EAI Endorsed Transactions
on Context-aware Systems and Applications

Research Article **ICST.ORG**

A federation of simulations based on cellular automata in cyber-physical systems

Hoang Van Tran[1,*], Hiep Xuan Huynh[1], Vinh Cong Phan[2], Bernard Pottier[3]

[1]Department of Software Engineering, College of Information & Communication Technology, Cantho University, Cantho city, Vietnam
[2]Department of Computer Engineering, Faculty of Information Technology, Nguyen Tat Thanh University, Hochiminh city, Vietnam
[3]Département Informatique, Université de Bretagne Occidentale, Brest, France

Abstract

In cyber-physical system (CPS), cooperation between a variety of computational and physical elements usually poses difficulties to current modelling and simulation tools. Although much research has proposed to address those challenges, most solutions do not completely cover uncertain interactions in CPS. In this paper, we present a new approach to federate simulations for CPS. A federation is a combination of, and coordination between simulations upon a standard of communication. In addition, a mixed simulation is defined as several parallel simulations federated in a common time progress. Such simulations run on the models of physical systems, which are built based on cellular automata theory. The experimental results are performed on a federation of three simulations of forest fire spread, river pollution diffusion and wireless sensor network. The obtained results can be utilized to observe and predict the behaviours of physical systems in their interactions.

Keywords: Cyber-physical system (CPS), cellular automata (CA), federation, high-level architecture (HLA), mixed simulations.

Received on DD MM YYYY, accepted on DD MM YYYY, published on DD MM YYYY

doi: 10.4108/_______________

1. Introduction

In recent years, more and more research focus on the cyber-physical systems (CPSs) [1], which are defined as integrations of computation, networking, and the physical systems. Taking advantages of wireless sensor network (WSN) [12], the sensing ability of CPSs is widely considered over the last years. This ability allows CPSs to be able to sense the physical world and transmit sensed data to base station for performing studies, analysis, and decision making.

Simulating the sensing ability before real implementations highly reduces cost and effort of the development of CPSs. However, due to uncertain interactions of complex physical systems, simulating such type of system is much more complicated compared to the traditional computing systems. One of the critical challenges is involving of interoperability in the models.

Recently, several approaches have been suggested to confront with that issue [5][6][20]. But, they do not so far consider on federating physical systems instead of tightly combining of existing tools and languages. Furthermore, those solutions are targeting embedded systems. Natural systems and phenomena have not still been involved and examined in literature. Thus, modelling and federating such systems and phenomena are taken into account in the context. For the rest of this paper, physical system and natural system are interchangeably used.

At present, cellular automata (CA) [2][3] model has emerged as a very promising technique for solving complex physical systems [4]. It has been used to address complex problems in many fields of science, engineering, computer science, and economy. In particular, parallel cellular automata models are effectively applied in fluid dynamics, molecular dynamics, biology, genetic, chemistry, road traffic flow, image processing, and environment modelling. Hence, we propose to use CA to facilitate modelling complex and large natural systems. A CA typically consists of two main components:

- *Cellular space:* This presents a lattice of cells. For each cell, we define a neighborhood that locally determines the evolution of the cell. All cells have

---

*Corresponding author. Email: tvhoang@ctu.edu.vn

Notably, the number of neighbours of each cell is determined by the chosen CA and they are typically equal for all cells. But, in this case, cells' neighbourhood can differ from cell to cell since their original positions in the physical system. For example, Figure 2 visualises a river cell system in which cells are close to the riverbanks have less neighbour than other cells.

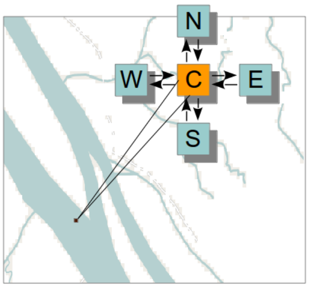


Figure 2. A cell system of a river system with Von Neumann 1.

b) The process of modelling physical systems: This is a set of ordered activities to achieve physical simulations from geographic data. Figure 3 simply depicts the process under the terms of the cell system definition. Geographic data are initially processed to generate cell systems, which are associated with definitions of states and transition rules to make up complete models. Physical simulations are obtained by just running the models.

