GIS AND SIMULATION, ENVIRONMENTAL SIMULATION

Pau Fonseca i Casas, pau@fib.upc.edu
Introduction

Cellular automaton

Intelligent Agents

Formalization

Examples
Discrete simulation

A brief description of the simulation engine
Event Scheduling 1/2

- System modifications only occur in certain time instants.
  - Changes determined by the incoming events.
- Allow detailed model construction.
- Allow different paradigm combination.
Event Scheduling 2/2

- Event Scheduling: Sample M|M|S

![Diagram of Event Scheduling]

- Server (Cn, Cn+1, Cn+2)
- Queue (τn, τn+1, τn+2)
- Time (s, w, x, n+1, n+2)
- Transition Times (tn, tn+1, tn+2)
Simulation engine

Simulation model (XML & VRML data)

LeanGen or SDLPS simulation engine

Simulation server (TCP/IP)

Graphical interface

2D render engine

3D render engine

Statistical engine

Introduction - Cellular automata - Intelligent agents - Formalization - Examples
Simulation client

- Simulation model (XML & VRML data)
- Simulation client (TCP/IP)
- Graphical interface
- 3D render engine

LeanTraining
Training client

Simulation model (XML & VRML data)

Simulation client (TCP/IP)

Sub-model logic

Graphical interface

3D render engine

INPUT Sensors  OUTPUT Sensors
Agent client

LeanGen

TCP/IP

Logic

Virtual representation

Out

In
GIS data structure and classification.
GIS data
GPS layers integration

<table>
<thead>
<tr>
<th>Layer</th>
<th>GPS integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEM</td>
<td>Geo referenced</td>
</tr>
<tr>
<td>2DLayers</td>
<td>Geo referenced</td>
</tr>
<tr>
<td>3DLayers</td>
<td>Geo referenced</td>
</tr>
<tr>
<td>Routes</td>
<td>Track points</td>
</tr>
<tr>
<td>2DObjects</td>
<td>Waypoints</td>
</tr>
<tr>
<td>3DObjects</td>
<td>Waypoints</td>
</tr>
</tbody>
</table>

- Integration of some Objects or layers with a GPS.
- Enables the training systems
2D Layers (Vectorial layers)

- Point, polylines, texts or lines.
- Usually represent information for the observer, but not represent information for the simulation model (don’t have any specific behaviour).
3DLayers (Rasters Layers)

- To represent a fixed population of elements
  - Forest or a city (each tree has his own position over the DEM).
- All the elements have the same virtual representation and the same behaviour.
Objects

- To represent element with an individual or concrete behaviour (2DObjects or 3DObjects).
  - **2DObjects**: lines polylines, texts and points with an individual behaviour, (a point with a touch sensor that shows some information).
  - **3DObjects**: virtual objects that are not associated with any layer. These objects can have own behaviour represented by a script or program connected with the world through EAI.
Routes

- The routes define 3DObjects movements through the virtual Landscape.
- These routes can be defined by the simulator.
- Enables the possibility of move different elements through to Landscape following the simulator logic.
Why use GIS data?

A classification of the models
## Interactive/evolutionary models

<table>
<thead>
<tr>
<th></th>
<th><strong>Static models</strong></th>
<th><strong>Dynamic models</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Interactive</strong></td>
<td>Static interactions. The only changes are in the composition. For instance, systems that are not modified over time. Through simulations, an approximate value can be obtained.</td>
<td>System interaction. Changes in the interactions between the different model components. For instance, an industrial plant.</td>
</tr>
<tr>
<td><strong>Evolutionary</strong></td>
<td>Evolutionary selection. Random acquisition of variations that change the composition of types.</td>
<td>Evolutionary system feedback that influences the supply of variation and the speed of evolution. Changes in type depend on the history of the system. For instance, the evolution of a society or wildfire with the interaction of an extinction model.</td>
</tr>
</tbody>
</table>

