The Edge of Chaos Phenomena

Conner Brinkley

02.05.2020

Biologically-Inspired Computing

The University of Tennessee

INTRODUCTION

The purpose here is to investigate one-dimensional cellular automata and how we can more reliably achieve complex "Edge of Chaos" behavior, by observing how lambda (λ) and entropy (*H*) values affect the states of the automata. This behavior (what we will call Wolfram class IV later on) has valuable properties that could be used in future areas of computing, so it is useful for us to know how to replicate it. To do this, we need to find the best λ and *H* values that correlate with typical class IV automata, so 40 unique experiments were conducted in hopes of collecting enough data and achieving this.

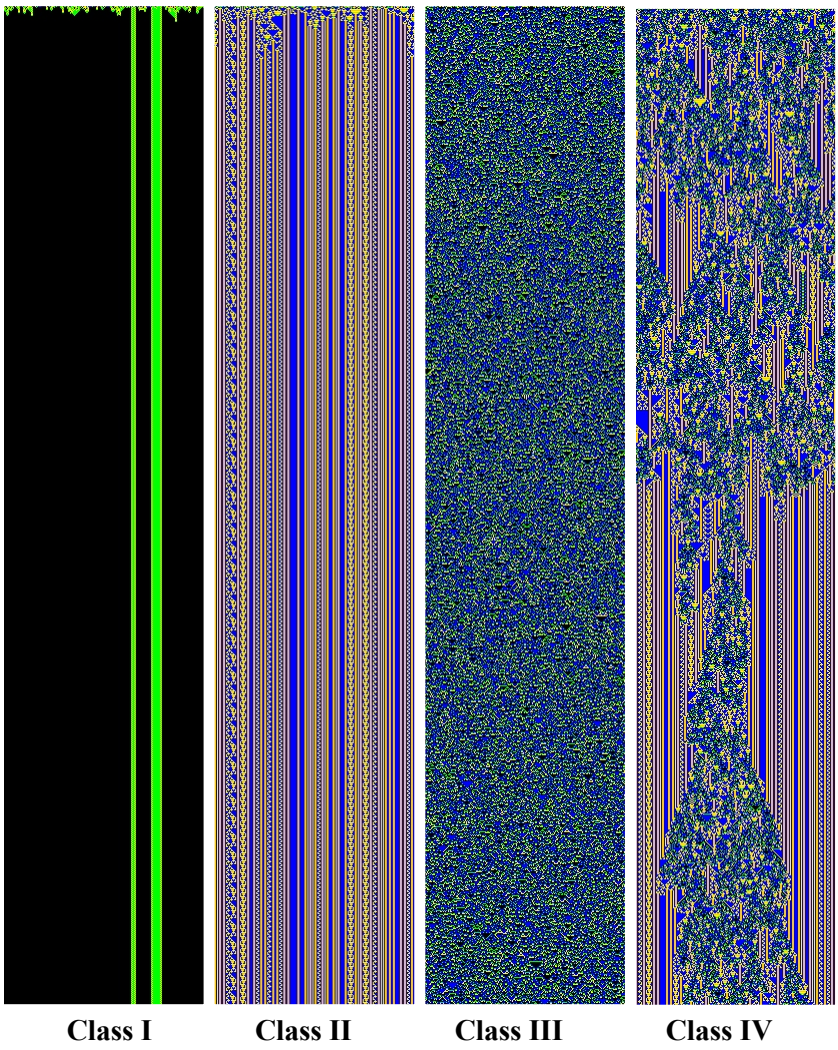
PROCESS (& THEORY)

To generate an experiment, we need a list of 12 random integers between 1 and 4 called the rule string. This rule string is used in the simulator to show how the one-dimensional automata will behave. Once we observe the behavior and assign it a Wolfram class (more on that below), then we want to zero out one of the numbers in the rule string and observe what happens before assigning it another class. This is called a step, and each experiment has 13 of them. This keeps repeating until the rule string is entirely zeroed out on the 13^{th} step and we reach a fixed state. This entire process is repeated 39 more times in order to obtain meaningful data.