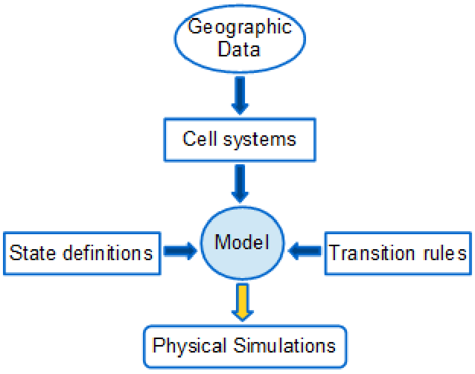


Figure 3. A process of developing physical simulations based on cell systems.

3.1. Eliminating useless calculations according to cell system model

Raster data are widely used as inputs in many natural phenomenon models, however, this often leads to useless time because of computations on the uninterested regions. In other words, they are regions outside the target systems, as depicted in Figure 4. Actually, several works have done to deal with that issue and one representative is presented in [26], in which cells are not belonging to any target areas are marked a label *"NoData"* in the preprocessing phase. During the execution of the models, those cells are omitted. It effectively works in most cases, however, time cost for data checking has not yet eliminated completely.
Cell system model helps to avoid meaningless processing by default. For example, in Figure 4, a cell system, which is extracted from geographic data, represents the river system. During executions, computations only occur on the river cell system. This approach is apparently more productive than the previous solutions.

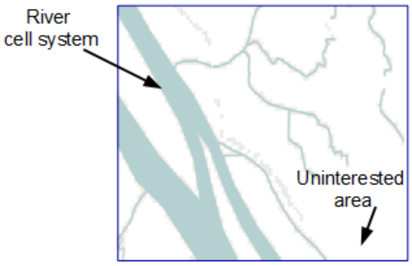


Figure 4. The river is considered as a target system.

In the next sections, we are going to present the models of two physical systems: forest fire spread model and river pollution diffusion model. In addition to those models, we suppose that a WSN [12] is deployed for monitoring fire in the forest, and its model is thus presented as well. Since the generations of cell systems are automatic by facilitating of an open source tool, the next considerations are definitions of local states and transition rules.

3.2. River pollution diffusion model

River pollution diffusion model is built based on the method presented in the previous section. In the context of pollution, it is possible to think of various potential situations such as chemical, oil, or contaminant. The diffusion of those generally depends on density. Thus, pollution density is chosen as cell state for this model.

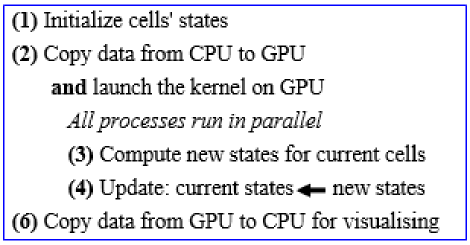
- *state:* The cell state holds a value of the pollution density.
- *transition rules:* Updating pollution density at $cell_i$ at time t+1, termed as S_i^{t+1}.

```
$S_i^{t+1} \leftarrow S_i^t/2$
for ($j$ in neighbours of $cell_i$)
    n $\leftarrow$ number of neighbours of $cell_j$
    $S_i^{t+1} \leftarrow S_i^{t+1} + S_j^t/(2*n)$
end for
```

mechanism has been regarded in academic as well as industrial works. Several new architectures have been proposed to make possible designing and development of high performance environment based on the CA theory. Significant instances of these environments are CAM [28], CAMEL [29], StarLogo [30], and NEMO [31]. These environments allow the exploitation of the inherent parallelism of the cellular automata model for the efficient simulation of complex systems that can be modeled by a very large number of cells with local interaction only.

