Cellular Automata

Jerilyn Zheng
Deerfield Academy, 7 Boyden Lane

Introduction

This assignment required groups to create a cellular automata model of an ecosystem with different organisms and interactions. The challenge was to design the organisms to be sustainable and interactive for an infinite amount of time so that none of the species would fully die out. The functionality of the ecosystem resulted from the specific class structure we used.

Some of the organisms would be static and others were able to move around. These systems mainly revolved around realistic human or animal interaction. The world were created to be dynamically sized. Each organism had a specific manner of moving and rules were created to govern the activity and ensure that all the organisms stayed sustainable and actively interacting throughout the lifespan of the world.

Background

Cellular automata models are a particular type of computer models that model social interactions between agents. Typically, these models are in a checkerboard structure, with each cell representing a location. Each location is a position where an agent/organism can move to. These grids are called cellular worlds, and they represent a very simplified model of how people live in a location. The cellular world showcases the local interactions of the individuals within the social system.

In cellular automata models, organisms can only interact with other organisms in close proximity to them. This is called a neighborhood. Each organism has a finite number of states (like an on and off state), and based on the rules of the organisms and the current state and the states of the other cells in the neighborhood, a new generation is created. The cells are updated and the states are changed.

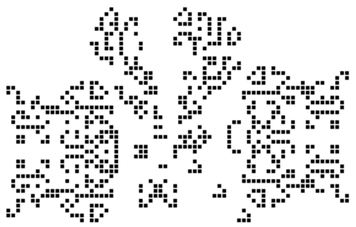

Figure 1. An example of Conway's Game of Life and the interactions.

Conway's Game of Life is an example of a famous 2-D cellular automaton. There were many different types of patterns that occurred through the extremely simple rules that were modeled based on the interactions of each cell with it eight neighbors. The births and deaths of the organisms were modeled to contain growth and keep the world sustainable.

Design

There were separate classes for each organism and one super class "Agents", in which they inherited the move method. If the rules enabled an organism to move, the move method picked a random direction from eight possibilities and the organism moved one space in that direction. The world was designed to be circular, so that the organisms would wrap around the the neighborhoods would still stay intact even on the edge of the board. Each organism class, Walls, Enemies, and Spies had their own set of rules governing the interactions and the spawns and deaths of the organisms. The actual locations of the agents were recorded in a 2-D array, with only the set states of the organisms representing their locations. There was also a list of all the organisms in an array list to count the number in each neighborhood for the rules. With the Grid class, the graphical representations were implemented and the values from the array were converted into the grid. The delay ensured that the grid could constantly update with the movements of the organisms. The user input used the Java IO Dialogue boxes to prompt the users about the screen size and dimensions of the world.

The organisms had specific rules to maintain sustainability. Walls were static and could only be spawned 50% of the time if the number of spies was equal to the number of enemies in a neighborhood. If there were less enemies than spies, the enemies would die, and vice versa. Booleans were implemented in to make the activities slightly more random and to occur less frequently, thus producing a constant state of growing and dying organisms.

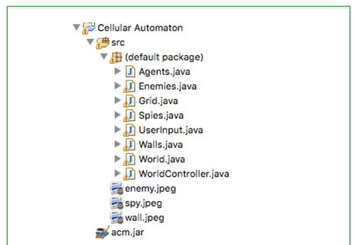

Figure 3. An example of the class structure of the cellular automata project.

Discussion

The main problems we encountered were centered around sustainability and coming up with rules. Before implementing the boolean in which some of the rules would only occur 50% of the time, often times either the spies or enemies would end up dying out after a certain number of iterations.

Results

The world stayed sustainable after n number of iterations and stayed efficient even with a large world. Because of the very short delay, sometimes it would appear that organisms were not dying off but instead just moving around, but the rules worked accordingly and exactly as the criteria specified. In addition, the relative number fo each type of organisms was pretty much even throughout the iterations.

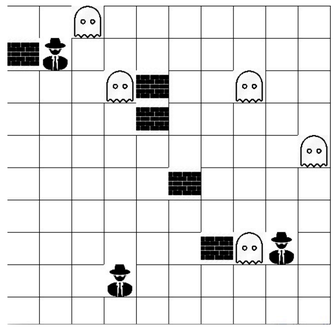

Figure 2. An example of the spies, walls, and enemies in action within the world.

Conclusion/ Future Directions

We were able to solve the problem. The system was sustainable and all three types of organisms stayed alive after any number of iterations. In the future, I would like to implement the different states of the organisms as a more active portion of the world. We had two states for each organism, but they ended up not mattering because the organisms would just disappear from the world if they died. There would sometimes be a barrier of walls because the walls were static, so maybe in the future I would adjust the rules to find a more spaced out version of the organisms.

Sources

http://pentadecathlon.com/lifenews/
https://www.youtube.com/watch?v=EyrwOf239M4
http://natureofcode.com/book/chapter-7-cellular-automata/