COSC 420/427/527: Biologically-Inspired Computation

Project 1: "Edge of Chaos" in 1D Cellular Automata

Due: Friday, February 9, 11:59 PM

Introduction

In this project you will explore "Edge of Chaos" phenomena (Wolfram class IV behavior) in 1D cellular automata. You will do this by systematically modifying randomly generated transition tables and observing the lambda and entropy values associated with phase changes in the behavior of the automata.

Experimental Setup

Simulators

You can use an "off the shelf" simulator for this project. One is the NetLogo version on the class website: <http://web.eecs.utk.edu/~mclennan/Classes/420-527/NetLogo/CA 1D General Totalistic.html>. See the instructions on that page about running the simulator on your own computer.

Alternately, you can use the **ca** simulator provided with Flake's *Computational Beauty of Nature*. There are several different versions that you can use; see <http://mitpress.mit.edu/sites/default/files/titles/content/cbnhtml/home.html>.

Perhaps the best solution is to use the java program hosted on Dr. Van Hornweder's help page: <http://web.eecs.utk.edu/~mclennan/Classes/420-527/projects/project1/ca.html>. This page also contains submission instructions, additional tips, and classification examples and images. Make sure to look at it carefully before starting on your project!

Finally, if you want, you can implement your own CA simulator; it's not very hard. Since programming the simulator is not part of the assignment, feel free to work together to implement a simulator, or share code or ideas.

As I said, it's up to you which you use. You will probably be doing some cutting and pasting of rule strings between the simulator and your λ-*H* calculator (see below), so you will have to find a way of doing the experiments that is not too tedious. You might want to write a shell script to automate some of the procedure. The program on Van Hornweder's webpage automates almost everything.

Table-walk-through Procedure

As explained in class, we cannot simply allow the simulator to pick a random rule string with a certain λ value, since there is too much variability between unrelated rule strings. Therefore we will use Chris Langton's table-walk-through method, which involves generating a series of rule strings differing only in the number of quiescent entries. The following describes an operational procedure.

Pick a random seed and record it. See the last page of this handout for a form that you can use for recording your experiments (or invent your own spreadsheet); for your convenience, a blank form is available online: <http://web.eecs.utk.edu/~mclennan/Classes/420-527/handouts/Experiment-Record-1.doc>.

Each experiment is conducted on a random rule string. The easiest way to get this is to have your simulator generate one (use the button in the NetLogo simulator, or set `lambda`=1.0 in the **ca** simulator). (You can also generate one yourself by picking 12 random integers in the range 1 to 4, since we exclude the quiescent state; the first number is always 0, giving 13 all together.) Make sure to record your random rule string.

You will now *decimate* your rule string by zeroing one non-zero entry at a time. For your original rule string and for each of the decimated rule strings you will compute a λ, λ_T, H, and H_T value. Whether you compute them as you go along or do them all at the end is up to you; which is easier will probably depend on the cutting-and-pasting or other experimental procedure that you have selected. Note that the NetLogo simulator computes them for you. You will also observe and record the behavior (I, II, III, or IV) of the original string and each of its decimations.

For each decimation, randomly select one of the non-zero entries in the rule string and set it to zero. Observe the behavior of the resulting CA and record it. Note: if you are using the Java version of the **ca** simulator, make sure you set `lambda=-1` (or any negative value) before you enter your decimated string, or it will ignore your string and regenerate the undecimated string! If you are using the NetLogo simulator, just press the Decimate button.

I think you should do at least 20 table walk-throughs to have meaningful statistics. However, you can do more, and better investigations will earn higher grades.

Lambda and Entropy Calculations

You will be investigating lambda and entropy values computed in two different ways. (The NetLogo simulator computes these for you, but you still need to understand what they mean.) The simplest way gives the totalistic parameters λ_T and H_T, which are based on how frequently each state appears in the totalistic rule table (that is, the rule string that you have recorded). The minimum sum of the neighborhood states is 0 and the maximum is $N(K-1)$, where *K* is the number of states and $N=2r+1$ is the size of the neighborhood for radius *r*. Therefore the size of the totalistic table is $S=N(K-1)+1$. In your case *r* = 1 and *K* = 5, so *S* = 13. Let *R* be the rule table so that R_k gives the new state for the sum *k* of the neighborhood states. We are interested in the distribution of new-state values, so let $m_s = \left|\{R_k \mid R_k = s\}\right|$, that is, m_s is the number of table entries that map into new state *s*. The totalistic lambda value λ_T is simply the fraction of totalistic rule table entries that *do not* map into the quiescent state *s* = 0:

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

The entropy of the totalistic rule table is a function of its probability distribution. Therefore let $p_s = m_s/S$ so that the entropy H_T of the totalistic transition table is defined:

$$H_T = -\sum_s p_s \lg p_s ,$$

where $\lg x$ is the logarithm of *x* to the base 2. (Note that $0 \lg 0 = 0$, which you will have to handle as a special case since $\lg 0 = -\infty$.)

To facilitate comparison with non-totalistic CAs, the λ and *H* values are usually defined over the complete (non-totalistic) transition table, which has $T = K^N$ entries. However, we have an abbreviated rule table, of length $N(K-1)+1$, which gives the new state R_k for a sum *k*; in most cases there are multiple configurations having the same sum.