To carry it out, we propose to use the graphic processing unit (GPU) [7][8] in hopes of accelerating very time-consuming simulations. The GPU provides hundreds of threads running in parallel. In this case, a cellular automaton is instanced as a SIMD (Single Instruction, Multiple Data) program. In fact, CA implemented as a number of processes mapped on threads that execute the same code on different data simultaneously. According to this approach, the transition function of a single cell of the system must be specified. As a result, the computations on cells are thus to be executed at the same time.

The details of the implementations of parallel simulations in accordance with the Cuda [10][9] model are formally presented in Figure 7.



```
(1) Initialize cells' states
(2) Copy data from CPU to GPU
    and launch the kernel on GPU
        All processes run in parallel
        (3) Compute new states for current cells
        (4) Update: current states ← new states
(6) Copy data from GPU to CPU for visualising
```

Figure 7. An implementation model of parallel simulations based on Cuda programming.

4.2. Mixed simulations

A mixed simulation is defined as a collection of parallel simulations organized as a distributed system. In such systems, many parallel simulations are concurrently run on different hosts connected by a network infrastructure. However, many important aspects of this type of system have to be taken into account the context.

For such systems, the major consideration is how to synchronize their activities among hosts having their own time, which may be different. Obviously, it is impossible to achieve a global time for all hosts. Therefore, a new mechanism is required for synchronizing time between simulations in the context.

To do that, we propose a central component for the proposed architecture. It is not only responsible for connecting parallel simulations, but also for coordinating their activities in time, as depicted in Figure 8. In other word, this component plays a role as a coordinator in the system.

Along with time synchronization, data exchanging among simulations is one of main focuses of this study. Due to interacting via a network, it is necessary to find out a way to effectively transfer data among hosts, but still ensure loose coupling and scalability characteristic of the system. This will clearly be described in the next section.

By achieving interoperability among distributed simulations, mixed simulations are expected to be able to imitate not only behaviours of real systems, but also interactions between them.

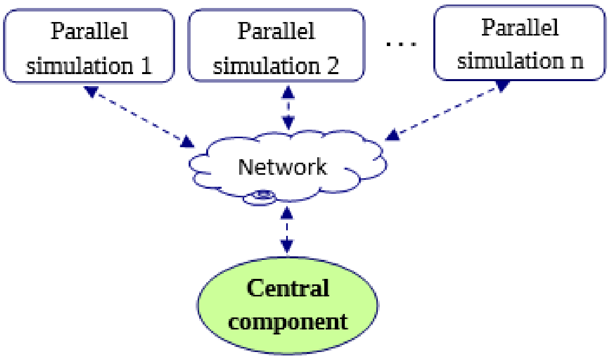


Figure 8. A general architecture of mixed simulations.

For instance, considering the interactions between the river and the forest system, as shown in Figure 9. Ashes produced by fire in the forest can pollute the river at the frontier between them. Otherwise, evaporation will also affects fired spread in the forest. Thus, those physical interactions will be put into models in this type of simulation.

Figure 9. An example about interactions being happened between river and forest system. (OpenStreetMap [16]).

activities based on local logical time together to ensure the causal relationships. It is understandable that two types of time are considered in the context, one is local time maintained by federates, the other is global time controlled by the central component.

This mechanism comes up with two properties, constrained and regulating. The former ensures the federates to be able to send updates. Meanwhile, the latter allows the federates to receive updates from the central component. Therefore, both of them are often enabled for all federates, and only the constrained property is assigned to the observer federate since it is designed without any sending. Table 4 shows time policies proposed for the proposed federation.

Table 4. Time management of the four federates.

Federate	Constrained	Regulating	Time advance
Forest	Yes	Yes	Time stepped
River	Yes	Yes	Time stepped
WSN	Yes	Yes	Time stepped
Observer	Yes	Yes/No	Time stepped

In order to synchronize time, each federate associates its logical time with sending data, so-called time-stamp. Thanks to this information, the central component is capable to calculate the next time step for the federation as well as to coordinate federates as a synchronous system.

Time stepped federates will calculate values based on a point in time and process all data being sent up to the next point in time (current time + time step). Thus, to advance logical time for the time-stepped simulation, each federate has to send its request to the central component. Then, all receive data, which have been sent from federates, with the time-stamp less than or equal to the time requested will be released from the central component. After those data have been received by federates, a time grant is returned to the requesting federate. And then, the federate is able to advance its logical time.