Wolfram's classification table for one-dimensional automata is relatively simple, which is as follows:

Class I:	Evolve to fixed, uniform states
Class II:	Evolve to periodic patterns
Class III:	Generally chaotic noise
Class IV:	Complex patterns with seemingly no structure (the "Edge of Chaos" class)

This can be a little confusing to visualize without seeing it, so it might be better to compare the difference side by side. Below are four examples of one-dimensional cellular automata simulations, labeled by the class that each one falls under.



Wolfram's classification of one-dimensional cellular automata.

Above we can see that the class III automata mostly just looks like pure static, and for the most part, this is generally true for a lot of this class. Class IV gets interesting, showing clear signs of chaotic behavior with holes in the simulation that have no apparent pattern. Generally, class IV can be really tricky to spot, since a lot of the time it looks very similar to class III noise. Unfortunately, to get a clearer understanding you would need to sift through many more different examples, but for our purposes, hopefully this helps visualize it.

For each step we run, we want to calculate four meaningful values: lambda (λ), entropy (H), totalistic lambda (λ_T), and totalistic entropy (H_T). The totalistic parameters are simpler to calculate, and they are based on the frequency of each state in the totalistic rule table. These two values end up being the most useful in our analysis, so we will focus on them. Below is the formula to calculate totalistic lambda:

$$\lambda_T = \frac{S - m_0}{S} = 1 - \frac{m_0}{S}$$

This is true where S is the size of the totalistic table, and m_0 is the number of table entries that map into a new state of 0. Now, totalistic entropy can be calculated by:

$$H_T = -\sum_S p_S \lg p_S$$

Totalistic entropy is a function of its probability distribution, where $p_S = {}^{m_S}/_{S}$. Once all of the calculations were made and stored in a CSV file, the next step was to go through all 480 automata images generated by the simulation for all of the experiments and classify them based on Wolfram's automata classes. To make things easier for graphing the data, I assigned classes I and II a numeric value of 0, class III a value of 2, and class IV a value of 1.

Once all of the data was collected, I took every instance of a class IV step, and calculated the mean and standard deviation for each of the four parameters. What this tells us is the average value for each and how spread out the data is. For example, if we have an average λ value of 2.5 with a standard deviation of 0.25 and an average *H* value of 2.1 with a standard deviation of 0.12, that tells us that the entropy value is probably more likely to predict class IV behavior since the data is overall less spread out than lambda. This is very important later.

The final step was to create four separate graphs for each of the four parameters, and graph their values related to the class I assigned them. This is also useful later because it shows visually where the data points cluster.

CALCULATIONS

All of the table walkthrough calculations were completely automated by a Java program written by a former student – the source code can be found in the CASimulator.java file. For this project, I ran 40 randomly seeded experiments with the Java program, which generated a master CSV file with the already calculated λ, λ_T, *H*, and H_T values for each experiment.

For calculating the average and standard deviation for each of the four parameters, I created a new sheet only for the steps that resulted in class IV behavior and used Microsoft Excel's built in *average* and *stdev* formulas. All four graphs were also created with Excel's plotting tool.