Therefore, let C_k be the number of neighborhood configurations having the sum *k*. For *K* = 5 and *r* = 1, C_k is given by the following table:

k	0	1	2	3	4	5	6	7	8	9	10	11	12
C_k	1	3	6	10	15	18	19	18	15	10	6	3	1

Define n_s to be the number of configurations leading to state *s*:

$$n_s = \sum_{\{k \mid R_k = s\}} C_k .$$

That is, n_s is the total number of configurations that are mapped into state *s* by the corresponding non-totalistic rule table. The probability of a new state in the complete transition table is given by $p_s = n_s/T$. (Note: these p_s are different from the p_s defined above for H_T!) Then Langton's λ is defined:

$$\lambda = \frac{T - n_0}{T} = 1 - n_0 \Big/ T = 1 - p_0 \ .$$

The entropy H of the complete transition table is defined:

$$H = -\sum_s p_s \lg p_s \ .$$

Since you will be computing the λ, λ_T, H, and H_T values for each rule table, you will have to use a program to do it. This is not an important part of the project, so you can do it in any way convenient. You can implement your own program, or several (or all) of you can get together and share a program. Also, some versions of the **ca** program will compute λ (but not λ_T, H, and H_T) for you.

Additional Work for COSC 427 and 527 Students

If you are taking this course for honors or graduate credit (**COSC 427/527**), in addition to the foregoing four parameters, you will need to compute a fifth parameter for the rules you test. But you get to pick or design the parameter. So think about what property of the transition tables might be correlated with the behavior class of a CA and figure out a way to compute it. Your writeup (see below) should include a short (one paragraph) motivation for your parameter (i.e., what makes you think that it might be correlated with behavior) and a description of its computation. Your parameter might not turn out to be very good, but that doesn't matter; the point is to make a reasonable conjecture and to test it. In your writeup, evaluate your new parameter in the same ways you do the others (λ, λ_T, H, H_T).

Students taking this course for undergraduate credit (**COSC 420**) can also do this part, which will earn you extra credit.

Writeup

Calculations

Compute the average and standard deviation of the λ, λ_T, H, and H_T values for all simulation instances that exhibit class IV behavior. Which (λ, λ_T, H, or H_T) seems to be a more reliable indicator of class IV behavior?

Graphs

Make four graphs of behavior vs. λ, λ_T, H, and H_T. That is, use λ, λ_T, H, and H_T for the abscissas (x-axes) of the four graphs. (COSC 427/527 students should, in addition, make graphs of their fifth parameter.) For the ordinates (y-axes) of the graphs, use the following numerical values to indicate qualitative behavior: 0 for classes I and II, 1 for class IV, and 2 for class III. Each of your graphs should show all of your experiments as separate curves; try to use colors or other ways of making the curves distinguishable.

Discussion

Draw some conclusions about the range of values of λ, λ_T, H, and H_T that lead to class IV behavior. Note any anomalies. Did you ever observe class I or II behavior at high λ, λ_T, H, and H_T values? Did you ever observe complex (IV) or chaotic (III) behavior at λ, λ_T,

H, and H_T values that were otherwise in the simple (I, II) region? How do you explain these anomalies?

Extra Credit

The table above for the number of configurations is given by $C_k = D(k, 3)$, where $D(k, N)$ is the number of states, each in the range 0 to $K-1$, arranged in a neighborhood of size N, so that the states add up to k. (In other words, the number of solutions to $\sum_{i=1}^{N} X_i = k$, where $0 \le X_i \le K-1$.) It can be computed by the following equations:

$$D(k, 1) = \begin{cases} 1, & k < K \\ 0, & k \ge K \end{cases},$$

For $n > 1$:

$$D(k, n) = \binom{n+k-1}{k}, \quad k < K.$$

$$D(k, n) = \sum_{j=0}^{K-1} D(k-j, n-1), \quad k \ge K.$$

I will give extra credit to anyone who can find a *significantly simpler* formula for D and *prove it is correct*.

Submission

Tar or zip your files together and submit them via Canvas. The directory should include your report (in .pdf format) and the .csv or .xls experiment sheet.

Experiment Record: 1D CA "Edge of Chaos"

Your Name: Bruce MacLennan .

simulator: [✓] NetLogo, [ ] NetLogo applet,
CBN: [ ] java, [ ] unix, [ ] mac, [ ] windows, [ ] other: _____
K (states) = 5. *r* (radius) = 1
wrap: [✓] quiescence: [✓]

random seed = 1234567 . Experiment Number: 1 .

Random Rule Table:

0	2	1	2	1	4	2	4	1	1	1	1	1
0	1	2	3	4	5	6	7	8	9	10	11	12

Table Walk-through:

Step	Entry Zeroed	Class	λ	λ_T	H	H_T	Observations
0	—	III	0.992	0.923	1.595	1.669	Complex pattern, near IV
1	1	III	0.968	0.846	1.684	1.727	Same
2	4	**IV**	**0.848**	**0.769**	**1.947**	**1.834**	near III
3	8	I	0.728	0.692	1.988	1.884	Very long IV transient
4	9	I	0.648	0.615	1.916	1.884	ditto
5	3	I	0.568	0.538	1.833	1.738	
6	5	I	0.424	0.462	1.654	1.573	
7	10	I	0.376	0.385	1.532	1.489	
8	2	I	0.328	0.308	1.360	1.352	dies very quickly from here on down
9	11	I	0.304	0.231	1.235	1.145	
10	6	I	0.152	0.154	0.660	0.773	
11	12	I	0.144	0.077	0.595	0.391	
12	7	I	0	0	0	0	