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. CA have recently begun to be used in the field of agent-based modeling, which provides possible explanations for real-world phenomena using emergent behavior based on the interactions between individual agents. This project explores the usefulness of CA in modeling the effects of temperature on a population, as well as the capabilities of the NetLogo environment and Java's Graphics package in running CA-based simulations.

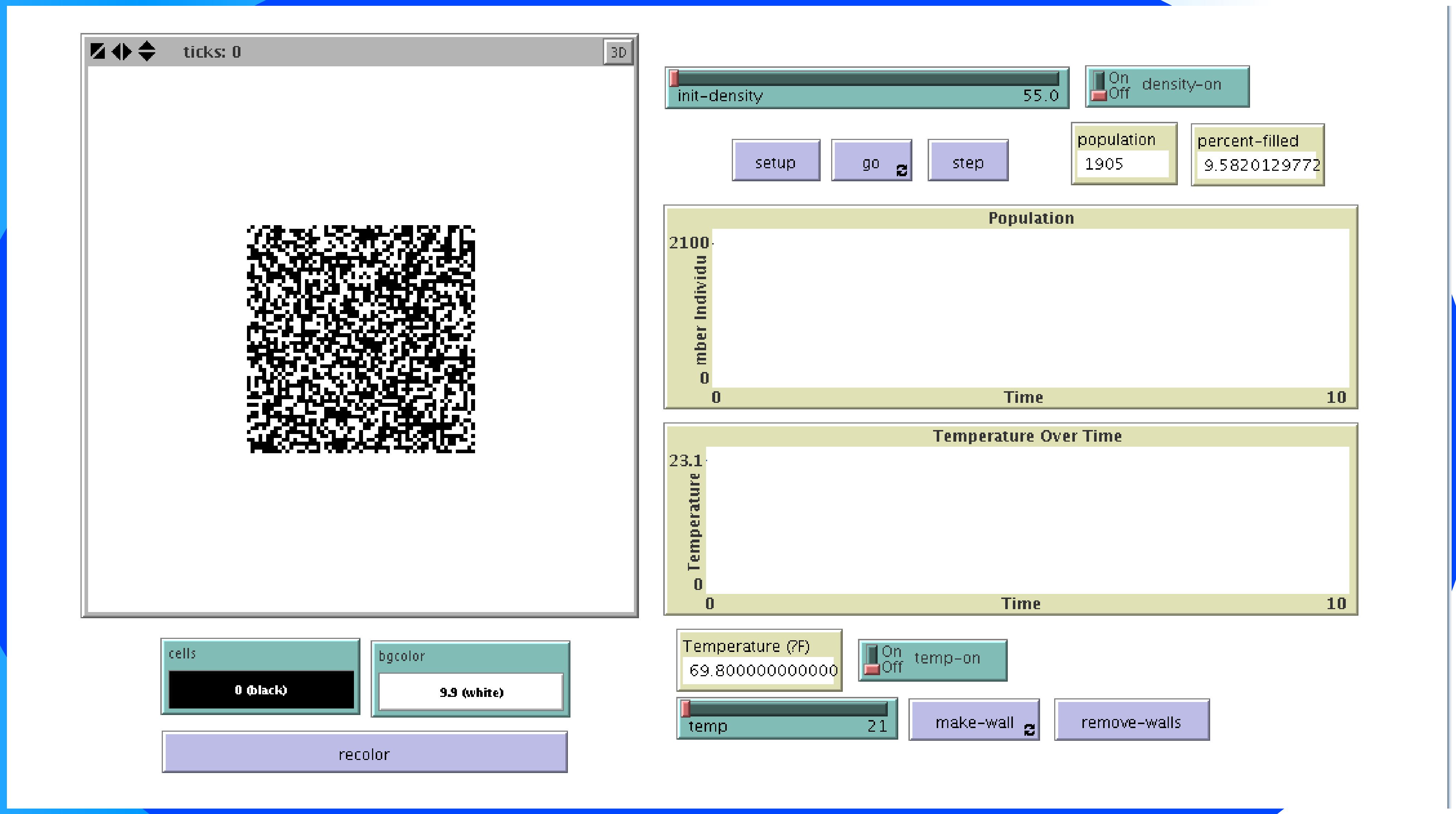
Background

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.

The rule used in this project to predict behavior is 14/3; in the terminology of Life rules, this means that a live cell with one or four live neighbors will survive a generation, and a dead cell with exactly three live neighbors will be replaced with a new live cell. This inherently suggests that two types of individuals exist: one which is less social (prefers one neighbor) and one which is more social (prefers four neighbors). This program models the effects of temperature and population and vice versa; population varies quadratically with temperature and temperature varies linearly with changes in population.