RESULTS

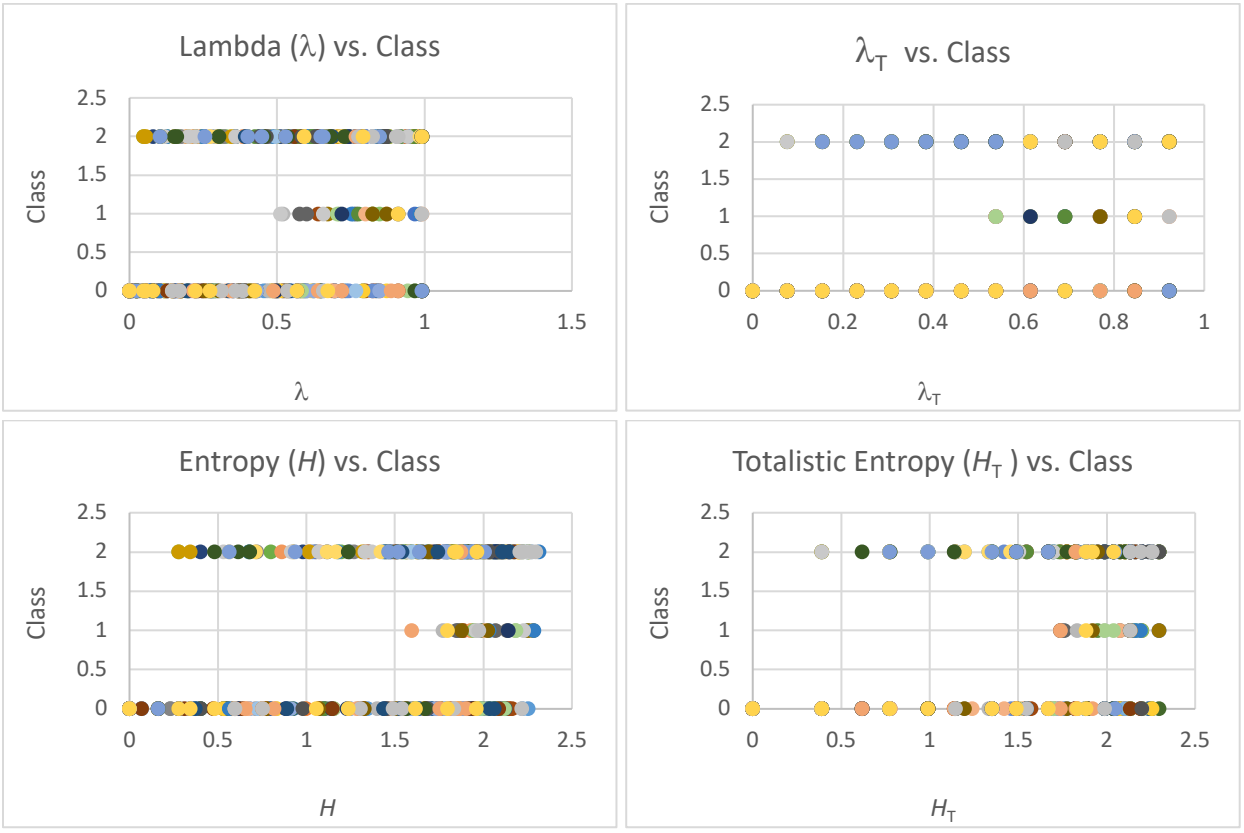
Below is the table with the computed averages and standard deviations for the class IV automata states.

CLASS IV CALCULATIONS

	λ	λ_T	H	H_T
Average	0.7775	0.7236	2.0159	2.0445
Standard Deviation	0.1393	0.1267	0.1733	0.1698

