A Quantization-based ODE Approximation and HPP-LGCA Approach to ARGESIM Benchmark C17 'SIR-type Epidemic' in a DEVS Environment based on MATLAB

Tobias Schwatinski*, Thorsten Pawletta

Wismar University of Applied Sciences, Dept. of Mechanical and Process Engineering,
Res. Group Computational Engineering and Automation (CEA),
PB 1210, 23952 Wismar, Germany; **tobias.schwatinski@hs-wismar.de*

Simulator. MATLAB is a widely used programming environment for numerical computations extended by different toolboxes. Using MATLAB's object oriented programming features a Parallel-DEVS simulation toolbox (PDEVStbx.) for discrete event oriented modeling and simulation is implemented [5]. The toolbox is based on the Discrete Event System Specification (DEVS) formalism by Zeigler and Chow [1, 2]. To support also continuous and hybrid system modeling the PDEVStbx is extended by specific components according to the Quantized-State-Systems (QSS) approach introduced in [3].

Modelling. The DEVS formalism provides a comprehensive framework for modeling and simulation based on systems theory. There are two basic system types used for modeling called atomic DEVS and coupled DEVS. The dynamic behaviour is specified in atomic DEVS. Both basic system types can be composed in coupled DEVS to define systems in a modular, hierarchical manner. The Classic DEVS formalism has been advanced several times. One extension is the Parallel DEVS formalism [1, 2]. A Parallel atomic DEVS is specified as follows:

$$\text{PDEVS} = \{X, Y, S, \delta_{ext}, \delta_{int}, \delta_{conf}, \lambda, ta\} \quad (1)$$

where X is the set of input values, S the set of sequential states, Y is the set of output values, δ_{ext} is the external transition function, δ_{int} is the internal transition function, δ_{conf} is the confluent function, λ is the output function and ta is the time advance function.

According to Figure 1 every atomic DEVS has an internal state $s \in S$. By the mean of the time advance function ta(s) the time step till the next internal event is calculated on the basis of the internal state s.

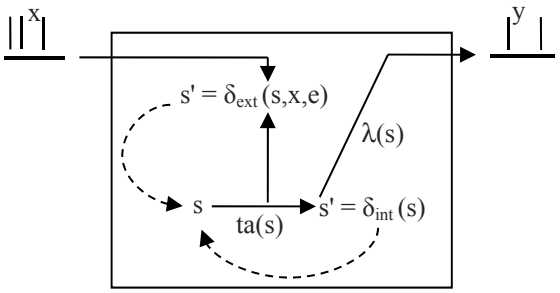


Figure 1. Dynamic behaviour of an atomic DEVS [1].

After having expired this time period the output function $\lambda(s)$ is carried out and as a result all output events $y \in Y$ at the output port are calculated on the basis of the internal state s. In the following the internal transition function $\delta_{int}(s)$ is carried out, which calculates the next state $s' \in S$ on the basis of the current state s. If external events $x \in X$ occur at the input port the external transition function $\delta_{ext}(s,x,e)$ will be executed. This function calculates the next state $s' \in S$ on the basis of the current state s and the elapsed time e since the last event and the current external events $x \in X$.

At the end the time advance for the next internal event is calculated by the time advance function ta(s). If external and internal events occur simultaneously the confluent function $\delta_{conf}(s,x,e)$ is used instead of the internal or external transition function for calculating the next state s'. A coupled DEVS is specified as follows:

$$\text{DEVN} = \{X, Y, D, \{M_d \mid d \in D\}, Z_{i,d}\} \quad (2)$$

where X is the set of input values, Y is the set of output values, D is the set of the component names, M_d is the DEVS system of component name $d \in D$ and $Z_{i,d}$ defines the coupling relations of a coupled DEVS.

SNE Simulation Notes Europe – Print ISSN 2305-9974 | Online ISSN 2306-0271
SNE 21(1), 2011, 57-60 | doi: 10.11128/sne.21.bn17.10053

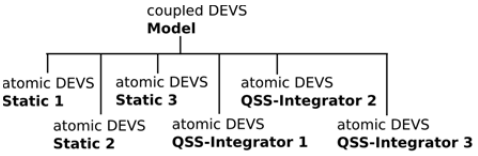
A modular, hierarchical DEVS model is executed by assigning a simulator to each atomic DEVS system and a coordinator to each coupled DEVS system. They control the evaluation of the system functions and the analysis of the coupling relations of the DEVS systems assigned to them. Simulators and coordinators communicate by messages. A special root coordinator manages the current simulation time, starts and stops a simulation run.

