Simulating Forest Fires with Finite Cellular Automata

Eric Furugori
38701116

Physics 210
Department of Physics
University of British Columbia
Instructor: M. Choptuik

December 5, 2012

Contents

Abstract

A finite cellular automaton model of forest fires is designed in python and its behaviour is analyzed. Two seperate models are created to analyze the effects that the initial distribution of trees has on the spread of a forest fire, and the evolution of the system due to a probability that a tree will ignite spontaneously and a probability that a new tree will grow. The former model is shown to produce a critical density over which fires spread very easily. The latter model is shown to determine that there is a correlation between the ratio of the probabilities of spontaneous combustion to spontaneous growth and the evolution of the system. Upon further analysis of the lightning-strike model, we find a set of initial probabilities for which the system is close to being critically self-organizing.

1 Overview

1.1 Introduction

A finite cellular automaton model of firest fires was designed and developed in python for the testing described in this report. In this report two models will be described. The density model takes an initial density probability P_D and, begins with all trees on the leftmost side of the forest burned. The system then evolves until the fire stops.

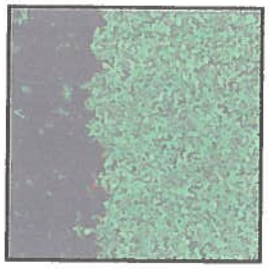


Figure 1: Visualization of Density Model with the Pygame Library

The lightning-strike model takes the probability of a tree spontaneously burning P_S and the probability that an empty cell will grow a new tree P_G. The initial probability density P_D is arbitrary in this model as over time, the only factors in the system's behaviour are P_S and P_G. Moreover, it is shown that there is a correlation with the behaviour of the ratio of P_S to P_G.

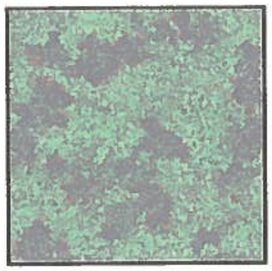


Figure 2: Visualization of Lightning-Strike Model with the Pygame Library

1.2 Mathematical Formulation

We start with an $M \times M$ dimensional lattice where each cell has both a state and an address. A cell state $S_{(i,j)} \in \{Empty, Tree, Burning\}$ is one of *Empty*, *Tree*, or *Burning*. A cell address is a set of integer coordinates (i, j) such that $i = 1, 2, ..., M$ and $j = 1, 2, ..., M$. We then define the state of each cell at iteration n to be $S^n_{(i,j)} \in \{Empty, Tree, Burning\}$. Each cell has a neighbourhood $N^n_{(i,j)} \subseteq \{S^n_{(i-1,j)}, S^n_{(i,j-1)}, S^n_{(i+1,j)}, S^n_{(i,j+1)}\}$ consisting of the cells directly above, below to the right, or to the left of it. A cell can only be added to a neighbourhood if it is burning, and it must exist within the lattice.

We then define a transition function $f : S^n_{(i,j)} \times S^n_{N(i,j)} \to S^{n+1}_{(i,j)}$ that will determine the next state of the cell at address (i, j). This transition function is dependent on pseudorandom numbers generated by python's random module and the respective probabilities that each model depends on. The details of this transition function is provided in the numerical approach of each model.

1.3 Techniques and Algorithms

The finite cellular automaton makes use of python's built-in list data structure. We make use of multidimensional lists to create a model of a square grid. The grid is defined as a python class, and one instance of a grid is created in the program. A cell class is then defined that contains cell-specific information such as state and address. Cells are instantiated when a grid is created, and there are exactly $CELL_X \times CELL_Y$ cells to fill up the grid. We then make use of a main class that will take all the parameters we change at the top of the program and run the loop.

While the original plan was to use numpy arrays to create the grid and unsigned integers to represent states, I was unable to provide a working implementation using these data structures. Instead I used the much slower approach of iterating over each cell in the grid. The main class has an update method that iterates over all the cells in the grid and performs a cell method on each cell depending on its state. It then updates the cell states. There is also a write state method that records the densities at each iteration. All these operations are run in the main method of the main class, which stops when either we have gone through our specified number of iterations nsteps, or when we want to terminate it prematurely, we set the class variable in main terminate to True. A the end of the program a plot is made and saved to the same directory that forestfire.py is in.

