Implementing the SCIDDICA Landslide Model in Cell-DEVS

Brian Webb
Dept. of Systems and Computer Engineering
Carleton University
1125 Colonel By Drive
Ottawa, ON. K1S 5B6, Canada
Email: bwebb@connect.carleton.ca
Phone: 613-520-2600 ext 3018

Gabriel A. Wainer
Dept. of Systems and Computer Engineering
Carleton University
1125 Colonel By Drive.
Ottawa, ON. K1S 5B6. Canada
Email: gwainer@sce.carleton.ca
Phone: 1-613-520-2600 ext 1957

Keywords: Discrete event simulation, DEVS, Cell-DEVS, cellular automata, landslide modelling

Abstract

The SCIDDICA model is a custom cellular automata used to simulate and analyse landslides and debris flows. It models landslides by tracking the amounts of kinetic energy and debris in the slide and using a set of simple rules to determine the movement of the kinetic energy and debris from cell to cell.

CD++ is a toolkit for implementing DEVS and Cell-DEVS models. We present an attempt to implement SCIDDICA using DEVS and CD++. A DEVS based definition of the SCIDDICA model introduces the advantages of the DEVS formalism to the model, and would allow formal verification of the DEVS definition of the model. This also permits us to integrate a SCIDDICA based landslide with other DEVS models, allowing larger and more complex simulations to be developed.

1. INTRODUCTION

SCIDDICA, the Simulation through Computational Innovative methods for the Detection of Debris flow path using Interactive Cellular Automata is a custom cellular automata that is being developed by the University of Calabria in Italy.[1]

This paper deals with the SCIDDICA S3-Hex version of the model[1] as the most detailed explanation of the SCIDDICA transition rules were available for it.

CD++ is a programming toolkit designed to implement models and cellular automata as described by the DEVS modeling formalism. CD++ has a library of built in functions to handle mathematical, logical and neighborhood operations. It is also possible to define custom functions for use in CD++ code by adding those functions to specific C++ source files for the CD++ simulator. The modified C++ source files are kept separate from the original source files, and stored along with the CD++ source files so that other projects on the same system still have access to the original simulator.

The three goals of this project were:

- To implement a working version of the SCIDDICA model in Cell-DEVS.
- To more accurately implement the SCIDDICA debris outflow rules by creating custom C++ functions for them.
- To simplify the Cell-DEVS implementation to make it easier to understand.

2. SCIDDICA DEFINITIONS

Landslides appear to be a good fit for cellular automata because the Navier-Stokes equations that describe debris flows cannot easily be solved without simplifications. Much of the complexity comes from irregular topography and changes in the nearly Newtonian fluids of the flows as water is lost[2].

SCIDDICA is one attempt to model landslides as cellular automata and has been successfully used to simulate the 1992 Tessina[3] and 1984 Ontake[4] slides. It is a deterministic model, and the S3-Hex version uses six parameters and eleven state variables to describe the terrain the slide will be travelling over and the slide itself.

The six parameters are constant for a given simulation but will vary depending on the location being simulated and the conditions at that location. These parameters are[1-2]:

- Ptime - The discrete time step taken by the cellular automata.
- Padhesion - The immovable amount of debris.
- Pf - Hieght threshold of the outflows, related to the friction angle.
- Prunuploss - Frictional energy loss factor.
- Pmt - Mobilisation threshold for the soil cover.
- Perosion - Progressive erosion of the soil cover.

There are eleven state variables using an hexagonal topology[1]. These variables are:

- Altitude: The combined height of the erodible and non-erodible material in a cell.
- Erodible Soil Depth: The thickness of the erodible material in a cell.
- Energy: The kinetic energy (KE) of the debris in the cell.
- Debris Thickness: How deep the debris is in the cell.
- Outflows: How much debris goes to each cell in the neighborhood, eight values.
- Kinetic Runup: Used to calculate the energy and outflows.

The rules and expressions used by SCIDDICA S3-Hex are[1]:

- Δd: The amount of soil eroded from a cell by the debris. 0 if there is no soil.
- Δr: The change in runup energy. Equals the runup loss parameter (Prunuploss) if (k* energy/debris - Prunuploss) $>$ debris otherwise equals (k*energy/debris) - debris.
- k: 2 / (rho * g * A) : rho = debris density, g = gravitational acceleration, A = cell area

Altitude: The altitude changes if soil was eroded from the cell.

$$newaltitude = oldaltitude - \Delta d$$

Erodible Soil Depth: The soil depth changes if there is both debris and energy in the cell, as the moving debris will erode more and more soil over time.

$$newdepth = olddepth - \Delta d$$

Energy: The energy changes as the debris picks up more mass from eroding soil, gaining momentum and lost due to friction. There are three rules for calculating the energy of a cell in SCIDDICA.