Modeling with QSS. The Quantized-State-Systems (QSS) method [3] deals with approximating differential equations, replacing the time discretization by a quantization of state variables. Thus a continuous system $\dot{\vec{x}} = f(\vec{x}(t), \vec{u}(t))$ is transformed to $\dot{\vec{x}} = f(\vec{q}(t), \vec{u}(t))$ where $\vec{x}(t)$ and $\vec{q}(t)$ are component-wise related by hysteretic quantization functions. During simulation the smallest time step h is calculated, such that $x(t+h) = x(t) \pm \Delta Q$ where ΔQ is the quantization level, t is the current simulation time and x is any quantized state value. It is important to point out that such an integration algorithm supports an adaptive step size control. QSS has many advantages in comparison with ordinary time-based ODE solvers. According to [3], the QSS method is always stable without using implicit formulae at all.

To solve the ODEs of benchmark C17 with the QSS method in a Parallel-DEVS simulation environment it is necessary to specify two basic atomic DEVS models according to definition (1) called Integrator and Static. Both basic atomic systems have a generalized DEVS specification [3]. Each Integrator handles and quantizes a specific continuous state quantity of an ODE system and each Static system calculates the derivative for one Integrator. Thus the structure of a QSS model looks quite similar to a control based block model like e.g. in MATLAB/Simulink.

Figure 2 shows the model structure of the example C17 using the QSS approach and the PDEVStbx. The atomic system types Static and Integrator are implemented as basic classes in the PDEVStbx. Each atomic DEVS component is incarnated from its basic class definition. Static objects have to be parameterized with a specific derivative formulae and Integrator objects have to be parameterized with the quantization level ΔQ and possibly with a hysteresis width ε.

Modeling with HPP-LGCA-cells using PDEVS. The behaviour of a single HPP lattice gas cellular automata cell can also be modelled based upon the common atomic PDEVS specification. In addition such atomic PDEVS components are specified in a coupled DEVS in accordance to the required domain.



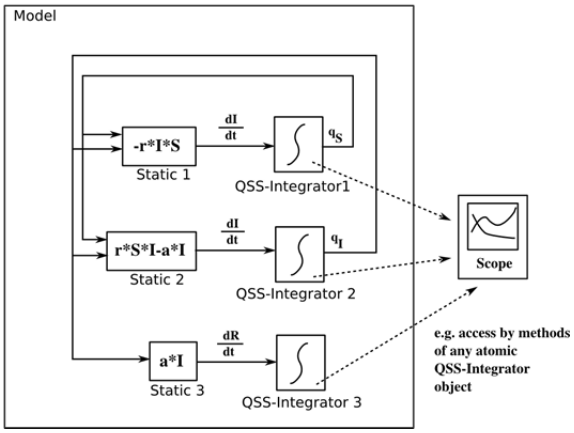


Figure 2. Composition tree and structure of the C17 model.

Therefore a HPP-LGCA atomic PDEVS component can be specified as follows:

- four input and output ports to bordering atomic DEVS used for the exchange of particles

X = {(p,v) | p∈InPorts, v∈Xp},
where: InPorts ∈ {I1, I2, I3, I4}
$X_{I1} = X_{I2} = X_{I3} = X_{I4} \in \{1, 2, 3\}$
Y = {(p,v) | p∈OutPorts, v∈Yp},
where: OutPorts ∈ {O1, O2, O3, O4}
$Y_{O1} = Y_{O2} = Y_{O3} = Y_{O4} \in \{1, 2, 3\}$

- two state variables

S = (sigma, storage) = ({1, ∞} x [Vector 1x4]),

where *sigma* is the time till the occurrence of the next internal event and *storage* is a vector which four elements present the occupancy of the HPP-LGCA cell with particles. In this context 0 indicates an empty field without any particle (1, 2 and 3 indicates the occupancy by a susceptible, infected or recovered particle).

- the internal transition function

  $\mathbf{s' = \delta_{int}(s),}$

where the new state s' is set to *storage* = [0, 0, 0, 0] and *sigma* = ∞.

