Performance Analysis of Continuous Cell-DEVS models

Gabriel Wainer
Dept. of Systems and Computer Engineering
Carleton University
1125 Colonel By Drive
Ottawa, ON. K1S 5B6. Canada.
gwainer@sce.carleton.ca

KEYWORDS
DEVS, Cell-DEVS, Quantized DEVS, CD++.

ABSTRACT
The Cell-DEVS formalism allows describing complex cellular models. When using very large models with continuous variables, the execution performance degrades. We experienced with different quantization techniques to reduce the number of messages generated by the simulator. We present two different strategies for the automatic update of the quantum sizes in different cells, and we discuss the use of quantized DEVS with hysteresis applied to cellular models. We obtained important reductions in the error involved, while maintaining the high performance of quantized DEVS models.

INTRODUCTION

In the last 20 years, cellular computing became popular as a tool for complex systems analysis. Cellular Automata (CA) [1] are organized as n-dimensional infinite lattices in which each element holds a state variable and a very simple computing function. These functions are local to each cell, and they execute synchronously using the state values of the present cell and neighbors. The Cell-DEVS formalism [2] permits describing cellular models as discrete-event systems based on the DEVS formalism [3]. A real system modeled with DEVS can be described as a composite of behavioral (atomic) and structural (coupled) submodels, which can be integrated into a hierarchy. In Cell-DEVS, each cell is seen as a DEVS atomic model, and a procedure for coupling cells is defined based on the neighborhood relationship. Explicit timing delays permit expressing complex timing conditions. The hierarchical nature of DEVS also permits the integration of these cellular models with others defined using different formalisms, resulting in enhanced facilities for modeling complex systems.

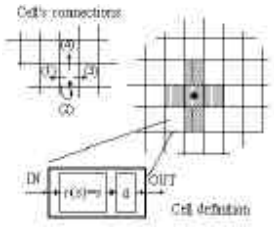

Figure 1. Description of a Cell-DEVS Model.

The CD++ toolkit [4] allows simulating DEVS and Cell-DEVS models, and it has been used to execute a variety of models (traffic, forest fires, biological systems and experiments in physics). As the complexity of the models grows, large data sets are generated, increasing execution. The situation is worse for models with continuous state variables.

Different efforts that showed how to simulate continuous DEVS models efficiently include quantized DEVS (Q-DEVS) [5], Q-DEVS with hysteresis [6], and Cell-DEVS with Dynamic Quantization [7]. When using quantized DEVS, a state value will be only informed to its neighbors if the cell's value crosses a threshold (called the *quantum*). This operation reduces substantially the frequency of message updates, while potentially incurring into error. DEVS with Hysteresis has strong stability, convergence and error bound properties. Cell-DEVS with Dynamic Quantization reduces the error by improving the precision of the most active cells.

We used different models as a workbench, including a model of the electrical activity of the heart tissue [8], a model of watershed formation [9], and a Flow Injection Analysis (FIA) system [10], which studies the automated analysis of liquid samples. We analyzed different metrics (error, execution time and number of messages), and we could determine how the dynamic quantization techniques improve the amount of error introduced by the quantizers.

DEVS MODELING

A DEVS atomic model can be defined as: $AM = < X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta >$. A DEVS model in state $s \in S$ will remain in that state for a period as defined by ta(s). When ta(s) expires, an internal transition runs: the model outputs the value $\lambda(s)$, and changes to the state $\delta_{int}(s)$. A state transition can also happen upon reception of an external event: δ_{ext} is activated with the input value, the current state and the elapsed time.

Coupled models, are defined as: $CM = < X, Y, D, \{Mi\}, \{Ii\}, \{Zij\} >$. They consist of a set of basic models (Mi, atomic or coupled) connected through the models' interfaces. Component identifications are stored into an index (D). A translation function (Zij) is defined by using an index of influencees created for each model (Ii). The function defines which outputs of model Mi are connected to inputs in model Mj.