Energy lost to debris outflow, plus energy gained from inflow:

$$E_{new} = (Debris - \sum Outflow_i) * (\frac{E}{Debris}) + \sum (inflow_i * \frac{E_i}{Debris_i})$$

Energy gained from new mass of soil eroded:

$$E_{new} = k * (Debris + \Delta d) * (KineticRunup + \Delta d)$$

Energy lost to friction:

$$E_{new} = k * \Delta r * Debris$$

Debris Thickness: Debris thickness changes as the debris flows into and out of a cell, and as the soil in the cell is eroded.

$$Debris_{new} = Debris + \sum Inflow_i - \sum Outflow_i + \Delta d$$

Kinetic Runup: The runup is a function of the energy of the cell, the debris thickness and the constant k.

$$runup = k * \frac{E}{Debris}$$

Debris Outflow to cell i: The amount of debris that flows from the central cell to cell i in the neighborhood. The SCIDDICA rules for calculating outflow:

1. If the tangent of the angle between the current cell and cell i is greater than pf add cell i to set A.
2. Calculate the average outflow of all cells in A

$$Avg = (k * Energy * Debris - Padhesion) + \sum (Altitude[j] + Debris[j]) / Numberofcellsin A$$

3. For all cells in A if $(Altitude[i] - Debris[i]) > Avg$ remove cell i from A
4. If A has changed go back to step 2.
5. If cell i is in A $Outflow_i = Avg - (Altitude[i] - Debris[i])$
6. Otherwise $Outflow_i = 0$

3. CD++

The SCIDDICA parameters were defined in CD++ as macros, allowing them to be easily changed between simulations if necessary.

The current version of CD++ only allows for a single state variable per cell, and the cell space must be a square topology. To store the 11 variables for SCIDDICA's hexagonal topology a 3D CD++ cells space was defined, using the cells in the vertical dimension as the SCIDDICA variables.

The rules were modified to assume a square topology, which is the only topology 3D CD++ cell spaces support. This increased the number of state variables from eleven to thirteen. The neighborhood size was increased by two cells at each level so two more outflow variables were needed.

The cell neighborhood is, therefore, very large, consisting of 25 layers of 9 cells each, or 225 individual cells. The large neighborhood is necessary because CD++ defines the neighborhood relative to the current cell of interest. Cells in layer 0 need to access cells in layer 12, therefore, the neighborhood must reach up +12 cells. And cells in layer 12 need to access

cells in layer 0, which requires that the neighborhood must reach down -12 cells.

For the same reason the neighborhood must also be a square of nine cells over all 25 of its layers. Even though some variables, such as kinetic runup, depend only on cells in their own column.

All the SCIDDICA rules, except for the outflow calculations were implemented fully in CD++. These rules were simple translations of the algebraic SCIDDICA rules with some additional functionality to handle cells located on map edges. These repeated checks to see if the neighboring cells exist account for much of the bulk and complexity of the implemented rules.

The outflow rules were not fully implemented because they require iteration and temporary variables, neither of which the CD++ grammar permits. It was determined that custom functions would be needed to properly implement the outflow rules.

For testing purposes a simplified version of the SCIDDICA rules was created by removing the iteration and using the initial unrefined average outflow value. If the tangent of the angle between the current cell and the outflow cell was greater than pf, the initial average outflow was calculated and the outflow amount determined from the equation, $Outflow[i] = Average - (Altitude[i] - Debris[i])$.

This allowed us to simulate slides, but under this rule it is possible to outflow more debris from a cell than is present.

Results for adding custom functions to the simulator were mixed. Simple, two argument functions were added successfully to the simulator and could run properly. Those functions were: Δd, altitude change, and soil erosion.

However, algorithms which require more than two arguments could not be implemented. The CD++ parser and grammar support generic functions of up to two arguments only and most of the SCIDDICA functions take four or more arguments.

4. IMPLEMENTATION

The Cell-DEVS implementation of SCIDDICA was done over the course of a month and a half. Starting with translating the SCIDDICA variables, parameters and equations into the DEVS formalism.

Once the formalism was completed the transition rules of the formalism were implemented as rules in CD++. The greatest challenge in converting the transition rules was in adding the boundary condition checks. It was necessary to decide what would happen to a slide if it reached one of the edges of a map as this was not covered in the SCIDDICA papers [1-2].

In CD++ there are two ways to treat cell space edges. They can be treated as solid, impassible walls, or they can be treated as if the cell space were a cylinder. With this second method the cell space wraps around, so that moving past one edge places you on the opposite edge.

We decided it was better to treat the edges of a cell space as walls, rather then allow them to wrap around. Real world locations do not wrap around and we believed that simulated slides would be more accurate and usefull with wall-like edges than esges which allowed a slide to go off one edge of the cell space and reappeare elsewhere.

5. EXPERIMENTAL RESULTS

The values for the SCIDDICA parameters were taken from a simulation of the 1992 landslide in Tessina, Italy[2].

