Scalable Agent Platforms with friendly interaction for modeling practical problems

Presented by Luis Cabellos
Co-authors: Jesús Marco, Hector Rodríguez
Instituto de Física de Cantabria (IFCA)
UC, University of Cantabria
CSIC, NATIONAL RESEARCH COUNCIL
SANTANDER, SPAIN
Scope

- Introduction: Agents based models
- The “Machanguitos” platform
- Addressing a practical problem
- Ongoing work
- Interest of ABM for Federated Clouds
Introduction

- **Elements of Computational Models for Concurrent Computing:**
  - Agents
  - Actors
  - Entities*

- **Agent based computing:** systems are composed of multiple structures (agents) interacting over an environment. Components:
  - Agents
    - Internal State
    - Update Function
  - Environment

- **Actor model:** the system is composed of a single structure: the Actor. An Actor can:
  - Send/Receive messages
  - Have a Internal State/Logic
  - Spawn other Actors
ABM vs MAS

- ABM = Agents-Based Models
- MAS = Multi-Agent Systems
- They have different goals:
  - ABM: search for explanatory insight into the collective behavior of agents obeying simple rules
  - MAS: computerized system composed of multiple interacting intelligent agents within an environment.
  - Multi-agent systems can be used to solve problems that are difficult or impossible for an individual agent or a monolithic system to solve
- There is a considerable overlap: as we will see, the proposed platform, Machanguitos, can be seen as

*The Easiest Simplest Multi-Agent System*
Introducing Machanguitos: Features

- Agent-Granularity: agents at various scale
  - Only 1 scale
- Decision-making heuristics
  - Scripting Language for definition of Agent behavior
- Learning rules or adaptive processes
  - Agents with internal state
- An interaction topology
  - Stand-alone Agents
- An (non-agent) environment
  - Raster 2D
Introducing Machanguitos: scripting

function Agent:checkHill(delta)
self.dx = ((-20 + math.random(40))/ 100.0)*delta);
self.dy = ((-20 + math.random(40))/ 100.0)*delta);
tempX = self.x + self.dx;
tempY = self.y + self.dy;
local area = raster.area:get( 0, tempX, tempY);
if area > 0 then
    self.x = tempX;
    self.y = tempY;
    raster.position:increment( 0, self.x, self.y, 200 );
end
end

function Agent:eatAndPoop(delta)
local grass = raster.grass:get( 0, self.x, self.y);
if (grass > 0) then
    raster.grass:increment( 0, self.x, self.y, self.grassEated*delta );
end
local inc = self.grassToManure * delta;
raster.manure:increment( 0, self.x, self.y, inc );
end
Machanguitos Run Model

Diagram:
- Driver Process
- Update Rasters
- Update Agents
- Client Process 1
- Client Process N
- get/set Raster
- Save Agent Data
- Data Process
- Export Raster
- Image File
- Mongo DB
Machanguitos Run States

Diagram:

- Server:
  - Initialize Simulation
  - Create Agents
  - Create/Load Rasters
  - End Simulation

- Agent:
  - Initialize Agent
  - Update Agent
  - Save Agent Data
  - End Agent

- Process:
  - Start Step
  - Update Agents
  - Update Rasters
  - End Step
Addressing a practical problem

Advanced management of eutrophication problem in a water reservoir (LIFE+ project)
How to “assign” uncertainties to key but complex processes?

- Practical Problem: impact of cattle management
  - Extensive or semi-intensive
  - >6000 cows
  - >10000 sheeps
- Parameterization applied based on:
  - P and N deposits/animal
  - Run-off (7% if 30mm rain...)
- But real life is much more complex
**ABM simulation of cows impact**

- 1200 iterations
- 10K - 1M cows
- 5K - 500K sheeps

<table>
<thead>
<tr>
<th>Cores</th>
<th>15K agents</th>
<th>150K agents</th>
<th>1.5M agents</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>179</td>
<td>1086</td>
<td>8787</td>
</tr>
<tr>
<td>8</td>
<td>142</td>
<td>651</td>
<td>5645</td>
</tr>
<tr>
<td>16</td>
<td>141</td>
<td>722</td>
<td>6597</td>
</tr>
<tr>
<td>32</td>
<td>142</td>
<td>799</td>
<td>8149</td>
</tr>
<tr>
<td>64</td>
<td>167</td>
<td>1072</td>
<td>8289</td>
</tr>
<tr>
<td>128</td>
<td>206</td>
<td>1196</td>
<td>10759</td>
</tr>
</tbody>
</table>
On going work

Evolution of the Platform
- Better concurrent access to the data
- Add Actors properties
- Better definition of environment
- Other environments (Dynamic GIS)

Future of the Model
- Realistic scripts for cows and ships
- Realistic mineralization processes?
- Validation
Interest of ABM for Federated Clouds

Implementation as SaaS?

- Service orientation
- Collect scenarios, scripts
- Well suited to distributed execution
- Multilayer approach

Many areas of application

- Socio-economic systems
- Smart cities
Questions?