Cell-DEVS allows the creation of cellular models, in which each cell is defined as a DEVS atomic model. A Cell-DEVS atomic model is defined as: $TDC = < X, Y, S, \theta, N, d, \delta_{int}, \delta_{ext}, \tau, \lambda, D >$. A cell uses a set of input values N to compute its future state, which is obtained by applying the local function τ. A delay function is as-

Proceedings 18th European Simulation Multiconference
Graham Horton (c) SCS Europe, 2004
ISBN 3-936150-35-4 (book) / ISBN 3-936150-36-2 (CD)

sociated with each cell, after which, the new state value is transmitted to the neighbor cells. After the basic behavior for a cell is defined, a complete cell space can be built as a coupled Cell-DEVS: *GCC = < Xlist, Ylist, X, Y, n, {t1,...,tn}, N, C, B, Z >*. A coupled Cell-DEVS is composed of an array of atomic cells (C), each of which is connected to the cells in the neighborhood (N). The border cells (B) can be programmed with a different behavior than the rest of the space. The Z function defines the internal coupling of cells in the model. Xlist and Ylist define the coupling with external models.

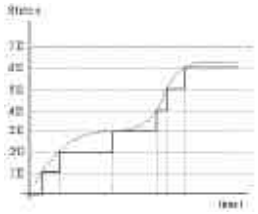

Figure 2. Quantized DEVS [5].

Performance of models with continuous variables is reduced due to the large number of messages interchanged by the simulation engines. The theory of quantized DEVS [5] represents continuous signals by the crossings of an equal spaced set of boundaries (defined by a *quantum* size), as showed in Figure 2. This operation reduces substantially the frequency of message updates, while potentially incurring into error. DEVS Quantized systems with hysteresis [6] improved these results (models showed to have strong stability, convergence and error bound properties). Let $D = \{d_0,.., d_m\}$ be a set of real numbers where $d_{i-1}<d_i$, $x \in \Omega$ is a continuous trajectory where: $x:R \rightarrow R$ and $b:\Omega xR \rightarrow \Omega$ is a mapping where $q=b(x,t_0)$ that satisfies:

$$q(t) \begin{cases} d_m & \text{if } t = t_0 \\ d_{i+1} & \text{if } x(t) = d_{i+1} \wedge q(t^-) = d_i \wedge i < r \\ d_{i-1} & \text{if } x(t) = d_{i-E} \wedge q(t^-) = d_i \wedge i > r \\ q(t^-) & \text{otherwise} \end{cases}$$

$$m \begin{cases} 0 & \text{if } x(t_0)<d_0 \\ r & \text{if } x(t_0)>d_r \\ j & \text{if } d_j <= x(t_0)<d_{j+1} \end{cases}$$

Here, the hysteresis width is E and the parameters d_0 and d_r are the lower and upper saturation values. In [6], the authors proved that when the hysteresis width is set equal to the quantum size, we obtain the smallest possible error.

Cell-DEVS with Dynamic Quantization [7] tries to reduce the error by improving the precision of the cells. An active cell can appear as quiescent due to the selection of a quantum size covering the activity area, and if the quantum size is reduced, a smaller error will be obtained. Simultaneously, if we increase the quantum size in the cells with steep update functions, the execution time can be improved at a low cost in terms of the error introduced. Two different strategies were proposed to adjust the quantum size. Let *q* be the base quantum, *r* the adjustment ratio for the dynamic quantum, and *d(t)* the quantum value used in time *t*. If *v=Last Threshold Value*, *v'=new value*, and *q(0)=q*, then:

Strategy 1 $\neg$ regionChange(v,v',d) $\Rightarrow$ d=q*(1 - ratio);
regionChange(v,v',d) $\Rightarrow$ d=q*(1+ratio);

Strategy 2 regionChange(v,v',d) $\Rightarrow$ d=q*(1-ratio);
$\neg$regionChange(v,v',d) $\Rightarrow$ d=q*(1+ratio);

where regionChange(v,v',q) = ($v=\phi \mid q=0 \mid (q \neq 0 \wedge [v/q] \neq [v'/q])$). Strategy 1 tries to reduce the quantum size if the result of updating the cell's value does not cross the threshold (otherwise, the quantum increases). This technique reduces the quantum size for cells with high update rates, and increases it for cells with low update rates. Strategy 2 reduces the quantum size every time a threshold is crossed (otherwise, it increases). This strategy reduces the number of messages involved in the simulation at a cost of a higher error.

