# A Cellular Automata Approach to Population Modeling

Alexa Silverman, TJHSST Computer Systems Lab, 2008-2009

## Abstract

Cellular automata (CA) are a very basic form of artificial intelligence, wherein each individual cell on a grid only knows its own state, its neighbors' states, and a rule instructing when to change states.This project explores the capabilities of the NetLogo environment and the Java programming language in running CA-based simulations. In two agent-based systems, CA are used to model the effects of temperature and virus spread on a population.

## Background

Cellular automata exist as 'cells' on a grid, wherein the behavior of each cell is determined by the states of its eight neighboring cells. In this way, a cell acts as an agent in an agent-based system. The "Life" collection of automata rules is based on the idea that survival of individuals in a population requires an adequate number of neighbors; that an individual can die due either to loneliness or overcrowding, and that an individual is only born when a "family" of individuals is already present.

This program utilizes both NetLogo and Java in running simulations. NetLogo is useful for its capabilities of running agent-based simulations and displaying their results in its graphics interface. Java's object-oriented nature allows for the creation of agents with more individual properties as well as allowing for easy export of data to a file.

## Development

The project has two components, one based in NetLogo and the other in Java. The NetLogo program runs the 14/3 Life rule, in which a live cell with one or four neighbors survives the generation, and a dead cell with exactly three neighbors will be born in the next generation. This rule was chosen because the growth of cells mimics the growth and migration of populations over long periods of time. The cell population is assigned a birth rate, which determines the likelihood of a dead cell to be born. This birth rate varies quadratically with the system's temperature, having a maximum when the temperature is 21 degrees Celsius. The temperature itself varies directly with changes in population.

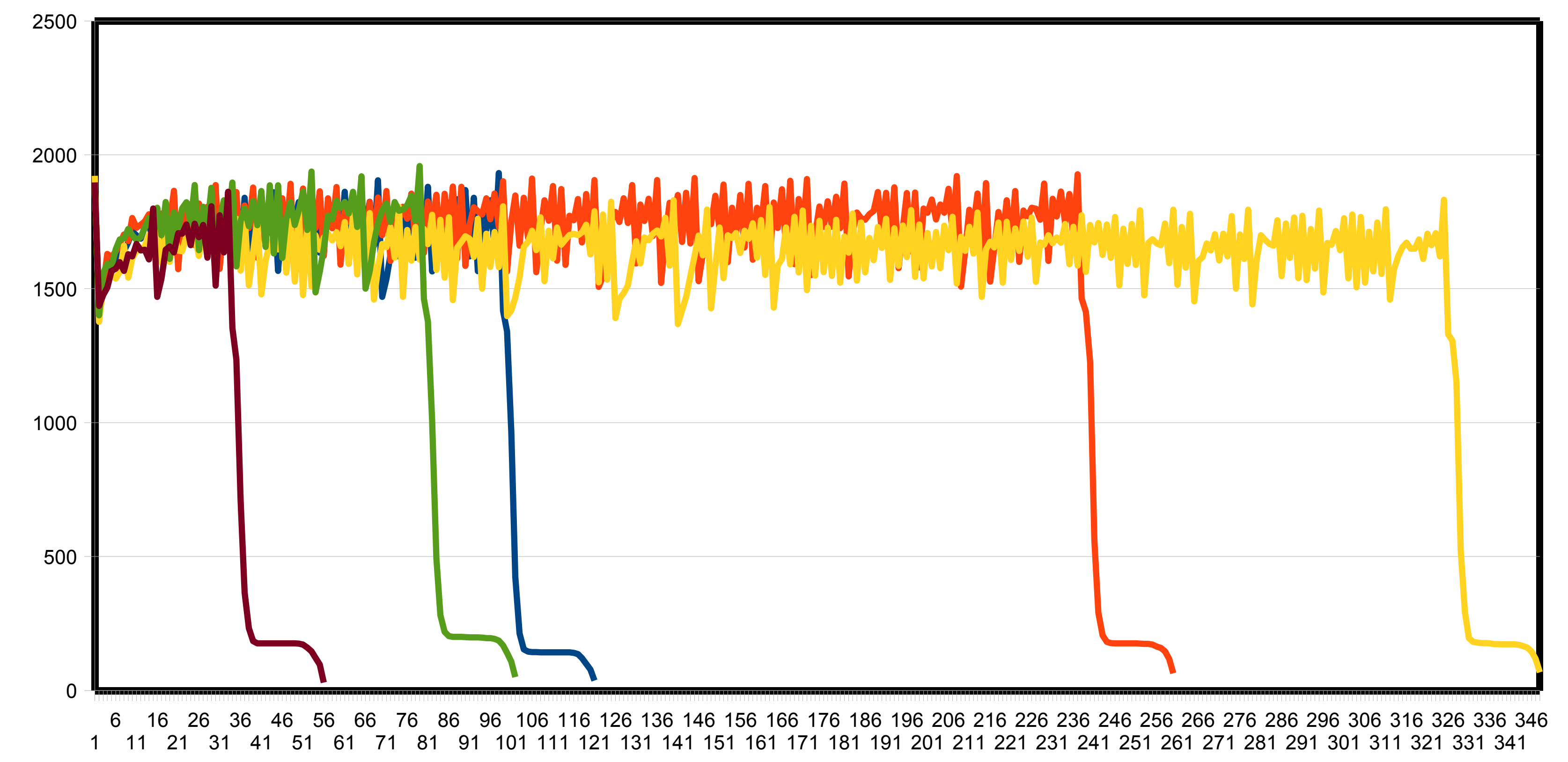
The user may change the initial parameters of temperature and population density, after which the simulation runs in the graphic interface and results can be observed both visually and as data in graphs of popultion and temperature over time.