- the external transition function

  $\mathbf{s' = \delta_{ext}(s, x, e),}$

  where at first *storage* is set to [0, 0, 0, 0] and afterwards the new state s' is composed by *storage* that is filled up with any incoming particles from the input x and *sigma* that is set to 1.

- the time advance function

  $\mathbf{\Delta t = ta\ (s),}$

  where the time period Δt till the next internal event is calculated from the current state s. In detail Δt is set to *sigma*.

- the output function

  $\mathbf{y = \lambda(s),}$

  where propagation and collision rules are implemented. As a result *y* contains the set of leaving particles distributed over the output ports.

- the confluent function

$\mathbf{\delta_{conf}(s, x, e) = \delta_{ext}(s, x, e),}$

which simply calls the external transition function δ_{ext}.

By the mean of HPP-LGCA-cells implemented as atomic PDEVS and composed in a coupled DEVS it is possible to analyze different vaccination strategies regarding *space* and *time* which are discussed in Task b.

A-Task: QSS, CA and ODE Simulation. To compare the results the given system of ODEs has been solved with MATLAB's ODE45 solver using the initial values and parameters shown in table 1.

S(t=0) = S0	16000
I(t=0) = I0	100
R(t=0) = R0	0
Infection rate r	0.6/10000
Recovery rate a	0.2

Table 1. Initial values and parameters for Task a.

Beside the simple QSS-method there exist different extensions which improve the abilities and computation speed. One improvement of QSS is to change hysteretic quantization to first order quantization.

In the ordinary QSS approach the output trajectory of an Integrator has to be constant within the quantization level. In contrast to QSS, the advanced QSS2 method allows a linear growth of the output trajectory within the quantization level [3]. In analogy to the QSS Integrator and Static system types QSS2 specific Integrator and Static system types are implemented in the PDEVStbx. Figures 3 and 4 show the simulation results for both QSS-methods, the PDEVS implementation of a HPP-LGCA approach and the standard MATLAB ODE45 solver. It is important to point out that the QSS solutions are computed using ordinary **PDEVS** simulators and coordinators.

The simulation results based on the QSS and QSS2 method are of similar qualitative and quantitative nature in comparison with ODE45. However, the QSS solution (b) in Figure 3 with a quantization level of ΔQ=10 causes higher numerical costs than the QSS2 solution (d) in Figure 4 with a smaller quantization level of ΔQ=5. The reason for this is the modified quantization function within the QSS2 method. The simulation result from the HPP-LGCA implementation differs from the ODE45 solution to the effect that the epidemic spreads slower. However, the qualitative nature of the solution is similar to ODE45.

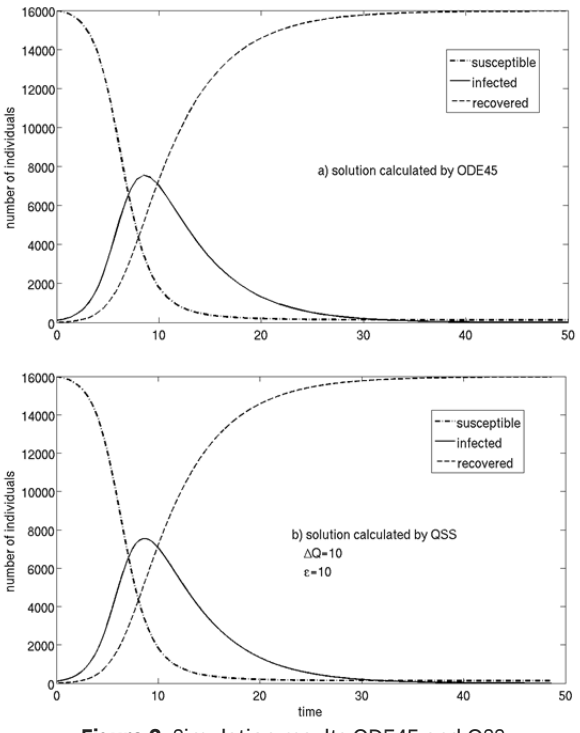


Figure 3. Simulation results ODE45 and QSS.

B-Task: Vaccination Strategies in CAs. The distribution of different particles over the cellular automata can be easily realized by setting the *storage* vector of every atomic HPP-LGCA cell during the generation of a coupled DEVS model. Figure 5 shows exemplarily simulation results of three possible vaccination strategies.