We tested these techniques on CD++, which was originally built as an implementation of DEVS and Cell-DEVS theories. Cell-DEVS models are defined using a built-in language. The behavior of a cell is defined using rules with the form: VALUE DELAY {CONDITION}. If the CONDITION is satisfied, the state of the cell will change to the VALUE, and this new state value will be spread to the neighboring cells after the DELAY. Common operators are included: boolean, comparison and arithmetic; trigonometric, roots, power, rounding and truncation, module, logarithm, absolute value, minimum, maximum, etc. [4].

DEFINING COMPLEX CELL-DEVS MODELS

We used different complex models as a workbench, including a heart tissue model [8], a watershed [9], and a Flow Injection Analysis (FIA) system [10]. These examples, are the most complex cases of a larger pool of models we executed, and they represent different categories of interest. The heart tissue model permits analyzing systems in which the behavior of all of the cells is alike. The watershed model, instead, permits analyzing systems that tend to a steady state after a transient period. Finally, the FIA model shows a case of a model in which the behavior of different cells does not follow a predefined pattern.

A model of the heart tissue behavior

The heart muscle is excitable, and its cells respond to external stimuli by contracting the muscular cells. Hodgkin and Huxley [11] originally characterized the behavior of this cell membrane. They showed that if a stimulus is too weak, the muscle does not respond; instead, if the voltage received is adequate, they contract at maximum capacity. Whereas solving the equations representing this behavior using numerical methods for one cell is feasible, the use of this model in a realistic reproduction of the heart tissue (probably consisting of millions of cells) can be computationally expensive.

Different authors tried to simplify the complexity of the equations; for instance [12], presented the use of CA (with simple rules for the model's behavior, but at a cost in precision) and in [8] we presented the use of Cell-DEVS to build a discrete variable model of heart tissue

conduction. Our Cell-DEVS model executes the Hodgkin-Huxley model in each of the cells, as follows.

```
[heart]
dim : (50,50)  delay : transport
border : nowrapped
neighbors : (-1,-1) (0,-1) (-1,1) (0,-1)  (0,0)
(0,1) (1,-1)  (1,1)  (0,1)
localtransition : heart-rule-AP

[heart-rule-AP]
rule : { AP(0,0) } 1 { ((-1,0) > 0 or
     (0,-1)>0 or (-1,-1)>0) and (0,0)=-83.0) }
```

Figure 3. Heart tissue model: Cell-DEVS definition.

We first define the size of the cell space (50x50 cells), the kind of delay, and the neighborhood shape (in this case, all the adjacent cells). Then, we define the local computing function, called *heart-rule-AP*. The rule will be evaluated only if the cell is resting and a positive voltage is detected in the cell's neighborhood. This rule will trigger the update of the cell state using the Hodgkin-Huxley equations in [11].

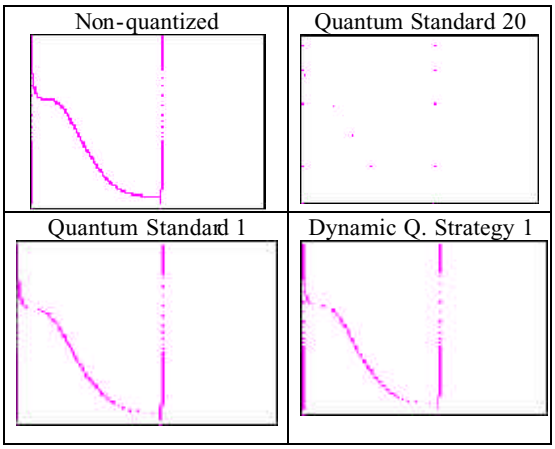

Figure 4. Sample execution of the Heart tissue model.

A Watershed model

Watersheds are regions defined by the shape of the land surface, which store up water because of rain, ice melting and rivers. In [13], the authors defined a hydrology model in which they identified several verticals layers composing a watershed: air, vegetation, water surface, land surface and stones . The model was divided in equal portions of land (cells), permitting analyzing the water distribution and the influence of the topology.