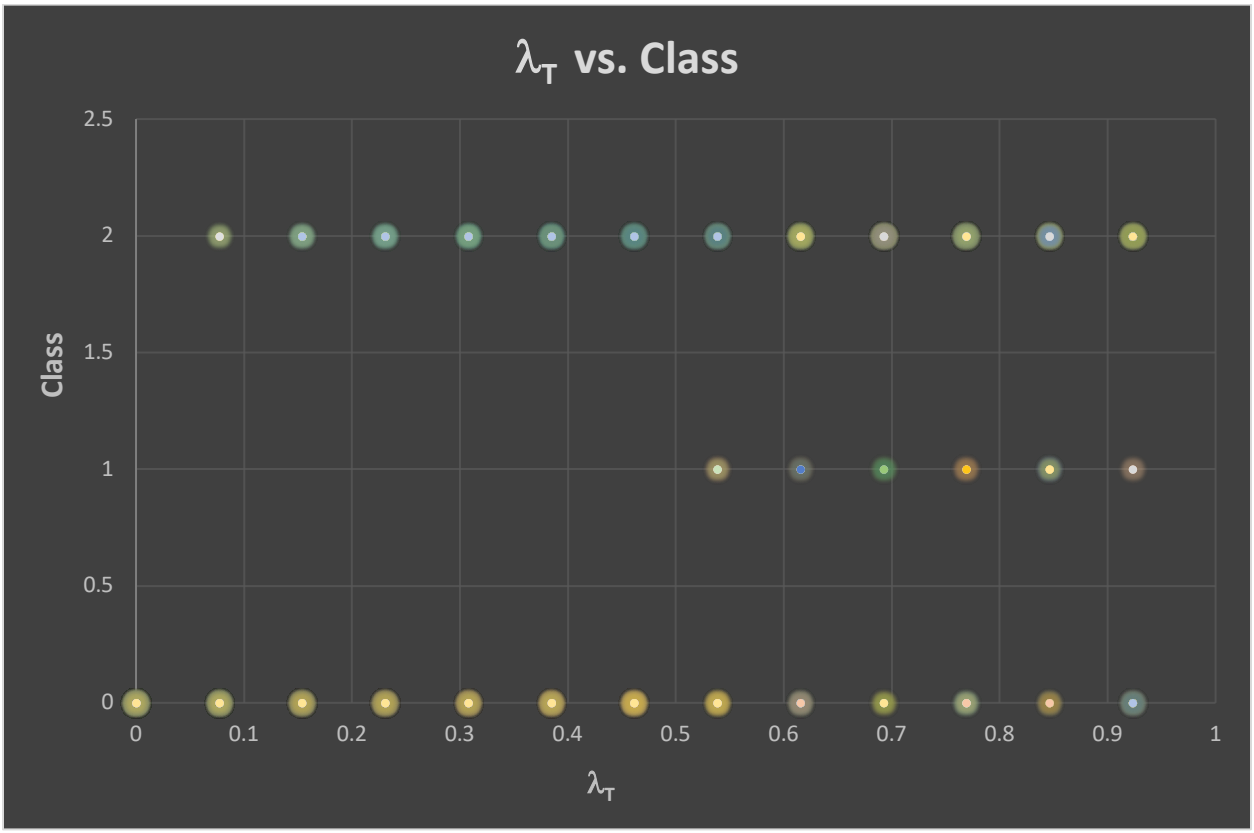
The calculations from Excel for class IV states.

Just by looking at the standard deviations, it appears that λ_T would be the most useful parameter in predicting class IV behavior since it is the lowest, but one thing these calculations do not take into account is the percentage of deviation from the average. This is easy to overlook until you look at all four of the generated graphs.