The Java program runs the 23/3 Life rule, also known as Conway's Game of Life. This rule was chosen because it mimics the behavior of a population over a shorter time period. Cells are randomly assigned, in addition to the 'on' or 'off' state, a state of 'infected' or 'healthy' at the beginning of the simulation. Infected cells have a 50% chance of infecting healthy neighbors. The Java program displays agent interactions visually using the Graphics package and also exports data about virus spread to a file.

## Results

### *NetLogo component*

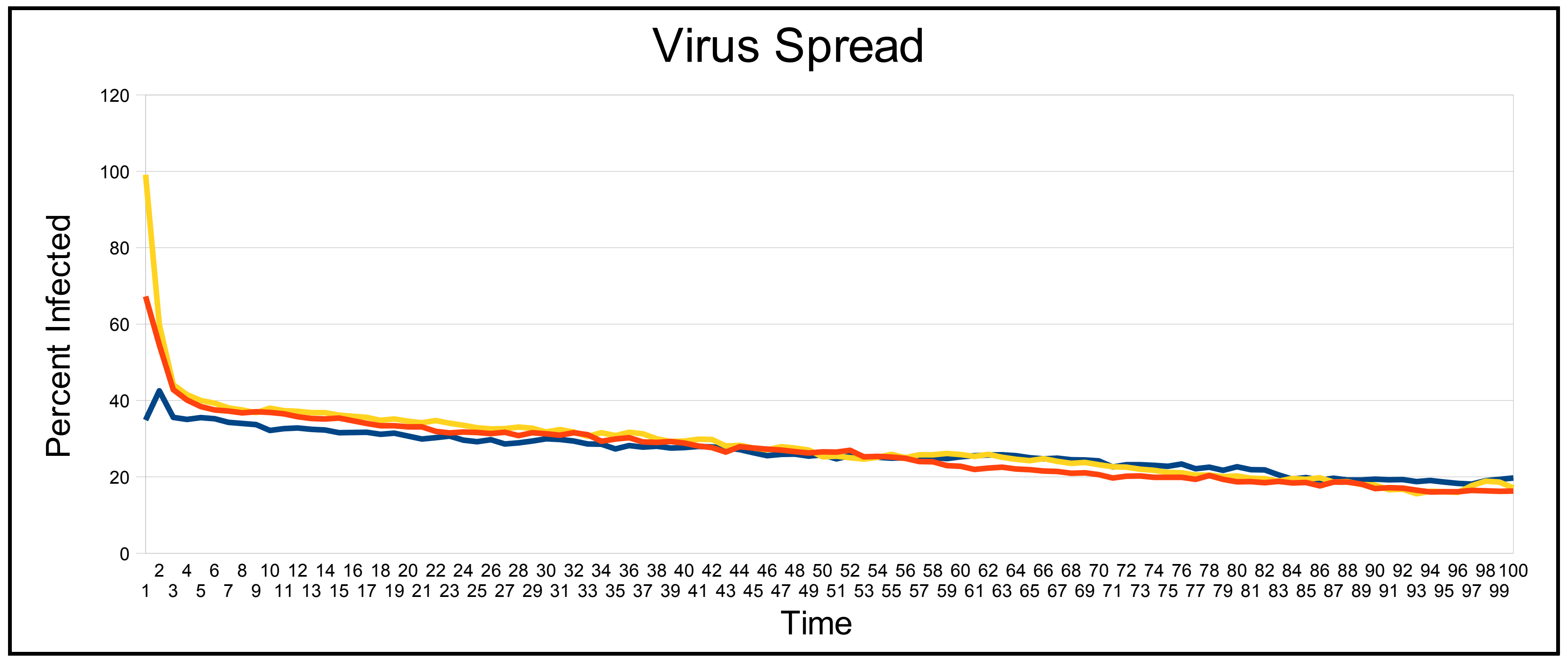
With default settings (temperature of 21 degrees Celsius, population density of 55 percent), the population eventually dies off. This may provide a (highly simplified) explanation for the phenomenon of global warming. This result is not always produced if the initial parameters are changed. The amount of time after which the population takes a sharp decline varies from run to run due to the nature of cellular automata to produce heterogeneous results. However, the trend remains the same.



*Graph of population changes in five test runs.*

### *Java component*

After 100 test runs, in which the initial percentage of infected cells was increased by 1% for each run, it appears that a general trend exists in which the percentage of infected cells decreases until it stabilizes at about 20% infected. This seems to suggest that the simulation has a carrying capacity for infected cells, but it is clear that if even a small percentage of cells are infected, the virus will continue to infect cells.



*Graph showing virus spread in three test runs with different initial infected percentages.*