Figure 5 shows a description of this model. When the rain is absorbed by the vegetation, the rest is received by the surface. Depending on the topology, the cells can also receive/send, water from/to the neighbors. Part of the water received is lost due to the filtration over the land and stones. The accumulated water on a period depends on: the quantity of effective water (rain), the quantity of water dumped from the neighbor cells (effective rain plus the water received from the neighbors minus water sent to the neighbors) minus the water filtered by stones and the soil.

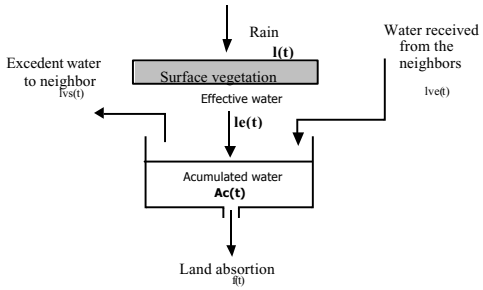

Figure 5. Hydrology model [13].

We can see the execution results of this model in Figure 6. We first show the initial state, representing the slope of the terrain before raining (each cell is 1x1m). The remaining figures show the execution results after intense rain (0.0022 mm/s) after 10 minutes of simulated time.

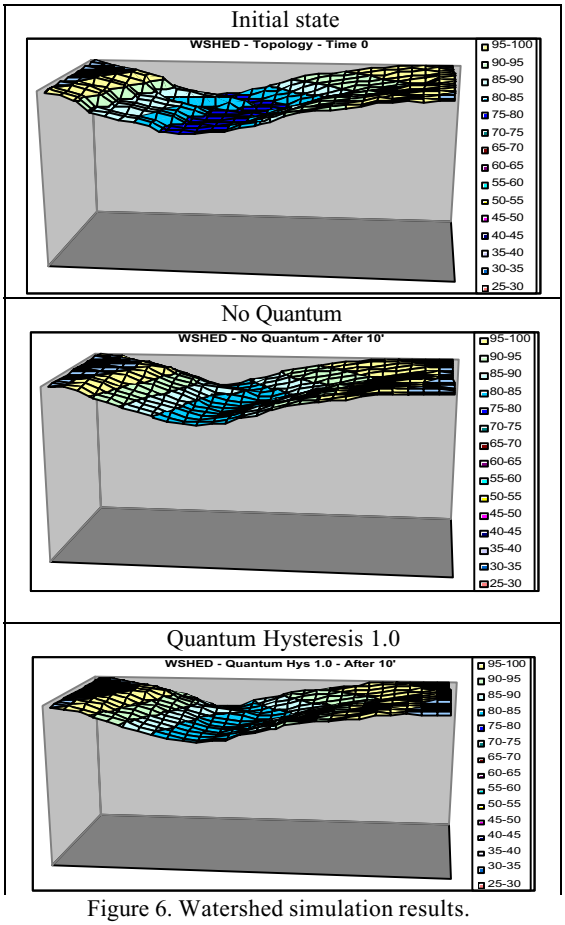

Figure 6. Watershed simulation results.