- Ptime = 0.1 seconds
- Padhesion = 1 dm.
- Pf = 25 deg.
- Prunuploss = 2.
- Pmt = 0.3
- Perosion = 0.6

Four maps ten by ten were defined for testing the model. Tests were run five times each for increasing initial values of KE and debris thickness.

The first two maps were a map with a shallow slope running from top to bottom, a map with a steep slope running from top to bottom. The shallow slope was at 1 meter intervals per cell, the steep 5 meter intervals. The purpose of these maps is to provide a simple test environment.

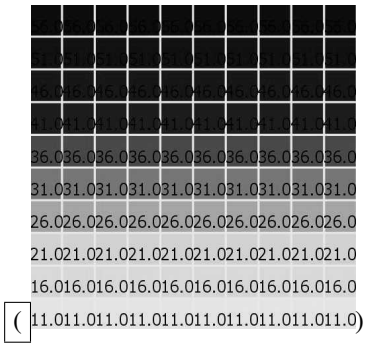

Figure 1. A simple slope

Starting a slide at the top of the slope for the shallow sloped map with KEs under four did not produce a slide. Slides produced by KEs of four and over went down the slope, but tended to slide to the left. An initial KE of two was enough to start a slide on the steep sloped map. The slides were faster, but behaved in the same way as the shallow sloped map.

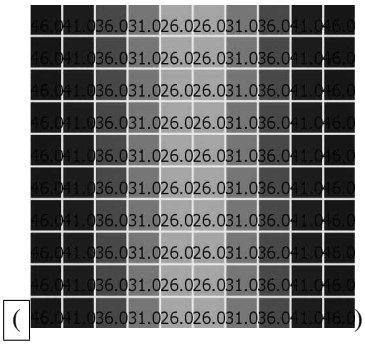

Figure 2. A valley

The third map was a valley whose slopes were at 5 meter intervals. The purpose of this was to test the actions of the model when a slide encountered a reverse slope.

Starting two slides on either side of the valley in the valley map produced two fan shaped slides that collided in the valley and then spread out to fill the valley. However, slides going right to left moved slightly faster than slides going from left to right.

The fourth map a complex one with several peaks and a U shaped valley. There are peaks in the upper and lower left corners, a peak midway along the upper edge and a peak midway along the right edge, which defines the U. The lowest areas of the map are in the upper and lower right corners. The slopes are in intervals of 10 meters. Its purpose was to see if the model could produce realistic results on complex terrain.

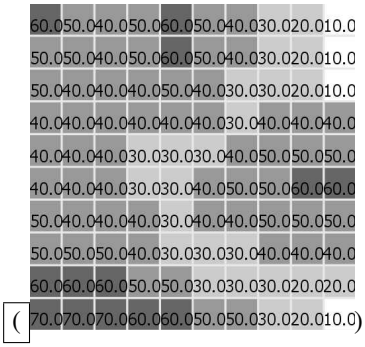

Figure 3. A more complex map

Slides on the more complex map behaved as expected, flowing down slope to the U valley and spreading out along the low areas. Slides on this map tended to move faster towards the top of the map.

On all maps the higher the initial KE and the steeper the slope the faster the slide travelled. The biases in the slide directions are incorrect but without being able to use the full outflow rules we could not determine if it was caused by the incomplete outflow rules, the shift from hexagonal to square topology or another factor.

6. CONCLUSIONS

The Cell-DEVS formalism and CD++ allowed us to build a working implementation of a complex cellular automata in a short time period, with limited resources. The SCIDDICA rules and variables translated easily into the Cell-DEVS formalism. And with the exception of the outflow rules, were easy to implement in CD++ once this had been done.

The working implementation with the partial outflow rules does allow us to simulate landslides. These landslides behave in a predictable and realistic manner. This gives us a useable, if imperfect, model that can be connected to other DEVS and Cell-DEVS models, allowing us to simulate larger and more complex systems.

7. REFRENCES

[1] D'Ambrosio, D. et al,"Simulating debris flows through a hexagonal cellular automata: SCIDDICA S3-hex", *Natural Hazards and Earth System Sciences*, vol. 3, pp. 545-559, 2003.

[2] Calidonna, C. R. et al, "A Network of Cellular Automata for a Landslide Simulation", *Proceedings of the 15th international conference on Supercomputing ICS '01*, 2001.

[3] Avoliol, M. V. et al, "Simulation of the 1992 Tessina landslide by a cellular automata model and future hazard scenarios", *International Journal of Applied Earth Observation and Geoinformation*, vol. 2, no. 1, pp. 41-50, 2000.

[4] Di Gregorio, S. et al, "Mount Ontake Landslide Simulation by the Cellular Automata Model SCIDDICA-3", *Physics and Chemistry of the Earth, Part A: Solid Earth and Geodesy*, vol. 24, no. 2, pp. 131-137, 1999.