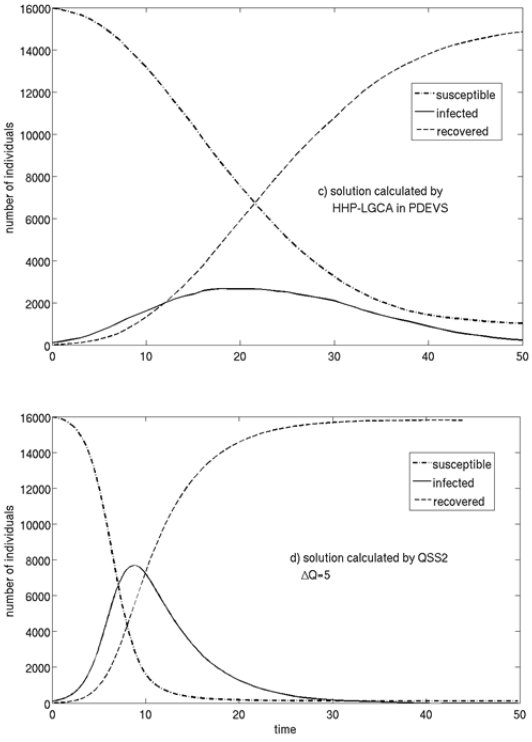


Figure 4. Simulation results HPP-LGCA and QSS2.

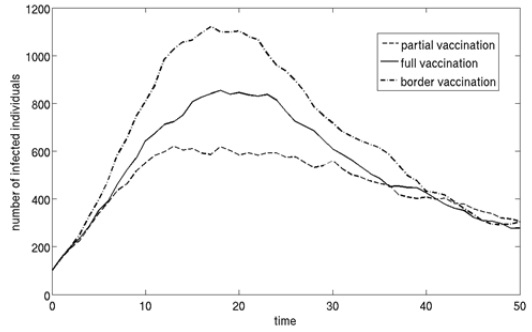


Figure 5. Simulation results of vaccination strategies.

At every strategy all infected particles are randomly distributed at the left half of the domain. Following Figure 5 the maximum amount of infected particles reaches the highest value for the border vaccination strategy where vaccinated particles are randomly distributed at the border of the whole domain. It is followed by the full vaccination strategy where vaccinated particles are randomly distributed over the whole domain.

The smallest amount of infected particles is reached for the partial vaccination strategy where vaccinated particles are randomly distributed together with infected particles at the left half of the domain.

Résumé. The benchmark shows on the one hand that both QSS approaches deliver the same results as MATLAB's ODE45 solver. Moreover the QSS approach offers additional methods for solving stiff-systems. These methods use future derivatives similar to implicit time integration methods. Because the QSS approach can be seamless integrated in a discrete event simulation environment like the PDEVStbx., its application is particularly useful in hybrid system simulation.

On the other hand the benchmark shows that even CAs can be modelled on the basis of the common PDEVS specification. Hence, it is possible to simulate models regarding to time and space. However, the generation of the domain is difficult and the simulation quite slow. The reason for this is the fact that PDEVS coordinator algorithms focus only on the coupling relations between individual atomic DEVS and not on their position on a domain or their neighbourhood. This leads to ineffectiveness and difficulties if the model is composed of many thousand square lattices which are only coupled with its four nearest neighbours. These drawbacks can be overcome by using the Cell-DEVS formalism [4] which allows the definition of cell spaces based on the DEVS formalism and CA models.

References

[1] B. P. Zeigler, H. Prähofer, T. G. Kim. *Theory of Modeling and Simulation,* Second Edition, Academic Press 2000.

[2] A. C.-H. Chow. *Parallel DEVS: A parallel, hierarchical, modular modelling formalism and its distributed simulator* , Transactions of the Soc. For Modeling and Simulation Int., Vol. 13 No 2, June 1996, 55-67.

[3] F. E. Cellier, E. Kofman. *Continuous System Simulation*, Springer 2006.

[4] G. A. Wainer. *Discrete-event modeling and simulation: a practitioner's approach*, CRC Press 2009.

[5] T. Schwatinski, T. Pawletta. *An Advanced Simulation Approach for Parallel DEVS with Ports*. In: Proc. of Spring Simulation Multiconference 2010, Orlando/Florida, USA, April 11-15, 2010.

Received: January 2011
Revised: March 6, 2011
Accepted: March 30, 2011