*(Henning 2001)*
### MAS

<table>
<thead>
<tr>
<th>Static</th>
<th>Dynamic</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Relation with the environment</strong></td>
<td><strong>Relation with the environment</strong></td>
</tr>
<tr>
<td><strong>Discrete</strong></td>
<td><strong>Continuous</strong></td>
</tr>
<tr>
<td><strong>Strong</strong></td>
<td><strong>Autonomat</strong></td>
</tr>
<tr>
<td><strong>Semi-Isolated Evolution</strong></td>
<td><strong>Complex Interactions</strong></td>
</tr>
</tbody>
</table>

(Márín and Mehandjievi 2006)
Using GIS in Simulation models

- Allows environment modeling.
- Dynamical use of the GIS data.
  - Dynamical modification of the GIS data.
  - Dynamical acquisition of the GIS data.
Cellular automata

Modeling the environment
Cellular automaton

- Simplifies the use of environment in a simulation model.
Cellular automaton

- Structure based in:
  - Set of rules.
  - Matrix of data.

- Modify the matrix following the set of rules.

![Cellular automaton example](image)
The **Game of Life** is a cellular automaton devised by the British mathematician **John Horton Conway** in 1970. It is the best-known example of a cellular automaton.

- Glider gun and glider
Cellular automaton

- Only one matrix of data.
- Only one set of rules.
- The space is discrete.
- The space of states can be huge.
- Not all the states can be reachable → maps of states space.
Extending the capabilities of the cellular automata
m:n-CA^k

- A multi n dimensional cellular automaton is a cellular automaton generalization composed by m layers with n dimensions each one.

- **Aim:**
  - Allows multiple layers.
  - Allows vectorial layers (continuous space).
  - Allows multiple set of rules (evolution functions).
The representation is:

\[ m : n - A C^k \]

Where

- \( m \): is the automaton number of layers.
- \( n \): is the different layers dimension.
- \( k \): is the number of main layers (1 by default).
m:n-CA^k

- Defined over the mathematical topology concept.
- 1:n-AC\(^1\) is the common cellular automata if the topology used is the discrete topology defined over N or Z.
- The implementation, as is usual, can be a matrix.
State of the automata

- $E_m[x_1,..,x_n]$, layer m state in $x_1,..,x_n$ position
  - $E_m$ is a function describing cell state in position $x_1,..,x_n$ of layer m.

- $EG[x_1,..,x_n]$, automata status in $x_1,..,x_n$ position.
  - $EG$ returns automata global state in position georeferenced by coordinates $x_1,..,x_n$.

$$
\Psi(E_1[x_1..x_n], m^{-2}, E_m[x_1..x_n]) = EG[x_1..x_n]
$$
Evolution function $\Lambda_m$

- Evolution function allows global automaton state change through cells value modification.
- Defined for the layer $m$ to modify its state through the state of others layers using combination function $\Psi$, and vicinity and nucleus functions.
- Is only defined in main layers.
Evolution function $\Lambda_m$

- Function defined for the layer $m$ to modify its state through the state of others layers using combination function $\Psi$, and vicinity and nucleus functions.

- A $m:n$-AC automaton only presents one main layer, an $m:n$-AC$^k$ automaton presents $k$ main layers.
Vicinity topology

- Topology defining the set of points (neighbourhood) for layer $m$, to be considered for $\Lambda_m$ calculus.
- Vicinity function $v_n(x_1,..,x_n)$:
  - Function returning minimum open set of vicinity topology containing point $x_1,..,x_n$, and including maximum points that accomplishes the restriction and minimum points not accomplishing the restriction.
Nucleous topology

- Topology defining the set of points (neighborhoods) for layer \( m \), to be modified by \( \Lambda_m \) calculus
- Nucleus function \( nc(x_1,..,x_n) \)
  - Function returning minimum open set of nucleus topology containing point \( x_1,..,x_n \), and including maximum points that accomplishes the restriction and minimum points not accomplishing the restriction.
Vicinity function example

