**Real-time substorm**

**Hvara Ocar**

*Abstract:*

*A formalism for simulation called (*DEVS*)“Discrete Event System Specification” without approximation from differential equations is presented for the modeling of complex physical systems. A case of application of geomagnetic storm at real- time is studied.*

**I Introduction**

The aim of this study is to simulate the dynamics of magnetosphere- ionosphere coupling to

real-time monitor substorm solar activity. The model allows using observations of

WindNasa satellite to predict storm intensity.

The simulation involves the following steps:

* Entering data from wind satellites.
* Using the parameter of control BZ (component of interplanetary magnetic field). If BZ > 0, the processes of energy redistribution in the magnetospheric part do not occur because the conductivity of the ionosphere is equal to zero.
* When the simulation starts, we can appreciate how the contamination from magnetopause currents may occur.

**I I Background**

In order to understand how to detect a solar storm, OM equations are presented. The primary measure of the intensity of a geomagnetic storm is the strength of the ring current, as measured by *Dst*. Equations that relate the pressure-corrected *Dst* index to the solar wind driver given by *VBs,* where *VBs* is the rectified value of *VBz* that is positive when Bz is southward and zero when Bz is northward. Such equations are:

(1) = Q (

(2) Q ( =

(3) (

Where Q represents injection into the ring current minus , a loss term represented by the

recovery time , which depends on the strength of the ring current and is assumed to be

proportional to *Dst.[[1]](#endnote-2)*

Minimum *Dst*\* occurs when

(4) = 0

and hence ,

( 5) Q

we can model a solar storm through a cellular automaton.[[2]](#endnote-3) In the model, the state of each cell is described by two numbers: one corresponding to the energy content in a region of the current sheet in the magnetospheric tail and the other to the conductivity in the ionospheric domain, which is magnetically connected with such region.

The ionosphere conductivity is proportional to the electron density and the differential equation for electron density is:

= Q –

where Q is the ionization rate and is an effective recombination coefficient. The model of cellular automaton describes rules of transition between these situations.

**III The formal model**

The model is specified using CD++ specification language. The cellular automaton is divided into four planes corresponding to the variables *E* (energy in the magnetosphere), *E*max (maximal energy), *E*min (minimal energy) and *C* (conductivity of the ionosphere). The cell space shape and size and the rules that govern the model execution are defined in the solarwind.ma file. The first portion of the coupled model defines the cell-space geometry and initial values, as shown in Figure 1:

|  |
| --- |
| [top]  components : genSustancias@NodeGenerator system  link : aout@genSustancias in@system  [genSustancias]  distribution : poisson  mean : 3  [system]  in: in  link : in in@system(0,0,1)  link : in in@system(15,0,1)  type : cell  dim : (16,16,4)  delay : transport  defaultDelayTime : 100  border : wrapped  neighbors : system(-1,-1,0) system(-1,0,0) system(-1,1,0)  neighbors : system(0,0,-2) system(0,-1,0) system(0,0,0) system(0,1,0) system(0,0,2)  neighbors : system(1,-1,0) system(1,0,0) system(1,1,0)  neighbors : system(-1,-1,1) system(-1,0,1) system(-1,1,1)  neighbors : system(0,-1,1) system(0,0,1) system(0,1,1)  neighbors : system(1,-1,1) system(1,0,1) system(1,1,1)  neighbors : system(-1,-1,-1) system(-1,0,-1) system(-1,1,-1)  neighbors : system(0,-1,-1) system(0,0,-1) system(0,1,-1)  neighbors : system(1,-1,-1) system(1,0,-1) system(1,1,-1)  initialvalue : 0  zone : Energy { (0,0,0)..(15,15,0) }  zone : Energy\_max { (0,0,1)..(15,15,1) }  zone : Energy\_min { (0,0,2)..(15,15,2) }  zone : Conductivity { (0,0,3)..(15,15,3) } |

**Figure 1. Coupled model specification**

As shown in Figure 1, the cell space is composed of 16x16x4 cells divided into four zones: zone : Energy for the variable *E* (energy in the magnetosphere), zone : Energy\_max for the variable *E*max (maximal energy), zone : Energy\_min for the variable *E*min (minimal energy) and zone : Conductivity for the variable *C* (conductivity of the ionosphere). Each zone is 16x16.