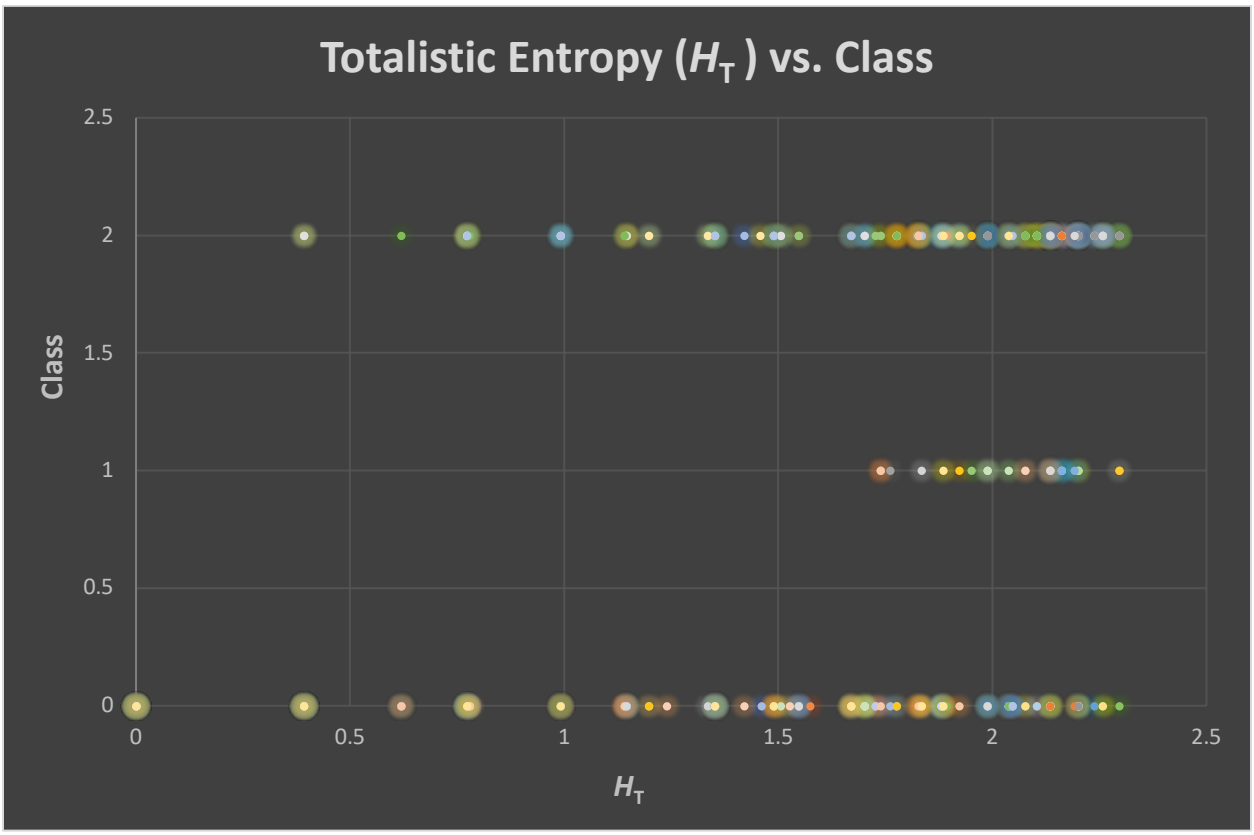
The four parameters compared to Wolfram's one-dimensional automata classification.

Earlier, I mentioned that the totalistic values were more useful in determining class IV behavior, so let us examine them a bit closer.



The totalistic lambda graph shows very evenly distributed data, despite having the lowest standard deviation.

Notice how distributed the λ_T graph looks for class IV (numeric class 1) – it is clearly the most distributed out of the rest of the graphs. For a meaningful average, it would make sense to have more clustering around the mean instead of an even distribution, even if the range is small. This safely eliminates λ_T from being a contender for the most useful parameter. Now let us take a look at H_T.



Notice how the rest of the data clusters around the calculated average totalistic entropy.

Looking back at our calculations, the average class IV totalistic entropy value was 2.0445. In this graph it is a lot more apparent that the data points revolve around that average, due to the clustering around 2. This makes H_T seem more reasonable to consider it as the most correlating parameter to class IV.

CONCLUSION

Going off of standard deviation alone, it would be easy to say that totalistic lambda correlates the most to class IV behavior. However, if we dive a little deeper and look at the graphs, we can quickly see that the lambda parameters are more evenly spread out, implying that the average value is less likely to be accurate for predicting class IV. The entropy parameters, on the other hand, seem much more accurate due to the clustering around the average. Totalistic entropy seems to be the best indicator, but just slightly, given its graph and standard deviation.

In the future, it might be interesting to investigate a little further and remove the class IV states that were on the borderline of class III to see if that alters and gives more accurate results.