The time advancement of a Federate F is posed after sending UpdateAttributeValues(UAV) service, this phase has three steps:

Step 1: X sends a request using TimeAdvanceRequest (TAR) service.

Step 2: X can receive reflectAttributeValue (RAV) callbacks. Then, X may update its system with received data, for example.

Step 3: X waits for the granted time t2, TimeAdvancedGrant(TAG). At the TAG(t2) reception, the local time of the federate will be advanced to t2.

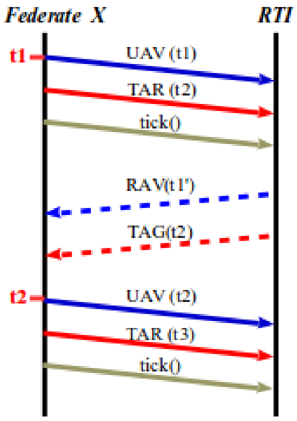


Figure 12. Time advancement process.

At the beginning of a federation execution, a synchronizing point is basically required. Figure 13 illustrates how to initialize the synchronizing point for all federates.

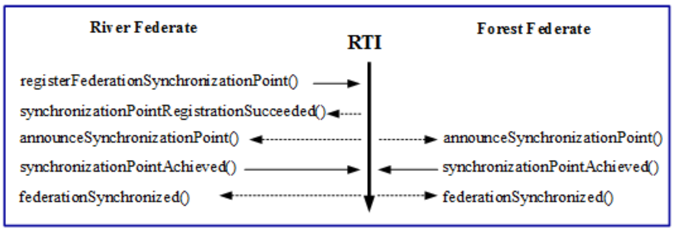


Figure 13. Federate synchronization for the river and the forest federate.

First of all, the river federate sends a synchronizing request to the RTI. The RTI will then responses to it and send an announce to the forest federate to achieve a synchronization point. Next, services will be used by both federates to confirm the synchronized point achieved.

Therefore, for the proposed federation, the time synchronization of the four federates can be obtained via the following steps.

Step 1: All federates connect to the federation initialized earlier.

Step 2: The federates are put into a common time progress by requesting synchronization point services, Figure 13.

Step 3: The federates update the new states, and then send the updates to the central component if they are publishers.

Step 4: Each federate sends an advance time request to the central component.

observer. In which, the two first ones run on the GPUs. In this case, the river federate created and joined the federation on the RTI-CERTI. It waited for the forest federates to enter. The river federate sent a request to others to achieve a synchronization point in the federation. And then the synchronization point was achieved.

Figure 16 showed that the ashes (brown points) formed by the fire (red points) polluted the river from the step 4 as they spread close to the river. This also shows that models based on the CA can work together in the common time progress.

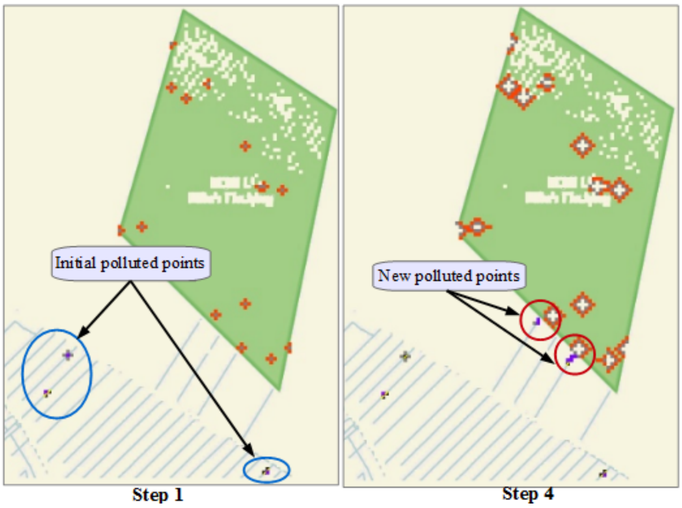


Figure 16. The screen shoot was taken from the observer federate. It shows the data exchange between the the river and forest federate in the federation. Two regions marked the small red circles represent the new polluted points created by the ashes, which are formed from the forest fire after 4 steps.