These results were obtaining by running the hydrology equations in CD++, using a mechanism similar to the one presented for the heart tissue model. These rules represent that the present water of the cell, and the rain are added. Then, we consider how much water must be passed to the neighbors, and how much water is received from the inverse neighborhood. The different layers are represented as planes in a three dimensional model.

```
[Watershed]
dim: (30,30,2)      border: nowrapped
delay: transport    localtransition: Hydrology
neighbors : (-1,0,0)(0,-1,0)(0,0,0)(0,1,0)
(1,0,0)(-1,0,1)(0,-1,1)(0,0,1)(1,0,1)(0,1,1)

[Hydrology]
rule : {0.0022+(0,0,0) - if((-1,0,0)!=?)and
((0,0,1)+(0,0,0)>((-1,0,1)+(-1,0,0)),((0,0,0)+
(0,0,1)-(-1,0,0)-(-1,0,1))/1000)*(0,0,0))
/1000),0)-if((1,0,0)!=?) and((0,0,1)+(0,0,0))>
((1,0,1)+(1,0,0)),((0,0,0)+(0,0,1)-(1,0,0)-
(1,0,1))/1000)*(0,0,0))/1000),0)-if((((0,-1,0)
!= ?) and ((0,0,1)+(0,0,0))>((0,-1,1)+(0,-1,0)
),((0,0,0)+(0,0,1)-(0,-1,0)-(0,-1,1))/1000)*
(0,0,0))/1000),0)-if((0,1,0)!=?) and ((0,0,1)
+(0,0,0))>((0,1,1)+(0,1,0)),(((0,0,0)+(0,0,1)-
(0,1,0)-(0,1,1))/1000)*(0,0,0))/1000),0)+ if(
(-1,0,0)!=?) and ((-1,0,1)+(-1,0,0))>((0,0,1)+
(0,0,0)),((-1,0,0)+(-1,0,1)-(0,0,0)-(0,0,1))*
(-1,0,0))/1000),0) + if((1,0,0) != ?) and
((1,0,1) + (1,0,0))>((0,0,1)+(0,0,0)),((1,0,0)
+(1,0,1)-(0,0,0)-(0,0,1))* (1,0,0))/1000),0)+
if((0,-1,0)!= ?) and ((0,-1,1)+(0,-1,0)) >
((0,0,1)+(0,0,0)),((0,-1,0)+(0,-1,1)-(0,0,0)-
(0,0,1))*(0,-1,0))/1000),0)+if((0,1,0)!=?) and
(((0,1,1)+(0,1,0))>(0,0,1)+ (0,0,0)),((0,1,0)
+ (0,1,1)-(0,0,0)-(0,0,1))*(0,1,0))/1000),0) }
1000 { cellpos(2)=0 }
```

Figure 7. Hydrology model in CD++.

A Flow Injection Analysis Model

Flow-injection methods are used for automated analysis of liquid samples. In a flow injection analyzer, a small fixed volume of a liquid sample is injected as a discrete zone using an injection device into a liquid carrier. Because of convection at the beginning, and axial and radial diffusion later, this sample is progressively dispersed into the carrier as it is transported along the tube, which can be sensed by flow-through sensors.

In [9] we built a Cell-DEVS model describing the integrated conductivity in detail. The model studied a 0.025 cm radius tube, a 10.75 cm loop and a 9,25 reactor coil. We assumed the total tube length of the tube to be of 20 cm and defined a cell space of 25 rows and 200 columns.

```
[fia]
width : 200  height : 25    delay : inertial
border : nowrapped
neighbors : (-1,-1) (-1,0) (-1,1) (0,-1) (0,0)
(0,1) (1,-1) (1,0) (1,1)
localtransition :  transport

[transport]
rule : {(0,-1)} {0.1/(22.578*(1-
power(cellPos(0)*.001+.0005,2)/.000625))*1000}
       {cellPos(1)!=0}
rule : { 0.8 } {.1/(22.578*(1-power(
cellPos(0)*.001+.0005,2)/.000625))*1000}
       {cellPos(1)=0}

[diffusion]
rule: {((-1,0)+(0,0)+(1,0))/3} 1 {cellPos(0)!=
0 AND cellPos(0)!=24 }
rule: {((-1,0)+(0,0))/2 } 1 {cellPos(0)!=0 AND
      cellPos(0) = 24}
rule: {((0,0)+(1,0))/2} 1 {cellPos(0)=0 AND
       cellPos(0) != 24 }
```

Figure 8. A model of FIA in CD++.

The value of each cell represents the concentration of nitric acid in the carrier. To deal with convective transport and radial diffusion at the same time, the model reacts in two phases: transport and diffusion.

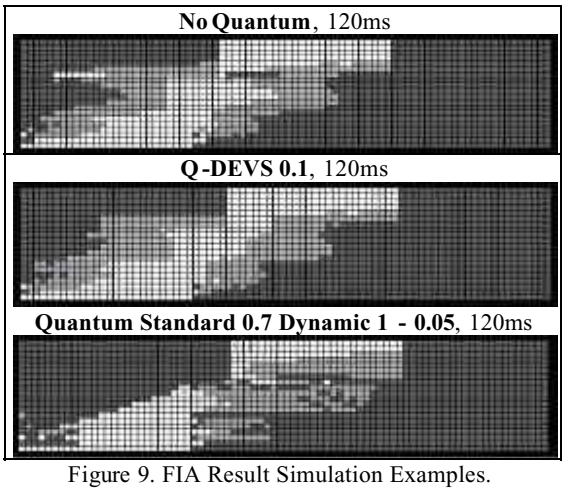

Figure 9. FIA Result Simulation Examples.