- **Over Z**
  \[ vn(x_1, \ldots, x_n) = \{(x_{1-1}, x_{2-1}, x_{n-1}), (x_{1-1}, x_{2-1}, x_n), (x_{1-1}, x_{2-1}, x_{n+1}), (x_{1-1}, x_2, x_{n-1}), (x_{1-1}, x_2, x_n), (x_{1-1}, x_2, x_{n+1}), (x_{1-1}, x_{2+1}, x_{n-1}), (x_{1-1}, x_{2+1}, x_n), (x_{1-1}, x_{2+1}, x_{n+1}), (x_{1+1}, x_{2-1}, x_{n-1}), (x_{1+1}, x_{2-1}, x_n), (x_{1+1}, x_{2-1}, x_{n+1}), (x_{1+1}, x_{2+1}, x_{n-1}), (x_{1+1}, x_{2+1}, x_n), (x_{1+1}, x_{2+1}, x_{n+1}) \} \]

- **Over R**
  \[ vn(x_1, \ldots, x_n) = \text{returns the open set centered in the point } x_1, x_2, x_3 \text{ for the topology that defines the vicinity.} \]
  \[ B(x, r) = \{ y \in \mathbb{R}^m / d(x, y) < r \} \]
Nucleus function example

- **Over Z**
  \[ nc(x_1,\ldots,x_n) = \{(x_1,\ldots,x_n)\} \]

- **Over R**
  \[ nc(x_1,\ldots,x_n) = \text{returns the open set centered in the point } x_1,x_2,x_3 \text{ for the topology that defines the nucleus.} \]
Intelligent agents

Reacting to the environment
Intelligent agent

- An structure reacting to the environment through his actions, and perceiving the environment through the sensors.
Reflexive intelligent agent
Model based reflexive agents

Agent
World evolution?
Actions influence?
State

Sensors
World STATUS NOW?

What to do NOW?
Condition/action RULES

Efectors

Environment
Goal based agents

- Agent
  - World evolution?
  - Actions influence?
  - State

- Sensors
  - World STATUS NOW?
  - What happens doing A?
  - What to do NOW?

- Efectors

- Environment
Utility based agents

- Agent
  - World evolution?
  - Actions influence?
  - State
- Sensors
  - World STATUS NOW?
  - What happens doing A?
  - I feel better?
  - What to do NOW?
- Effectors
- Environment
A brief note of how to formalize simulation models
Why?
Why?

Obtaining answers, data from the executions of the model.

Implementing the results?
Alternatives

- DEVS, Petri Nets, SDL, SysMPL, UML,...
- What is the tool we have?
- What is the personnel involved in the project?
- We have components to reuse?
- Are this formalisms and languages ready to represent the needed structures for our models?
Tendencies

- More and more the languages and formalisms are increasing the interest in this area
  - Implies the support to represent cellular automata or intelligent agents
  - CELL-DEVS
SDL language

- Object-oriented, formal language defined by The International Telecommunications Union as recommendation Z.100.
- Intended for the specification of complex, event-driven, real-time, and interactive applications involving many concurrent activities that communicate using discrete signals.
Reflexive agent specification

Diagram:
- Waiting
- Executing action
- Processing information

Transitions:
- Action executed
- Environment information
- Information processed
Reflexive agent specification

- Time to process information represent the delay due to the understanding of what happens in the world
Examples

The wildfire model and the slap avalanche model
Wildfire model.

- **Motivations:**
  - Dangerous environment.
  - Difficult to experiment.
  - Simulations involves natural resources and personnel.

- To develop an experimental framework to simulate a wildfire
  - Propagation.
  - Extinction.

- Working with:
  - CREAF data.
  - *Bombers de la Generalitat* (fireman).
Wildfire propagation (over R)

- Implemented using SDLPS.
- BEHAVE model.
- Raster data describing the landscape.
GIS Data

- Input data files:
  - **Mapa**: file containing the DEM (Digital Elevation Model).
  - **Model**: file that represents the propagation model implemented for each cell.
  - **Slope, Aspect**: files that store the slope and his direction. These files are calculated using the DEM. (Mapa files)
  - **M1, M10, M100, Mherb, Mwood**: files that contain the combustible description.

- The results files are two files:
  - **ignMap.dtm**: Stores ignition time.
  - **flMap.dtm**: Stores flames elevation.
GIS data: IDRISI32.