1.4 Discussion of Computations

The computations done differed depending on the model being analyzed. For the density model, we modified the control parameter P_D, the initial density of the distribution of trees. Multiple simulations were run using the same value of P_D and the results were averaged. This was necessary since these numbers are inherently random, so there is no "correct" value.

For the lightning-strike model, we focused on analyzing the trends of the system over long periods of time after setting the control parameters P_S and P_G. The intent was to find what set of parameters threw the system into non-constant behaviour.

2 The Density Model

2.1 Numerical Approach

Consider the transition function previously defined by $f : S^n_{(i,j)} \times S^n_{N(i,j)} \rightarrow S^{n+1}_{(i,j)}$. In the density model, f is defined by the following cases (where $rand \in \Re, 0 \leq rand \leq 1$, rand is uniformly distributed):

- If $S^n_{(i,j)} = Empty$ then $S^{n+1}_{(i,j)} = Empty$
- If $S^n_{(i,j)} = Tree$ then for each $S^n_{N(i,j)} = Burning$:
  - If $rand < P_N$ then $S^{n+1}_{(1,j)} = Burning$, else no change
- If $S^n_{(i,j)} = Burning$ then $S^{n+1}_{(i,j)} = Empty$

Note that $P_N = 1$ always in this particular analysis of the finite cellular automaton.

2.2 Analysis

The overall time that the fire spread was measured for P_D with increments of 0.1. Each result was averaged over 10 simulations and rounded to the nearest integer as a precautionary measure towards anomalous data. The simulation was run with a lattice dimension of 150×150.

P_D	0.1	0.2	0.3	0.4	0.5	0.6
n iterations	4	8	11	17	34	357

Table 1: Averaged n iterations took for fire to stop spreading at different P_D

We notice that the fire starts to spread somewhere between $P_D = 0.5$ and $P_D = 0.6$. The same procedure is then used for increments in P_D of 0.01 between these two values.

P_D	0.51	0.52	0.53	0.54	0.55	0.56	0.57	0.58	0.59
n iterations	43	50	53	52	71	112	137	148	296

Table 2: Averaged n iterations took for fire to stop spreading at different P_D between 0.5 and 0.6

It is evident that for densities of $P_D > 0.58$ that the fire begins to spread for much longer. So we have found that there does indeed exists a critical density at which the fire will spread over a large area of forest. This can be seen in the following plots of 2 discrete simulations at $P_D = 0.58$ and $P_D = 0.59$.

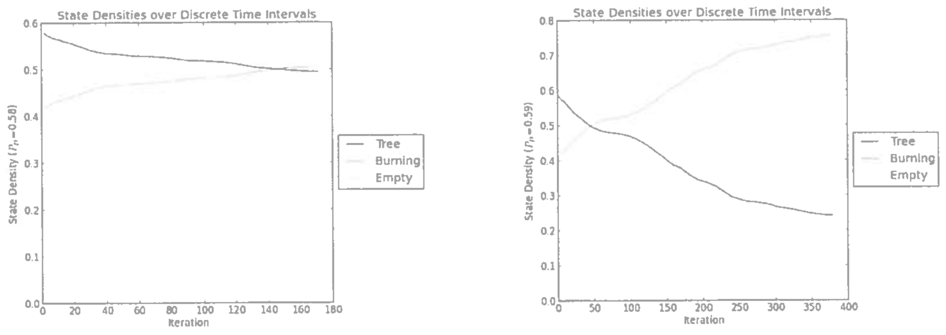


Figure 3: Comparison in the fire spread at $P_D = 0.58$ and $P_D = 0.59$

We see that for $P_D = 0.58$ we have that the change in P_D, $\Delta P_D \approx 0.1$ while for $P_D = 0.59$, $\Delta P_D \approx 3.5$ over roughly twice as many iterations. We then conclude that for a forest emulating this nature, if its tree density P_D is greater than the critical density 0.58 then if a fire were to start, it will spread much more than for $P_D \leq 0.58$ (if we consider only increments in P_D of 0.01). It would then be beneficial to consider that logging companies not let the densities of the forests they are harvesting increase to over 0.58 as a precautionary measure so that fires will not spread so easily, thus safely maximizing the amount of trees per unit area.