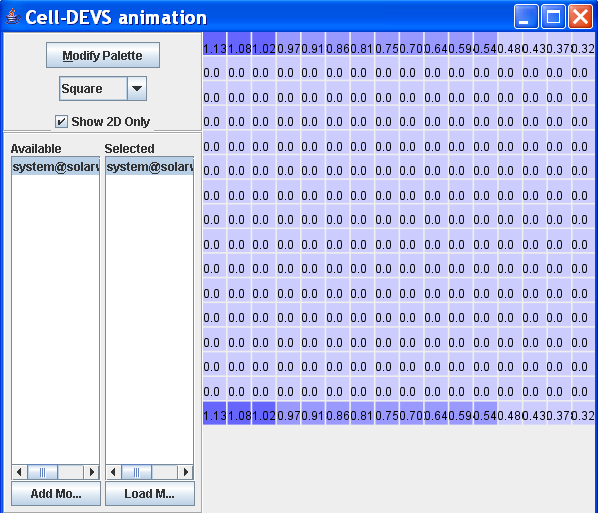
Figure 1 also shows that a component of the model is NodeGenerator. This component is implemented by an atomic model called generat.h which obtains the *Bz* value from the web page and sends this value to the main simulation. PHP 5.3 (5.3.20) is required to run the model because generat.h calls a php script called parser.php, which obtains the data from the web.

|  |
| --- |
| [Energy]  rule : {(0,0,0) + 0.25\*(-1,0,0) - 0.25\*(0,0,2) } 100 { (-1,0,0)>= #macro(E\_max) and cellpos(0)!=0}  …  [Energy\_max]  rule : {(0,-1,0)} 100 {(0,0,0)=0 }  …  [Energy\_min]  rule : {#macro(E\_min)} 100 { (0,0,0)=0 }  …  [Conductivity]  rule : {0.2\*(0,0,0)} 100 {(0,0,1) < #macro(E\_max) }  … |

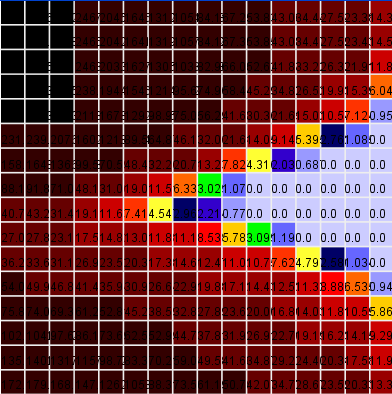
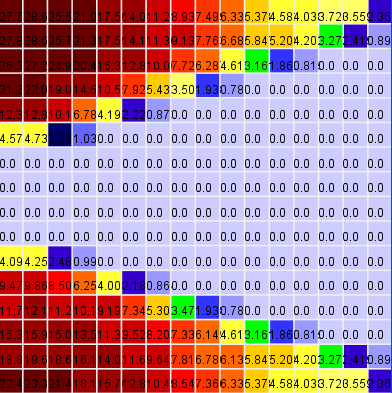
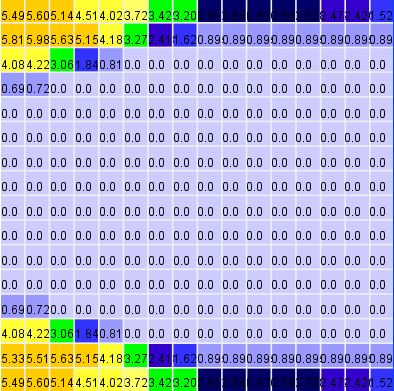
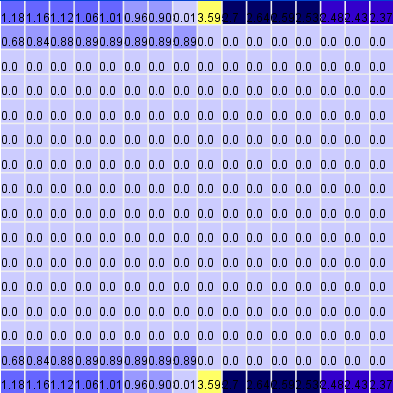
**Figure 2. Rule definitions**