- 1987, Research program of Clark's University.
- We use the IDRISI32 file format.
  - One file for the data.
  - Other for the information related to the data.
The $A_1$ function works with Moore neighborhood therefore vicinity function and nucleus function are:

- $vn(x_i,x_j) = \{p_{i-1,j-1}, p_{i,j-1}, p_{i+1,j-1}, p_{i-1,j}, p_{i+1,j}, p_{i-1,j+1}, p_{i,j+1}, p_{i+1,j+1}\}$
- $nc(x_i,x_j) = x_i, x_j$
The events that lead propagation model are:

- **EBurn**: Associate to ignite fire into simulation cell.
- **EPropagation**: Programmed time for fire propagation to neighbor cell.
- **EExtinguish**: Programmed time to put out fire in a cell.
- **dataUpdate**: Event that represent a modification in the data used to calculate spread time. When this event is received is necessary to recalculate propagation model, (for instance a modification of the wind speed or direction).
Moore neighbourhood
BEHAVE model
BEHAVE model

- The BEHAVE library, Andrews 1996.

- Based in a cellular automaton and a discrete simulation model.

- From a set of raster layers and an initial point the model calculates the ignition time and the elevation of the flames on each cell.

- In our model:
  - The fire starts in a know cell.
  - The results are calculated to the neighborhood cells.
  - Analyze what is the cell with the lowest ignition time.
  - Recalculate the results for this cell.
  - This loop is repeated while exist cells in the model.
- **EBurn**: Associate to ignite fire into simulation cell.
- **EPropagation**: Programmed time for fire propagation to neighbor cell.
- **EExtinguish**: Programmed time to put out fire in a cell.
- **dataUpdate**: Event that represents a modification in the data used to calculate spread time. When this event is received, it is necessary to recalculate the propagation model, for instance a modification of the wind speed or direction.
The third level of the SDL formalism.
Simulation model
Visual effects: objects
Visual effects: example
Visual effects: example

- Wildfire in action
More cells implies better results, better representation but more CPU time.
Results

- Height and temperature of the flames.
flMap.dtm (with IAgents)
Examples

Using cellular automata to represent an slap avalanche
Avalanche

Two main types of snow avalanche:

- **Loose-snow** avalanche originates at a point and propagates downhill by successively dislodging increasing numbers of poorly cohering snow grains, typically gaining width as movement continues down slope.

- **Slab avalanche**, occurs when a distinct cohesive snow layer breaks away as a unit and slides because it is poorly anchored to the snow or ground below.
Avalanche fatalities in IKAR Countries

Avalanche Fatalities in IKAR Countries 1976-2001

Number of Fatalities

Winter

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25


125 153 146 153 138 149 174 224 204 192 146 140 141 138 145 176 209 176
Some photos
## Avalanche Model data

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
<th>Qtt</th>
<th>Source</th>
<th>Modifiable</th>
</tr>
</thead>
<tbody>
<tr>
<td>Height</td>
<td>Raster</td>
<td>Layer representing the height of the environment.</td>
<td>1</td>
<td>ICC</td>
<td>No</td>
</tr>
<tr>
<td>Thickness of the snow</td>
<td>Raster</td>
<td>Represents the thickness of the “slab snow”</td>
<td>1</td>
<td>Meteocat</td>
<td>Yes</td>
</tr>
<tr>
<td>Floor features</td>
<td>Raster</td>
<td>Represents the kind of surface (rocks, sand, snow, ice,..). Each surface has his own specific rough parameter.</td>
<td>1</td>
<td>Meteocat Creaf</td>
<td>No</td>
</tr>
<tr>
<td>Snow that causes the slab features</td>
<td>Raster</td>
<td>Density, compactness of the snow.</td>
<td>1</td>
<td>Meteocat</td>
<td>Yes*</td>
</tr>
<tr>
<td>Obstacles</td>
<td>Raster</td>
<td>Represents the obstacles that have the environment (small rocks, big rocks, houses, trees,..)</td>
<td>N</td>
<td>Creaf</td>
<td>Yes</td>
</tr>
<tr>
<td>Crack</td>
<td>Vectorial</td>
<td>Line representing the breakdown of the ice.</td>
<td>1</td>
<td>Input data</td>
<td>Yes, at beginning.</td>
</tr>
<tr>
<td>State of the snow</td>
<td>Raster</td>
<td>Shows the state of the terrain, empty, static and dynamic</td>
<td>1</td>
<td>Meteocat</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Avalanche Model