6.6. Scenario 4 – The federation of mixed simulation with the four federates: river, forest, WSN, and observer

The models were presented in Section 4.2. The sensors nodes were represented by black points in the forest. They appeared with the sensing ranges (small circles) and communication ranges (large circles).

As the previous case, the four federates first need to achieve a synchronous point. At each step, these federates exchange data together via the RTI.

Figure 17 presents the results captured from the observer federate. In this scenarios, due to no fire close to the river, until step 4, there were no new polluted points created in the river. Meanwhile, since a sensor recognizes that the fire appeared within its sensing range (smaller circle), it will change its color, sensing and communication range (larger circle) to red color.

Figure 17. The interoperability of four federates under the context of the mixed simulation: river, forest, WSN, and observer. The sensors changed to red color since the fire was detected close to them.

7. Conclusion

In context of modelling and simulating for cyber-physical systems, we have described a new approach on the federation for simulations. The models of physical systems are based on cellular automata. In this method, they must have at least two components: cell system and transition rules. The FEMIS tool has developed in order to simulate those models in parallel and perform the federations of those simulations. The parallel computations on the GPU aim to reduce the simulating time for large and complex models. The experimental results were obtained by federating the three parallel simulations for forest fire spread, river pollution diffusion and wireless sensor network. By using federated simulations, the behaviour of physical systems with their interactions could be observed in simulation progress.

References

[1] Edward A. Lee (2010) CPS Foundations. *Proc. of the 47th Design Automation Conference (DAC). ACM.* 737-742

[2] Stephen W. (1984) Computation theory of cellular automata *Communications in Mathematical Physics, vol.96, n.1. Springer-Verlag.* 15-57

[3] Alfons G.H. Hoekstra, Jiří K., Peter M.A. Sloot (2010) Simulating Complex Systems by Cellular Automata *Berlin and Heidelberg, Springer-Verlag, Chapter 1.* 1-16

[4] Wolfram, S. (1984) Cellular automata: a model of complexity. *Nature 31*, 419-424

[5] Riley D., Eyisi E., Bai J., Koutsoukos X., Xue Y., Sztipanovits J. (2011) Networked Control System Wind Tunnel (NCSWT)- An Evaluation Tool for Networked Multi-agent Systems. *The fourth International Conference on Simulation Tools and Techniques (SIMUTools), Barcelona, Spain*, 9-18

[6] Al-Hammouri, A.T., Branicky, M.S., Liberatore, V. (2008) Co-simulation Tools for Networked Control Systems. *Hybrid Systems: Computation and Control, Springer and Heidelberg, vol. 4981.* 16-29

[7] Li, Dan and Li, Xia and Liu, XiaoPing and Chen, YiMin and Li, ShaoYing and Liu, Kai and Qiao, JiGang and Zheng, YiZhong and Zhang, YiHan and Lao, C. (2012) GPU-CA model for large-scale land-use change simulation. *Chinese Science Bulletin, vol.57, n.19. SP Science China Press.* 2442-2452

[8] Du, Jun and Liang, Qiang and Xia, Yongchun (2012) Parallel Simulation Based on GPU-Acceleration. *AsiaSim 2012. Springer Berlin Heidelberg.* 355-362

[9] Blecic, Ivan and Cecchini, Arnaldo and Trunfio, GiuseppeA (2013) Cellular automata simulation of urban dynamics through GPGPU. *The Journal of Supercomputing, Vol.6, N.2. Springer US.* 614-629

[10] Gulati, Kanupriya and Khatri, SunilP (2010) GPU Architecture and the CUDA Programming Model. *Hardware Acceleration of EDA Algorithms. Springer US, Chapter 3.* 22-30

[11] Bernard P., Pierre-Yves L. (2014) Dynamic networks âĂŞ NetGen: objectives, installation, use, and programming. *Université de Bretagne Occidentale*