Figure 2shows part of the rule definition. The[Energy] rule implements the distribution of energy in the magnetosphere. The [Energy\_max] rule implements the distribution of the *Bz* value along the boundaries of the cell space. The [Energy\_min] rule governs the minimal energy of each cell. Finally, the [Conductivity] rule implements the conductivity of the ionosphere.

Although there are four variables, we are only interested in the behavior and evolution of the *E* variable, which is shown in Figure 3.

****

**Figure 3. Initial distribution of energy *E* along the boundaries**

****

**Figure 4. Evolution of energy *E***

**IV Testing**

In the following, we test the rules (1), (2), (3) and (4), assuming a simple testing scenario where *Bz*=-3 and *E*min(i,j)= 0.013.

**Rule (1)**

When the threshold level *E*max(i,j) is exceeded, a certain part of the stored energy, Δ*E* = *E*t(i,j)-*E*min(i,j), is distributed between four adjacent cells:

*E*t+1 (i,j) = *E*min(i,j),

*E*t+1 (i+1,j) = *E*t (i+1,j) + 0.25Δ*E,*

*E*t+1 (i-1,j) = *E*t (i-1,j) + 0.25Δ*E,*

*E*t+1 (i,j+1) = *E*t (i,j+1) + 0.25Δ*E,*

*E*t+1 (i,j-1) = *E*t (i,j-1) + 0.25Δ*E.*

This rule is implemented as follows. Let

* (0,0,0) be the *E*t(i,j) value,
* (0,0,2) be the *E*min(i,j) value,
* #macro(E\_max) be the *E*max(i,j) value,
* (-1,0,0), (1,0,0), (0,1,0), (0,-1,0) be the *E*t(i,j) where the *E*max(i,j) value is reached.

Then, the rules are:

|  |
| --- |
| rule : {(0,0,2)} 100 { (0,0,0)>= #macro(E\_max)}  rule : {(0,0,0) + 0.25\*(-1,0,0) - 0.25\*(0,0,2) } 100  {(-1,0,0)>= #macro(E\_max) and cellpos(0)!=0}  rule : {(0,0,0) + 0.25\*(1,0,0) - 0.25\*(0,0,2) } 100  {(1,0,0)>= #macro(E\_max) and cellpos(0)!= (-1+#macro(CantFilas)) }  rule : {(0,0,0) + 0.25\*(0,1,0) - 0.25\*(0,0,2) } 100  {(0,1,0)>= #macro(E\_max) and cellpos(1)!= (-1+#macro(CantFilas))}  rule : {(0,0,0) + 0.25\*(0,-1,0) - 0.25\*(0,0,2) } 100  {(0,-1,0)>= #macro(E\_max) and cellpos(1)!=0} |

The drawlog of the simulation shows that *E*(i,j) has reached –Bz at position (0,0) of the following grid:

|  |
| --- |
| 0 1 2  +-------------------------------  0| 3.060 3.000 2.940  1|  2| |

In the next step of the simulation, we see that position (0,0) is set to *E*min(i,j)=0.013 and

Δ*E* =0.762 is added to the neighbors (0,1) and (1,0). Thus, the implementation works:

|  |
| --- |
| 0 1 2  +-------------------------------  0| 0.013 3.762 3.000  1| 0.762  2| |

**Rule (2) and (3)**

For conductivity in discrete form, we have the rules:

*C*t+1 (i,j) = *aC*t(i,j) + *b* (2)

*b* = Δ*E* = *E*t(i,j)-*E*min(i,j), for *E*t(i,j) ≥ *E*max(i,j) (3)

*b* = 0 , for *E*t(i,j) < *E*max(i,j)

These rules are implemented as follows. Let

* (0,0,0) be the *C*t(i,j) value,
* (0,0,1) be the *E*t(i,j) value,
* (0,0,-1) be the *E*min(i,j) value.