PERFORMANCE ANALYSIS

We executed simulations of the models presented in the previous section, analyzing two main metrics: execution time (number of messages involved in the simulation), and error. We executed a large number of tests in different categories, including:

- Non-quantized (noted as DEVS in the figures).
- Standard quantization (noted as Q-DEVS).
- Quantization with hysteresis (noted as H).
- Dynamic quantization, strategy 1.
- Dynamic quantization, strategy 2.

Different combinations of the previous categories with different quantum sizes and update ratios were used. The error is obtained by comparing the values in quantized versus non-quantized cases.

Figure 10 presents the cumulative error obtained with different strategies for the heart tissue model. The figure compares the different strategies (using different update ratios for the Dynamic DEVS strategies 1 and 2). The results obtained with standard and hysteresis quantum overlap, because the results of Hysteresis quantum differ from the standard when direction changes are present (and, as seen in Figure 4, there is only one).

The lowest error was obtained with dynamic quantum Str1 with ratio 0.9. Str1 results were better than the Str2 and standard Q-DEVS (the larger the ratio, the better the result), as expected. The quantum size is adjusted very quickly, which reduces the amount of error obtained. Similarly, in the non-linear section of the function, the quantum is quickly adjusted to a smaller size. Str2 error is large because each update the quantum size increases.

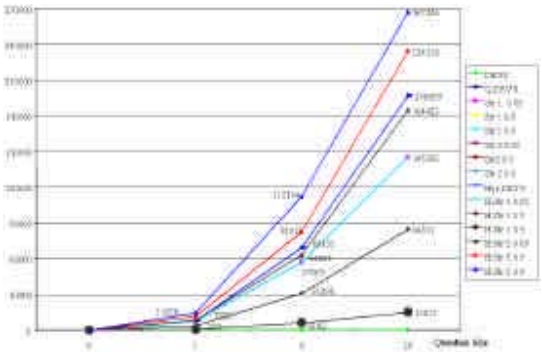

Figure 10. Heart tissue model: cumulative Error.

The number of messages is reduced up to a 99.95%. These empirical results verify the theoretical conclusions presented in [4], and reproduce the general shape of the message reduction found in [7]. Nevertheless, the amount of error involved was highly reduced.

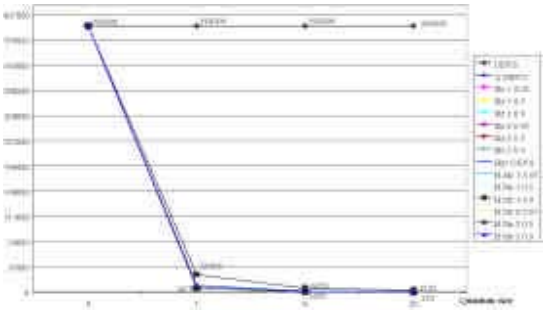

Figure 11. Heart tissue model: No. of messages.

Adaptive quantization improves the error when compared with Q-DEVS. Str2 expands the quantum size every time a threshold is not crossed, increasing the associated error. Str1 obtained the least improvement when compared with the rest, nevertheless, its overall execution time improves (6.57% vs. 1.58% for Q=1, and 0.8% vs. 0.06% for Q=20).

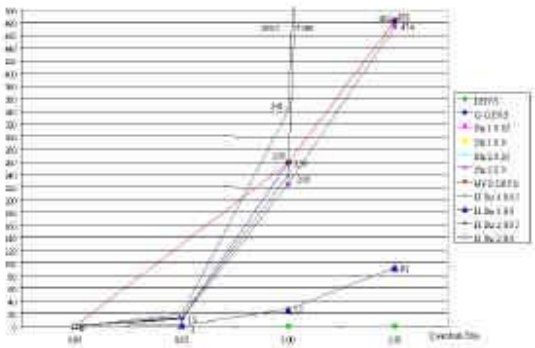

Figure 12. Watershed model Cumulative Error.

The Watershed model was tested using the land topology presented in Figure 5. In Figure 12, we present the cumulative error obtained when running this model.