3 The Lightning-Strike Model

3.1 Numerical Approach

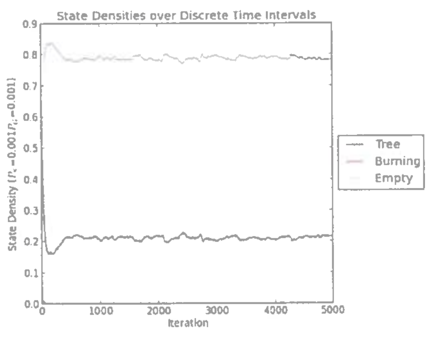
Consider the transition function previously defined by $f : S^n_{(i,j)} \times S^n_{N(i,j)} \rightarrow S^{n+1}_{(i,j)}$. In the lightning-strike model, f is defined by the following cases (where $rand \in \Re, 0 \leq rand \leq 1$, rand is uniformly distributed):

- If $S^n_{(i,j)} = Empty$ then:
    - If $rand < P_G$ then $S^{n+1}_{(i,j)} = Tree$, else no change
- If $S^n_{(i,j)} = Tree$ then for each $S^n_{N(i,j)} = Burning$:
    - If $rand < P_S$ then $S^{n+1}_{(i,j)} = Burning$, else:
        - If $rand < P_N$ then $S^{n+1}_{(i,j)} = Burning$, else no change
- If $S^n_{(i,j)} = Burning$ then $S^{n+1}_{(i,j)} = Empty$

Note that $P_N = 1$ always in this particular analysis of the finite cellular automaton.

3.2 Analysis

First the tendencies of the systems were measured over long period of time (over 5000 iterations) so that we can be sure that system is indeed converging into a certain trend. First the case where $P_S = P_G$ was examined with the following values:



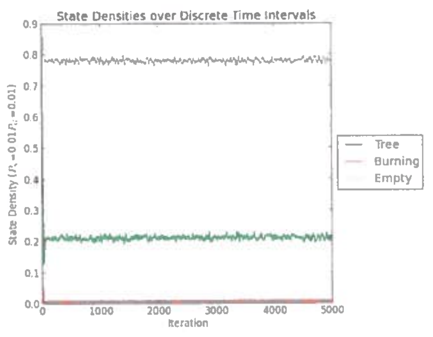
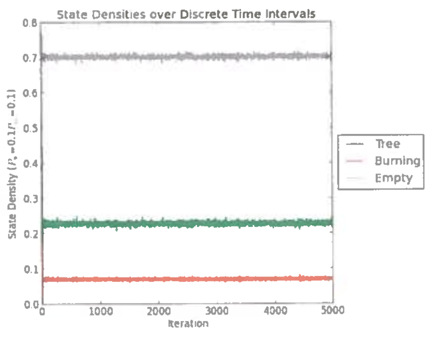


Figure 4: System tendency at $P_S = P_G = 0.001, 0.01$



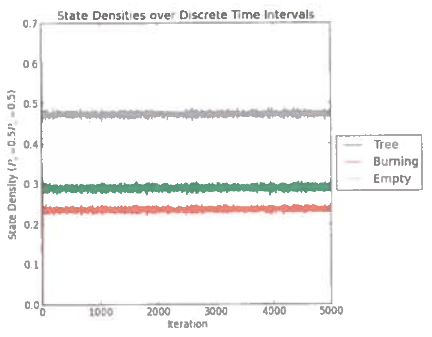


Figure 5: System tendency at $P_S = P_G = 0.1, 0.5$

We can conclude that we know the system converges to a preferred state when $P_S = P_G$. We have also found that if the amount of trees burning is negligable, then the density of cells in the *Tree* state becomes roughly 0.2 whilst the density of cells in the *Empty* state becomes 0.8. As P_S and P_G are increased, then the amount of trees burning at a single time is increased, and the amount of noise in the converging density values increases. So rather than converging to a single discrete value, we get that densities of states have a range of values at which they converge to. This behaviour is demonstrated in the above figures. It is notable, however, that for the forest to be burning at a significant percentage all the time is unrealistic, and thus such results are hypothetical in nature. We then explore ratios of P_S to P_G at different orders of magnitude over long periods of time (5000 iterations). The following plots were produced:

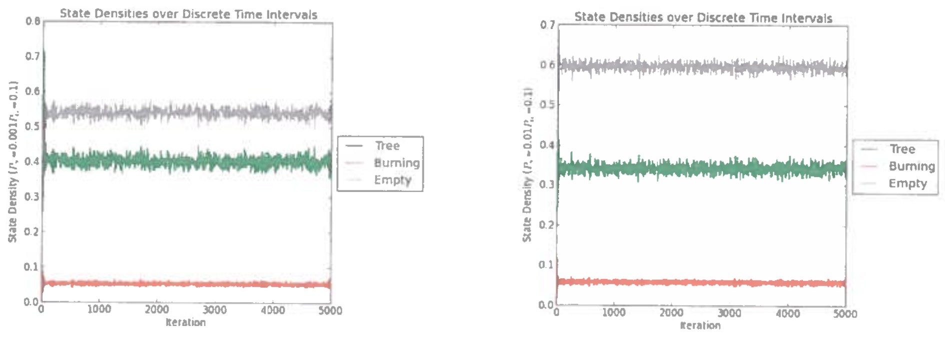


Figure 6: System tendency at $P_S 0.001, P_G = 0.1$ and $P_S = 0.01, P_G = 0.1$

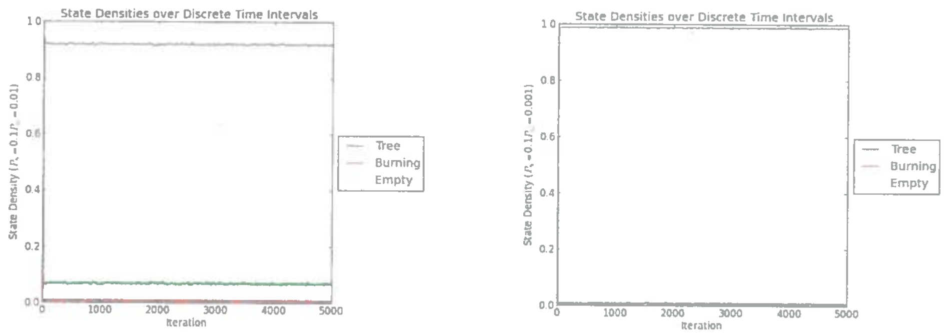


Figure 7: System tendency at $P_S = 0.1, P_G = 0.01$ and $P_S = 0.1, P_G = 0.001$

It is observed that for $P_S > P_G$ the system is less dynamic than for $P_G > P_S$ due to a noticable increase in noise in the converging density values.

decrease?

Since we are looking for a self-organizing critical state, we will focus on the case where $P_G > P_S$. In particular, we are looking for the values which do not seem to converge to a single point, such as the ones above. We begin by decreasing the scale of P_S and P_G by 2 orders of magnitude.

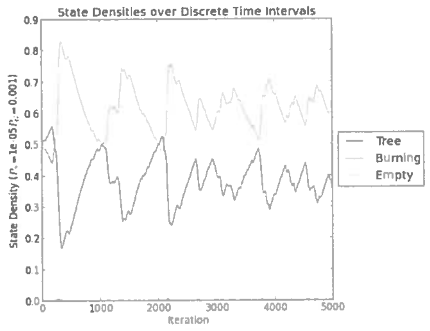


Figure 8: System tendency at $P_S = 0.00001, P_G = 0.001$

Now despite the fact that fluctuations are decreasing over time, we are much closer to a self of self-organization. The densities fluctuate in a very unpredictable manner. We will now look to increase and decrease the ratio of P_G/P_S to find a critical attractor. The simulation was run with $P_S = 0.00001, P_G = 0.005$ and $P_S = 0.00001, P_G = 0.0005$.

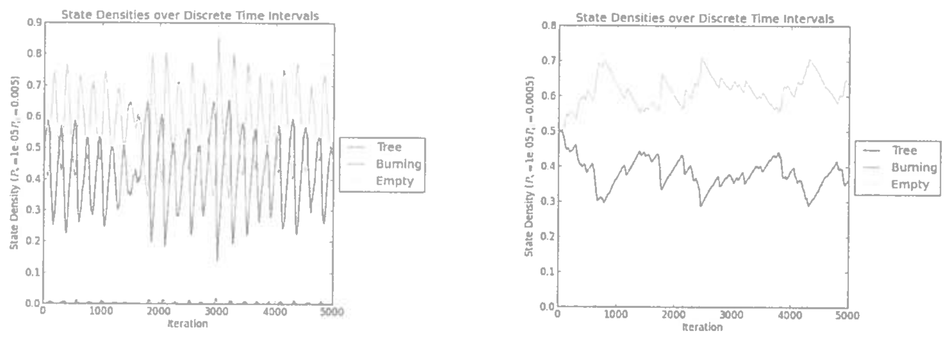


Figure 9: Tendency of simulation at $P_S = 0.00001, P_G = 0.005$ and $P_S = 0.00001, P_G = 0.0005$