Development

The NetLogo program correctly runs the cellular automata rule 14/3 and allows the user to select the initial population density. It also graphs the percentage of live versus dead cells and provides population counts.
In test runs, the percentage and population of live cells are monitored. Variations occur due to the introduced variable of initial population density and the individual behavior of cells, which changes based on the random placement of cells on the grid. The nature of cellular modeling produces slightly different results with each run. After an initial temperature is selected, temperature varies linearly with the changes in population (that is, the difference between the current population and the population at the end of the previous generation).
During a test run, monitors display the population count (number of live cells), temperature in degrees Fahrenheit, and percentage filled of the grid (ratio of live to dead cells in the grid). Two graphs display the changes in temperature (in Celsius) and population over time. The user may also use the add and remove walls buttons to draw walls around the cells, separating a section of the population from the larger group. These walls cannot be crossed by cells and are considered 'dead' by the cells but are unable to become alive at the next generation.
In the Java program, cell interaction is accomplished via the cells being considered as belonging to a Grid class. The Grid contains a matrix in which each data point contains an individual cell, which knows its number of live neighbors and its own state. A JPanel contains the Grid and Cells and a Timer class that dictates the actions of the cells at each turn.



init-density 55.0

density-on: On / Off

population 1905

percent-filled 9.5820129772

cells 0 (black)

bgcolor 9.9 (white)

Temperature (?F) 69.800000000000

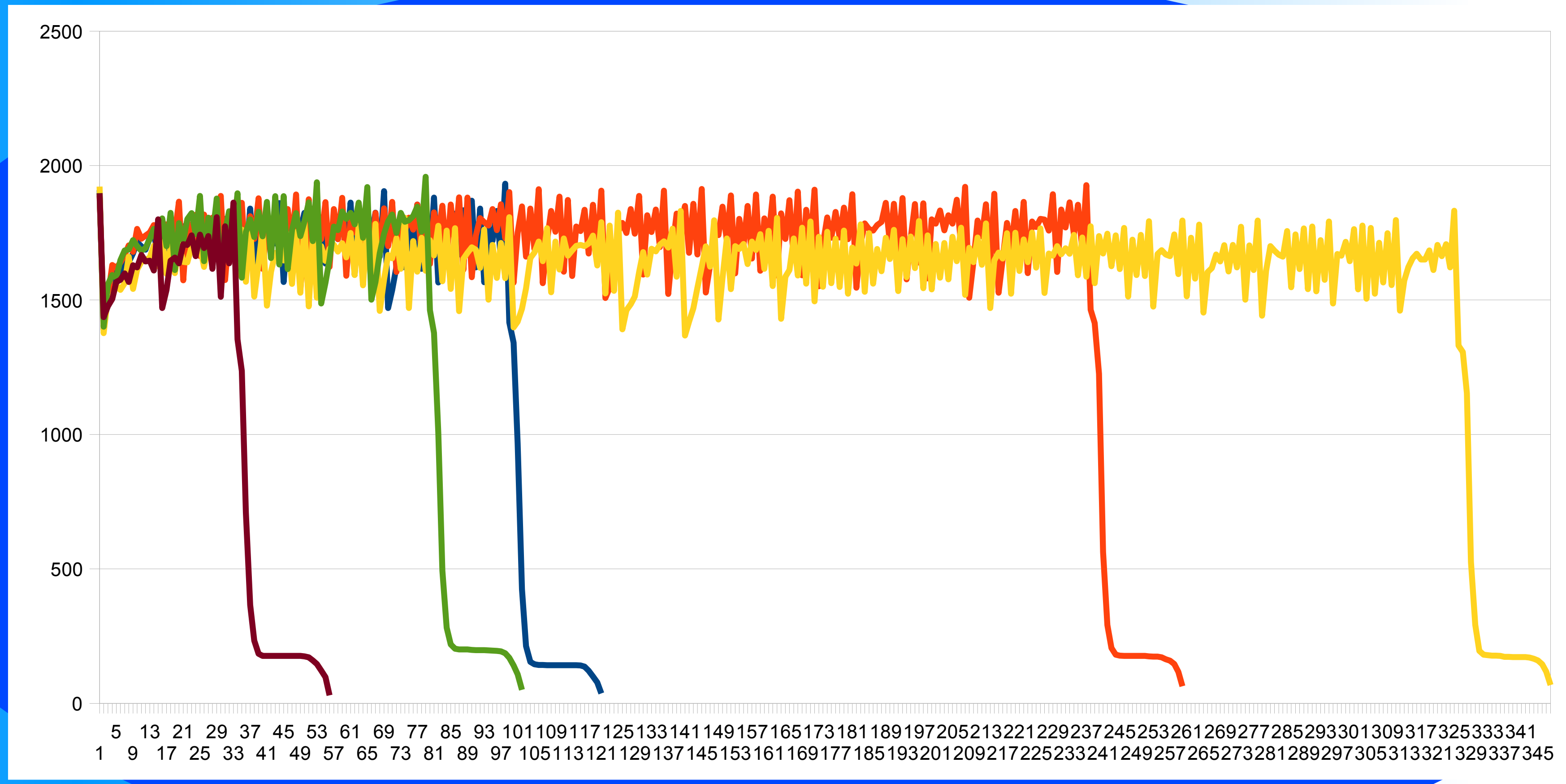
temp-on: On / Off

temp 21

The program in the NetLogo interface.

Results

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.



Test results showing several test runs.