- $6+N:2-\text{AC}^{4+N}$ on $\mathbb{Z}^2$
Vicinity and nucleus function

- Vicinity function: \( \text{vn}(x_1, x_1) = \{(x_{1-1},x_{2-1}), (x_{1-1},x_2), (x_{1-1},x_{2+1}), (x_1,x_{2-1}), (x_1,x_2), (x_1,x_{2+1}), (x_{1+1},x_{2-1}), (x_{1+1},x_2), (x_{1+1},x_{2+1})\} \)

- Nucleus function: \( \text{nc}(x_1, x_1) = \{(x_1,x_1)\} \)
Evolution functions

- $E_2[i]$: Thickness of the snow. The function that rules this layer is “Modify information(p)”
- $E_4[i]$: Density, compactness of the snow, in our case is 0.5 (Mears 1976).
- $E_6[i]$: State of the snow. The function is defined in the next diagrams.
- $E_N[i]$: Obstacles. The function that defines the obstacles we use in the model.
Moore neighbourhood
Λ: state of the snow
Empty process

- Empty
  - Break
    - Empty
  - Displace
    - Empty
  - Send(snowQt)
    - Empty
  - Receive(snowQt)
    - Modify information(snowQt)
      - Dynamic
Static process

- Static
  - Break
    - d = Calculate propagation()
      - d?
        - yes
          - Break
        - no
          - Dynamic
  - Displace
  - Send(snowQt)
    - Modify information(snowQt)
    - Dynamic
  - Receive(SnowQt)
    - Static
    - Static
Evolution function

- The increment in the force is used in the next expression to determine if the snow continues its movement to other cell, or stops its movement, if the force is equal to zero.

\[ F_{i,t} = \max(IF_{i,t} + \Delta F_{i,t}, 0) \]
Evolution function

- \( IF_{i,t} \) = Impulse force, depends on the quantity and quality of the snow, and the slope.
- \( SFF_{i,t} \) = Sliding friction force between the avalanche and the underlying snow or ground.
- \( IFF_{i,t} \) = Internal dynamic shear resistance due to collisions and momentum exchange between particles and blocks of snow, (internal friction force).
- \( ASFF_{i,t} \) = Turbulent friction within the snow/air suspension, (air suspension friction force).
- \( AFF_{i,t} \) = Shear between the avalanche and the surrounding air, (air friction force).
- \( FFF_{i,t} \) = Fluid-dynamic drag at the front of the avalanche (front friction force).
- \( OFF_{i,t} \) = Obstacle friction force.

\[
\Delta F_{i,t} = IF_{i,t} - (SFF_{i,t} + IFF_{i,t} + ASFF_{i,t} + AFF_{i,t} + FFF_{i,t} + OFF_{i,t})
\]
Results
Results
Results (5)

Desencadenant:
Localització: 8 cel·les, de (108, 33) fins (115, 33)
Gruix de neu de placa: 50cm (per totes les cel·les fracturades)
Terreny subjacent: Neu dura
Obstacles: No

Característiques de l’allau:
Terreny subjacent del camí: Neu dura
Màxima distància recorreguda: 1101,14m
Desnivell superat: 520,40m
Massa transportada: 10625kg
Massa de neu en dipòsit: 9957,50kg
Massa de neu perduda pel camí: 667,5kg
Velocitat màxima: 67,23m/s
Results (5)
Results (5)
Results (1 vs 3)

Figura 7.9 – Velocitat Sim.1 VS Velocitat Sim.3
Some references


Some references

- Doran, Jim. E. 2000, HARD PROBLEMS IN THE USE OF AGENT-BASED MODELLING, Proceedings of the Fifth International Conference on Logic and Methodology, Cologne, October 3-6


Some references

- Batty, Michael, 2995, Cities and complexity. The MIT Press.
Thanks!

Pau Fonseca i Casas
pau@fib.upc.edu

Technical University of Catalonia
Barcelona School of Informatics
Computing laboratory
Barcelona
Jordi Girona 1-3
(+34)93401773