Inderpal Singh

Prof Thanos

CSCI 191t

Mayn 5 2022

Motivation

Our motivation for doing this project was to have a better intuition of Stephen Wolfram's work on various cellular automata like totalistic automata, mobile automata, and most importantly elementary cellular automata. As well as to take the multitude of concepts discussed by Wolfram in his book *A New Kind Of Science* and apply them to some real world problems like simulating how fire could spread in a forest.

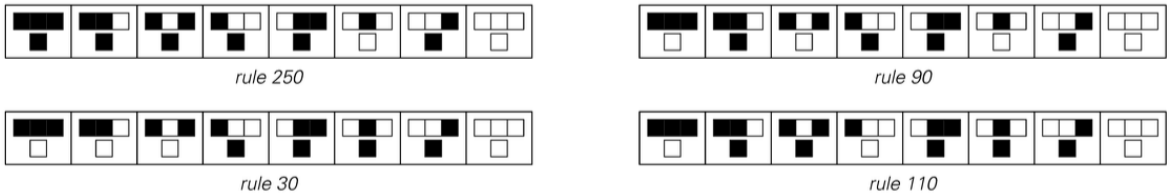
Problem Statement

One of our two objectives is to implement what Wolfram Described as an Elementary Cellular Automata. Which he classified as one of the simplest classes of one dimensional cellular automata that consist of only two possible values, and operate on using basic rules that take into account only local information that was available in previous time steps, resulting in them being completely deterministic. In order to observe some of the behaviors different rules exhibit, whether they be simple or very complex random structures.

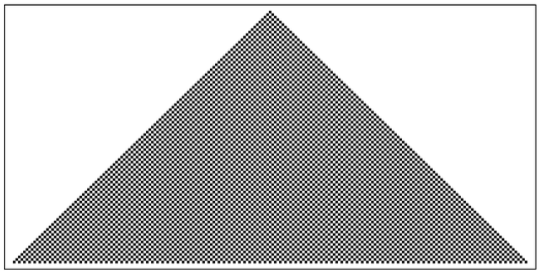
Our Second objective is to exploit the tendency for automata with simple rules to exhibit complex behavior to simulate how fire could spread in a simulated forest. By taking the principles of one dimensional cellular automata to make a more complex two dimensional, nondeterministic automata.

Additional Resources

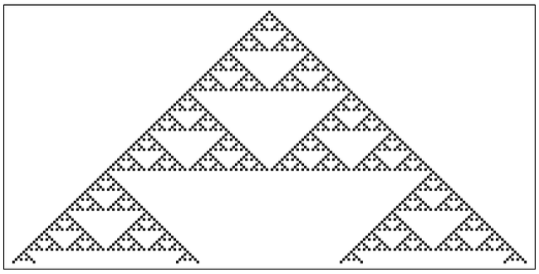
Most of our work is based on work in Stephen Wolfram: New Kind of Science, in which he described his work on different types of cellular automata, and expanding work done by other computer scientists. The simplest of the automata was the elementary automata. Where each row of an automata is determined by the state of the automata that came before. Like so:



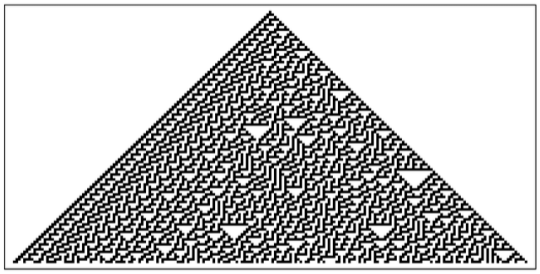
Here we can see that the state of a cell in any given row is determined directly by the three previous cells that came before. While simple in concept, in practice it can produce some interesting results. Wolfram described four possible categories that an automata can end up falling into. Repetitive, in which rows alternate or remain steady in a simple uniform pattern forever. Nesting where details seem to appear at a constant rate but with little to no variance in the same of the patterns. Random, where the pattern that is generated shows no clear overarching structure. Finally, there are localized structures, where the details that seem to emerge are not quite as uniform as those found in nesting structures but also not as chaotic as those in random automa. You can see an example of each type of automata below:



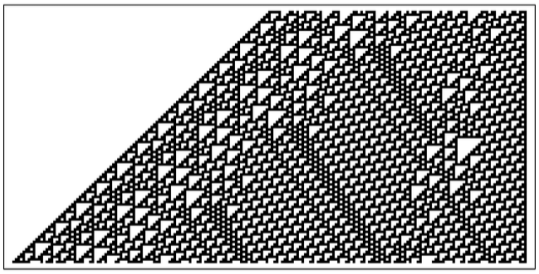
repetition (rule 250)



nesting (rule 90)

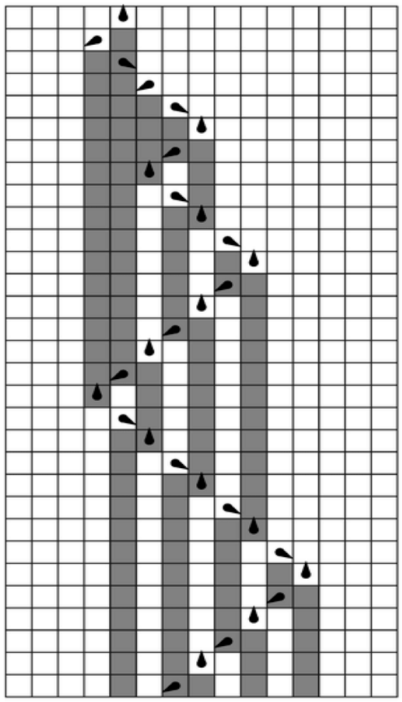


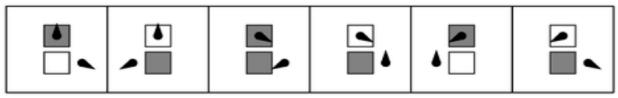
randomness (rule 30)



localized structures (rule 110)

Wolfram's book delves deeper into more interesting concepts, such as adding more complex rules, such as in mobile automata, and expanding on those rules to create a turing machine in cellular automata. as seen below:





An example of a Turing machine. Like a mobile automaton, the Turing machine has one active cell or "head", but now the head has several possible states, indicated by the directions of the arrows in this picture.

We will not be discussing these in much depth for the sake of brevity, but also because some of the more complex concepts such as substitution systems and register machines have little bearing on how we will implement a more simple elementary cellular automata.

Additionally we referenced a paper by Ioannis Karafyllidis, Adonios Thanailakis, called *A model for predicting forest fire spreading using cellular automata* to help guide us on how to implement our own fire spreading cellular automata.

Approach

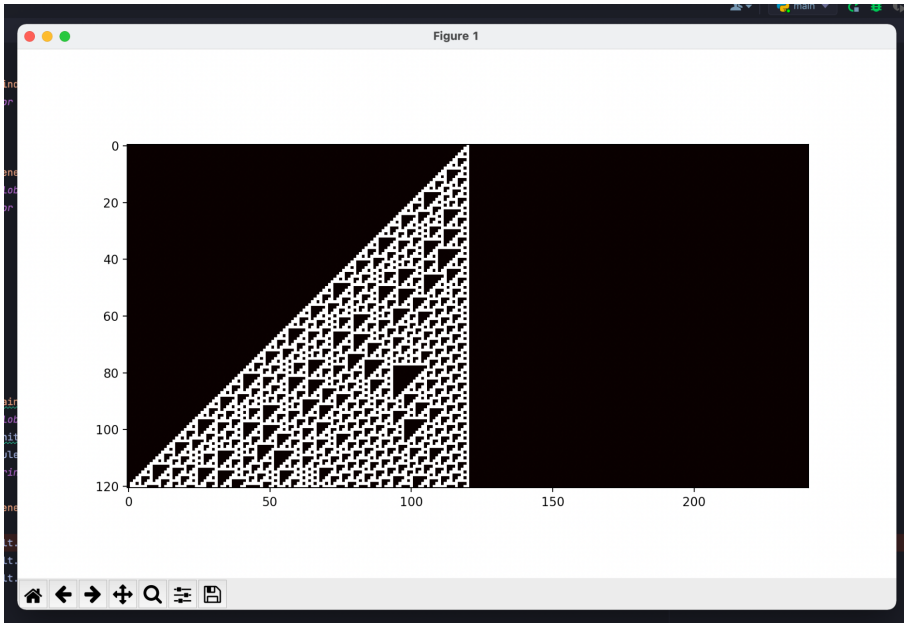
Our approach was to first implement our elementary cellular automata and our fire spread automa separately in python. As even though both are types of automata, they both have different set up requirements in how they are initialized and how rules are processed. Both automta were implemented in Python as it is a language that is quite popular within the mathematics and data modeling communities with lots of resources available, including several useful visualization libraries like matplotlib, which we used in our project.