Then, the rules are:

|  |
| --- |
| rule : {0.2\*(0,0,0)} 100 {(0,0,1) < #macro(E\_max) }  rule : {0.2\*(0,0,0) + (0,0,1) - (0,0,-1) } 100 {(0,0,1) >= #macro(E\_max) } |

Suppose that the *E*max(i,j)=3 is reached at the position (0,0) of the *E*t zone. Then, the following drawlog of the simulation shows that *C*(i,j) is increased, according to rule (3), from 0 to Δ*E* =3.060-0.013 = 3.047:

|  |
| --- |
| 0 1 2 0 1 2  +------------------------------- +-------------------------------  0| 3.060 3.000 2.940 0| 3.047  1| 1|  2| 2|  *E*t zone *C*t zone |

In the next step of the simulation, the second part of rule (3) is applied. We see that the position (0,0) of the *C*t zone in set to *a*\**C*t(i,j)=0.2\*3.047=0.609

|  |
| --- |
| 0 1 2 0 1 2  +------------------------------- +-------------------------------  0| 0.950 0.013 3.937 0| 0.609 3.749  1| 0.762 0.937 1|  2| 2|  *E*t zone *C*t zone |

The other values are set by the application of rules (1) and (2). Thus, the implementation works.

**Rule (4)**

If the ionospheric conductivity exceeds *C*max=5, then the conditions of energy redistribution in the current sheet are changed as follows:

*E*min, t+1 (i,j) = *k E*max (i,j) for *C*t (i,j) *< C*max(4)

*E*min, t+1 (i,j) = 0 for *C*t (i,j) *≥ C*max

This rule is implemented as follows. Let

* k= 0.75
* (0,0,1) be the *C*t (i,j) value.

Then,the rules are:

|  |
| --- |
| rule : {0.75 \* #macro(E\_max) } 100 { (0,0,1) < #macro(C\_max) and (0,0,1)!=0}  rule : 0.0001 100 { (0,0,1) >= #macro(C\_max) } |

In the first rule, we add the condition (0,0,1)!=0 to distinguish the initial case where *C*t (i,j) is zero as default value.

We consider the drawlog of the previous rule where the position (0,0) of the *C*t zone is 0.609. Since this value is different from zero and *< C*max , then applying rule (4), the position (0,0) at the *E*min zone is set to *E*min (i,j) = *k E*max (i,j) = 0.75\*3 = 2.25.

|  |
| --- |
| 0 1 2 0 1 2  +------------------------------- +-------------------------------  0| 2.250 0.013 0.013 0| 0.609 3.749  1| 0.013 0.013 0.013 1|  2| 0.013 0.013 0.013 2|  *E*min zone *C*t zone |

The second part of rule (4) is confirmed in a similar way. To see details, we refer to the complete drawlog file.

**V Conclusion**

Here, we demonstrated how to use DEVS to model physical complex systems at real time. In particular, we studied the simulation for a solar storm. Thus, considering that the geomagnetic activity is expired for the period 2013-2014 further studies should be carried out to validate the model for different levels of solar substorms. Furthermore, other variables of impact in the north-west hemisphere should be studied associated with the contamination of the magnetopause.

1. Journal of Geophysical Research , XXXX, DOI : 10.1029 “Major geomanetic storms (Dst nT) generated by corotating interaction regions”. I.G.Richardson, D.F.Webb, J.Zhang,D.B.Berdichevsky,D.A.Biesecker, J.C.Kasper,R.Kataoba, J.T.Steinberg, B.J.Thompson, C.-C.Wu, and A.N Zhukov [↑](#endnote-ref-2)
2. Annales Geophysicae (2003) 21:1931 – 1938 c European Geosciences Union 2003 “ Cellular Automata model of magnetospheric –ionospheric coupling.” B.V.Kolesov and T.V.Kolesova. [↑](#endnote-ref-3)