These results repeated the pattern obtained for the heart tissue model: no difference between standard and hysteresis quantization (the watershed function has an increasing linear function for the topology and rain conditions chosen), the lowest error was obtained with Str1 with ratio 0.9, and Str1 results were better than Str2 and Q-DEVS. The order of the different strategies is maintained, while the total error is smaller.

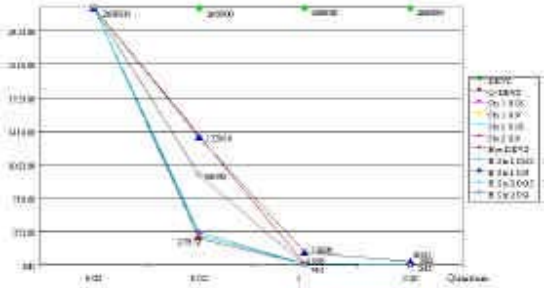

Figure 13. Q-DEVS Watershed model Output Messages

The best execution time was for Q-DEVS and Str1 with an update ratio of 0.05 (larger update ratios adjust the values quicker, reducing the error while increasing the number of messages). When using q=0.05, Q-DEVS provides better results. As each cell increases approximately 0.07 units in each update, changing the quantum size makes it oscillate around the function value, resulting in an increase in the total simulation messages. This does not occur with fixed quantum size. Likewise, once the quantum size varies, the dynamic quantum strategies have lower message interchange. A low update ratio improves the number of messages involved, while increasing the error. Paying a small cost in the extra execution overhead, we were able to reduce the error involved (up to 75%). Str2 reduces the number of messages using higher rates when compared to Str1, but incurring in a higher amount of error. If we consider, for instance, q=1 with Str1 and ratio 0.9, the amount of error introduced is minimum and the number of messages has been highly reduced. If we consider now q = 3.5, the error obtained with Str1 is better, while the number of messages involved is comparable.

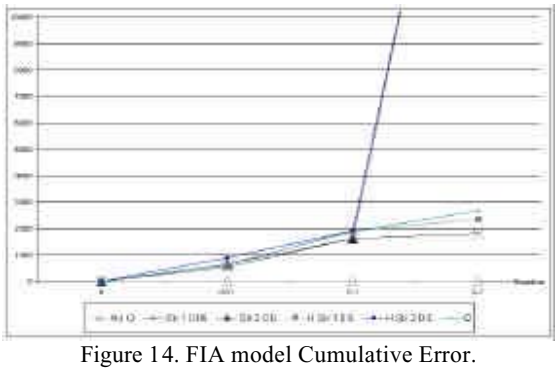

Figure 14. FIA model Cumulative Error.

In the FIA model, we observed a different error pattern: hysteresis quantum provided a more stable behavior. Str1 with a small update ratio improves the overall error because of the model behavior, presented in Figure

15. The diffusion combined with transport affect the results in cells closer/farther from the sample.

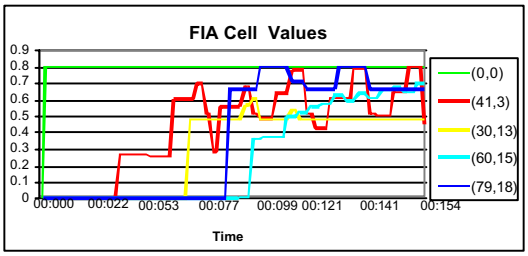

Figure 15. Individual cell behavior on the FIA model.

Here, Str2 with a larger update ratio always diverges. With Str1, a small update ratio improves the results (because it adjusts better to the different values).

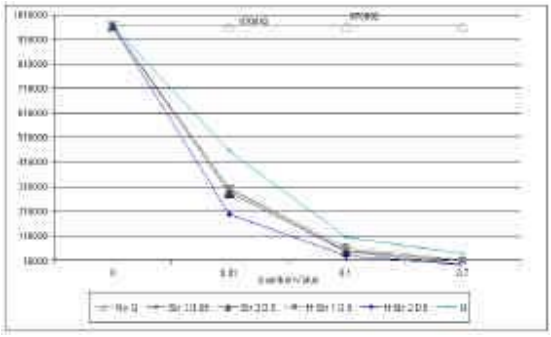

Figure 16. FIA model: number of messages.