[12] LuÃŋs M.L. Oliveira and Joel J.P.C. Rodrigues (2011) Wireless Sensor Networks: a Survey on Environmental Monitoring. *Journal of Communications. Vol. 6, N.2.* 143-151

[13] IEEE (2010) IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)– Framework and Rules. IEEE Std 1516TM-2010, 1-38

[14] IEEE (2010) IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)– Federate Interface Specification. IEEE Std 1516TM-2010, 1-378

[15] IEEE (2010) IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)– Object Model Template (omt) Specification. IEEE Std 1516.2TM-2010. 1-110

[16] OpenStreetMap Mekong Delta Region, South of Vietnam. `https://www.openstreetmap.org` (accessed on 18 October 2015)

[17] Noulard E., Rousselot J.-Y., and Siron P. (2009) CERTI, an open source RTI, why and how. *Spring Simulation Interoperability Workshop*

[18] Bruno A., Siron P., Eric N. (2008) Running Real Time Distributed Simulations under Linux and CERTI. *2008 Euro Simulation Interoperability Workshop Proceedings, 08E-SIW-061*

[19] Alvarado, J.Rodríguez and Osuna, R.Vélez and Tuokko, R. (2008) Distributed Simulation in Manufacturing Using High Level Architecture. *Micro-Assembly Technologies and Applications, vol.260. Springer US.* 121-126

[20] Lasnier G., Cardoso J., Siron P., Pagetti C., and Derler P. (2003) Distributed simulation of heterogeneous and real-time systems. *Proceedings of IEEE International Symposium on Distributed Simulation and Real-Time Applications, 2013.* 55-62

[21] Kyoung-Soo W., Jong-Chan K., Chang-Gun L. (2011) A novel simulation framework for supporting real-time cyber-physical interactions. *The Fifth IEEE International Conference on Service-Oriented Computing and Applications (SOCA).*1-3

[22] Lin, Jinzhi and Wu, Ying and Wu, Gongyi and Xu, Jingdong (2012) NCCPIS: A Co-simulation Tool for Networked Control and Cyber-Physical System Evaluation. *Network and Parallel Computing, Springer Berlin Heidelberg, vol.7513*

[23] ForwardSim Inc. Simulation and Technologies: HLA Toolbox for MATLAB - HLA Blockset for Simulink. (2013). [Online]. Available: http://www.forwardsim.com

[24] Robert W. Scheifler and Jim G. (1986) The X window system. *ACM Transactions on Graphics. vol.5, n.2.* 79-109

[25] Zhou F., Li S., and Hou X. (2008) Development method of simulation and test system for vehicle body CAN bus based on CANoe. *The 7th World Congress on Intelligent Control and Automation (WCICA). IEEE.* 7515-7519

[26] Cirbus J., Podhoranyi M. (2013) Cellular Automata for the Flow Simulations on the Earth Surface, Optimization Computation Process. *Applied Mathematics & Information Sciences 7, No.6.* 2149-2158

[27] Quartieri J., Mastorakis N.E., Iannone G., Guarnaccia C. (2010) A Cellular Automata Model for Fire Spreading Prediction. *The 3rd WSEAS International Conference on URBAN PLANNING AND TRANSPORTATION (UPT '10), Greece, Volume: Latest Trends on Urban Planning and Transportation.* 173-179

[28] Toffoli T. and Margolus N. (1986) Cellular Automata Machines A New Environment for Modeling. *The MIT Press*

[29] Spezzano G., Talia D., Di Gregorio S., Rongo R., Spataro W. (1996) A Parallel Cellular Tool for Interactive Modeling and Simulation. *IEEE Computational Science & Engineering 3.* 33

[30] Resnick M. (1994) Turtles, Termites, and Traffic Jams. *The MIT Press, Cambridge.*

[31] Hutchinson D., Kuttner L., Lanthier M., Maheshwari A., Nussbaum D., Roytenberg D., Sack J.-R. (1996) Parallel Neighbourhood Modeling: Research Summary. *Proc. SPAA '96, Padua, Italy.* 204