory and Experiment,* vol. 2013, p. P02010, 2013-01-01 2013.
[33] A. Byrski, M. Kisiel-Dorohinicki, "Agent-based model and computing environment facilitating the development of distributed computational intelligence systems", Computational Science–ICCS 2009, 2009.
[34] M. R. Endsley, "Toward a theory of situation awareness in dynamic systems", *Human Factors Journal,* 37, 32-64, 1995

ANDRZEJ WILCZYŃSKI graduated with distinction in Computer Science from Cracow University of Technology, Poland, in 2015. He also studied at Fontys University of Applied Sciences, Netherlands, in 2012. Currently, he is a research and teaching assistant at Cracow University of Technology and a Ph.D. student at AGH University of Science and Technology, he works as a programmer as well. For more information, please visit: www.awilczynski.me. His e-mail address is: and.wilczynski@gmail.com.

JOANNA KOŁODZIEJ has graduated in Mathematics from the Jagiellonian University in Cracow in 1992, where she also obtained the PhD in Computer Science in 2004. She is employed at Cracow University of Technology as a professor. She has served and is currently serving as PC Co-Chair, General Co-Chair and IPC member of several international conferences and workshops including PPSN 2010, ECMS 2011, CISIS 2011, 3PGCIC 2011, CISSE 2006, CEC 2008, IACS 2008-2009, ICAART 2009-2010. Prof. Kołodziej is a Managing Editor of IJSSC Journal and serves as a EB member and guest editor of several peer-reviewed international journals. For more information, please visit: www.joannakolodziej.org.