The simulations with the highest error rate have provided the best execution times, as expected. Nevertheless, the amount of error obtained has highly reduced at the cost of little overhead.

CONCLUSION

We presented the analysis of quantization techniques for the execution of continuous variable Cell-DEVS models. We presented two different strategies for automatic updating of the quantum sizes in different cells. We obtained important reductions in the error obtained, while maintaining the high speed of quantized DEVS models. We used different complex models as a workbench, representing different categories of interest.

In every case, the lowest error was obtained with the dynamic quantum strategy 1. According to the model, updating the dynamic quantum size with higher/lower ratios improved the simulation results. In every case, the number of messages was reduced with quantization (up to 99.95% of reduction). Likewise, we could see that the introduction of hysteresis quantum permits to obtain a more controlled behavior, even for applications with cells executing with a non-linear pattern.

REFERENCES

[1] WOLFRAM, S. "A new kind of science". Wolfram Media, Inc. 2002.

[2] WAINER, G.; GIAMBIASI, N. "Timed Cell-DEVS: modelling and simulation of cell spaces". In "Discrete Event Modeling & Simulation: Enabling Future Technologies", Springer-Verlag. 2001.

[3] ZEIGLER, B.; KIM, T.; PRAEHOFER, H. "Theory of Modeling and Simulation". Academic Press. 2000.

[4] WAINER, G. "CD++: a toolkit to define discrete-event models". In Software, Practice and Experience. Wiley. Vol. 32, No.3. November 2002.

[5] ZEIGLER, B.P. "DEVS Theory of Quantization". DARPA Contract N6133997K-0007: ECE Dept., The University of Arizona, Tucson, AZ. 1998.

[6] KOFMAN, E.; JUNCO S "Quantized State Systems. A DEVS Approach for Continuous Systems Simulation". Transactions of the SCS. 18(3). 2001.

[7] WAINER, G.; ZEIGLER, B. "Experimental results of Timed Cell-DEVS quantization". In Proceedings of AIS'2000. Tucson, Arizona. U.S.A. 2000.

[8] N. GIAMBIASI, G. WAINER. "Using G-DEVS and Cell-DEVS to model complex continuous systems". Accepted for publication in Simulation: Transactions of the SCS. 2003.

[9] AMEGHINO, J.; WAINER, G. "Application of the Cell-DEVS paradigm using CD++". In Proceedings of the 32nd SCS Summer Computer Simulation Conference. Vancouver, Canada. 2000.

[10] TROCCOLI, A.; AMEGHINO, J.; IÑON, F.; WAINER, G. "A flow injection model using Cell-DEVS". Proceedings of the 35th IEEE/SCS Annual Simulation Symposium. San Diego, CA. 2002.

[11] HODGKIN, A.; HUXLEY, A. "A Quantitative Description of Membrane Current and its Application to Conduction and Excitation in Nerve" Journal of Physiology 117: 500-544. 1952.

[12] SAXBERG, B.; COHEN, R. "Cellular Automata Models of Cardiac Conduction". In Theory of Heart. Springer-Verlag. 1991.

[13] MOON, Y.; ZEIGLER, B.; BALL, G.; GUERTIN, D., "DEVS representation of spatially distributed systems: validity, complexity reduction". IEEE Transactions on Systems, Man and Cybernetics. pp. 288-296. 1996.

GABRIEL WAINER received the M.Sc. (1993) and Ph.D. degrees (1998, with highest honors) of the Universidad de Buenos Aires, Argentina, and Université d'Aix-Marseille III, France. He is Assistant Professor in the Dept. of Systems and Computer Engineering, Carleton University (Ottawa, ON, Canada). He was Assistant Professor at the Computer Sciences Dept. of the Universidad de Buenos Aires, and a visiting research scholar at the University of Arizona and LSIS, CNRS, France. He is author of a book on real-time systems and another on Discrete-Event simulation and more than 70 research articles. He is Associate Editor of the Transactions of the SCS. He is Associate Director of the Ottawa Center of The McLeod Institute of Simulation Sciences and a the chair of the SISO DEVS standardization Study Group.