While the fire spread automata was also built using matplotlib and python the rest of the structure was completely different. The initial state of the forest would be generated completely randomly at runtime based on a density parameter. Then each time step any cell that was marked as a tree had a slim chance of randomly catching fire. If a tree was next to a cell that was burning then there would be some probability that tree would burn in the next time step. T

Experiment Setup

For the elementary automata we implemented 8 rules, at least one from each of the four different categories that can be defined at runtime by the end user to visualize the resulting pattern. As the automata itself is deterministic the resulting pattern for any given rule, and the provided initial state, should always result in the same pattern. The cellular automata has a max depth 120, though that can be altered within the code, large enough to notice certain patterns emerging but small enough to not bog down my computer.

An one dimensional array was used to store the initial state with only the center most cell being set to one, while the rules for the Elementary automata were stored in a dictionary to make looking rules quick and efficient. Each time step we would generate a new state by iterating over the previous time step state using a function to get the three cells above the current cell we were trying to calculate, those three cells would then be used to look into the dictionary containing the rules to generate the current cell's new value. the new state would then be appended to the array containing the initial state and then output using matPlotLib to produce results that looked this:



For our fire spread automata, after initiating a forest at random cells that contain a tree have a small chance to start burning at random as if they had been struck by lightning. These tree cells that have “been hit by lighting” act as nucleation sites from which fire can spread to neighboring trees.

From here our forest is no longer in a static state as now the fire spread rules can spring into action and do all the heavy work. The actual rules for fire spreading are pretty simple as well. Using a Moore neighborhood model trees next to a burning cell have a chance to catch fire themselves, though trees at a diagonal have a lower probability.So fire spread is a local fenomany contained only to the vicinity of burning trees. Burning cells only last for one time step before they are extinguished and replaced by empty so fires cannot just continue to burn until every block is on fire. While simple these rules can lead to very interesting, emergent behaviors that we will talk about more in depth in the next section.

Results

The results for our elementary automata were as we expected. Seeing as rules were all implemented correctly and the logic for determining a cell's value from the previous state is working correctly. Our results looked very similar to what Wolfram described in his book in terms of structure. Which is to be expected as elementary automata are deterministic and should always produce the same result no matter who is running them.

More interesting were the results of our fire spread automata. The output animation seemed to vary drastically depending on values of different parameters such as initial density, fire spread probability and lighting strikes. Highly dense forests with a greater chance for fire to spread burned almost in their entirety, but by just reducing the fire spread probability we got

completely different results. Instead of a complete burn leaving only bare ground, we now got patches of barren ground with some patterns that looked like lightning strikes, making long shallow burns in the forest. The burn pattern was also different, instead of looking like a wall of fire that was decimating everything in its path, the fire was more contained and instead looked like a smoldering ember slowly burning through veins of fuels. Animation for Both can be found in the presentation on slide 14.

Conclusions and future work

The ability for Cellular Automata to exhibit complex emergent behaviors from very basic and simple rules sets and Due to the nature cellular automata of being well relatively simple to execute using computers, there exists great potential use in the simulation of interconnected systems. A great potential application exists in chemistry where the certain laws could be simplified to simulate certain chemical processes like the Belousov–Zhabotinsky reaction. Another possibly useful application cellular automata may be in biology and simulating the immune system with more complex ruleset to more closely resemble the real world.

Currently our fire simulation does not have much merit on its own as it a rather crude and overly simplified model of the real world in the future I would like to greatly expand upon my work here by adding mode cell types that can represent different feature we find in nature such as grasslands, bushland, and natural barriers like rivers to better simulate how fire could be spread or blocked by certain natural features. To make this model even more accurate it would be pertinent to incorporate topological and meteorological data into the simulation to accurately

model the effect elevation and wind that can play in the spreading and containment of fire. I believe that incorporating those feature would greatly bolster the utility of our automata and may even help guide policy on fire prevention if accurate enough.