While the densities at $P_S = 0.00001$ and $P_G = 0.0005$ do not seem to be converging to something measureable, the fluctuations are much more tame than those of the densities at $P_S = 0.00001$ and $P_G = 0.005$. Notice that on the left figure in Figure 9 that at iteration $n \approx 1600$ the fluctuations wane, then wax again. This is a strong indicator that the preferred state of the system is an evolving one. We can then run the same simulation over a significant amount of iterations to see this trend developed further.

conclude that a ratio P_S/P_G in the order of magnitude of 10^2 (in particular $P_S/P_G = 500$) will result in non-constant densities as the forest transitions from a desert (a forest with few trees) to an overpopulated forest, and then back to a desert repeatedly, never stopping at a stable middle ground.

4 Summary

4.1 Findings

In the density model it was determined that there is a critical density of initial tree distribution $P_D = 0.58$. if $P_D > 0.58$ then fires are far more likely to be able to spread across a forest.

In the lightning-strike model it was found that there is a self-organizing critical state at P_G/P_S in the order of magnitude of 10^2. We found specifically that a ratio of $P_G/P_S = 500$ will result in a contantly evolving forest.

4.2 Limitations

It is not difficult to see why these models are not accurate representations of forests. If this experiment were to be extended to more accurately represent reality, then we would have to modify the transition function to include weather and geological conditions, among other extranneous circumstances. This is one of the reasons why this simulation is not feasible, since it is not the case that if one tree is on fire then a nearby tree must ignite.

Another limitation to the analyzed models is that they assume square

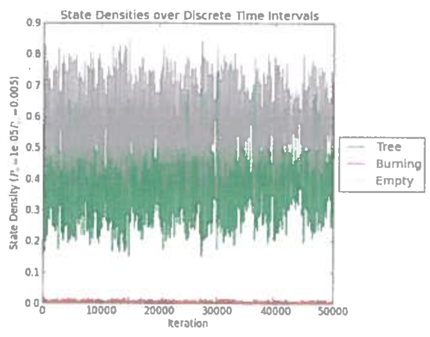


Figure 10: System tendency at $P_S = 0.00001, P_G = 0.005$ over 50000 iterations

We see that there is no indication that densities will converge. Then if the attractor of the system being a critical state is a property of the ratio P_G/P_S then it should be the case that if we divide both P_G and P_S by 10, and by 100, that we still see the same type of attractor.

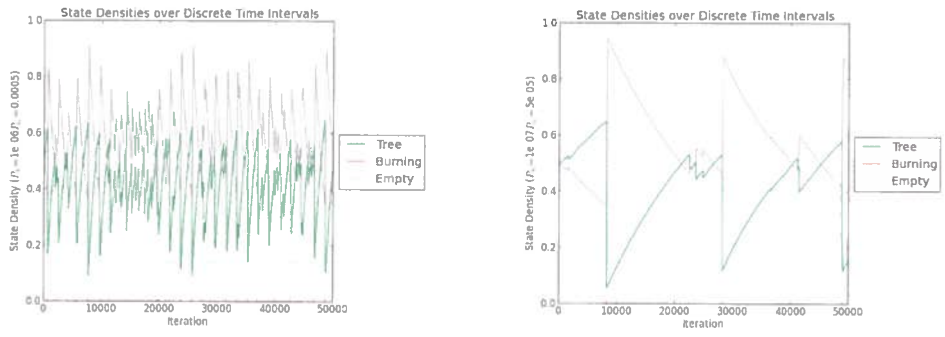


Figure 11: System tendency at $P_S = 0.000001, P_G = 0.0005$ and $P_S = 0.0000001, P_G = 0.00005$ over 50000 iterations

So while this may not be a rigorous proof, this is evidence to suggest that it is indeed the ratio of P_S/P_G that determines the attractor state. We then