Cellular Automata

Nicholas Geis

January 22, 2015

In Stephen Wolfram's book, *A New Kind of Science*, he postulates that the world as we know it and all its complexities is just a simple Sequential Dynamical System that is governed by a very simple set of rules. All of Physics, all of Biology, all of Mathematics are interlinked through those simple rules. Now, that seems just absolutely absurd and downright crazy. How can such a simple set of rules form the complex theories of Mathematics and Physics and the seemingly random processes of Biology? This week, we will examine the field that Wolfram himself forged and look at how simple rules can lead to very complicated structures. This field is called Cellular Automata.

Now, what exactly is Cellular Automata? Cellular Automata, or CA[1], has several features that are similar to the features of an SDS. These are traits that they share:

1. There is a collection of vertices that are from the set V.

2. There is a set, S, that characterizes the states of said vertices. For most CA, that set is $S = \{0, 1\}$. Those are the simplest of states and the ones that we will be working with the most in this class.

3. There is a defined function f_v that interacts with the vertices to output a new vertex or update an older one.

From there, differences start to arise between the two. Those differences include

1. Unlike SDS, CA always updates simultaneously. As in, each vertex updates at the same time. Each update is usually seen as one time step forward and thus the system evolves in discrete time steps.

2. CA usually consists of uniform structure[2], and as a result, all graphs that define a CA are always regular graphs. For example, $\mathbb{Z}^k$ where $k > 1$. This produces the simple geometric grid that everyone knows and loves when $k = 2$.

3. The edges of the graph form neighborhoods with all adjacent vertices. In general, we define a neighborhood, $n[v]$, as follows:

[1] This abbreviation is the accepted shorten for Cellular Automata

[2] This is intended because it includes translation invariance. Which implies that, at least in one direction, the object is infinite. As in, for any point $\mathbf{p}$, the set of points with the same properties due to translational symmetry form the infinite discrete set $\{\mathbf{p} + n\mathbf{a} | n \in \mathbb{Z}\} = \mathbf{p} + \mathbb{Z}\mathbf{a}$ (Wikipedia [4]).

We take a sequence N and define it as $N = (d_1, ..., d_m)$, where $d_i \in Y$ and Y is our strongly regular graph. Then define $n[v]$ to be

$$n[v] = v + N = (v + d_1, ..., v + d_m).$$

In short, take whatever grid structure your Graph, Y, and let the neighbors of v be what you think they are. Just the points on the grid surrounding vertex v. That's your neighborhood in $\mathbb{Z}^2$. However, changes can be brought to this definition that allow for a different idea of neighborhoods. The two most commonly used definitions are the von Neumann neighborhood and the Moore neighborhood. To put them simply, the von Neumann neighborhood for a basic grid structure in $\mathbb{Z}^2$ is the center box and then the boxes that border its sides. Only those 5 boxes constitute the neighborhood. And the Moore neighborhood is the von Neumann neighborhood with the addition of the corner boxes. So it forms a 3X3 grid of boxes that is made up of 9 squares.

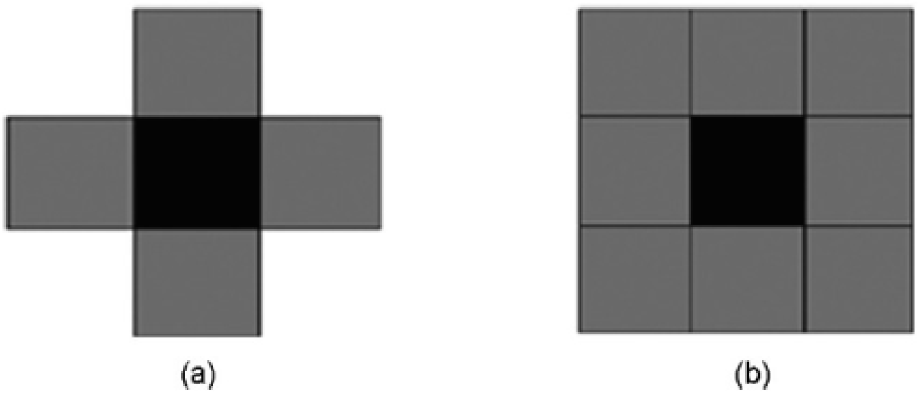


Figure 1: (a) is the von Neumann neighborhood and (b) is the Moore neighborhood.

Those are the main similarities and differences between CA and SDS (Wolfram [2]). From here, we will continue to work with CA but only in the simplest cases (most likely for the rest of the quarter). The simplest of CA actually have a formal name. They are called the Elementary CA.

These are the following rules to Elementary CA:

1. Our graph will be $\mathbb{Z}^2$. In this graph, each box will belong to our set of vertices V. And the states of each vertex is in S and $S = \{0, 1\}$.

2. Our function f will map the following

$$f \ : \ \mathbb{F}_2^3 \ \rightarrow \ \mathbb{F}_2$$

   where $\mathbb{F}_2 = \{0, 1\}$. In layman's terms, we are taking a string of length 3 that has a combination of 1's and 0's in it and mapping it to either a 1 or a 0.

3. Our neighborhoods will consist of a vertex, v, and the vertices to the right and to the left. Only 3 blocks will belong to each neighborhood.

Those are the rules to Elementary CA (Mortveit [1]). Now, some may ask how many different configurations are there for Elementary CA and how can we possible name each one in an organized manner? Well I will show you.

Claim: There are 256 possible CA configurations that follow the Elementary rules.

Proof: This is actually quite trivial. We know that we get to choose if there is a 1 or a 0 for the mapping so we have 2 choices but how many things are we mapping. In short, we are doing the following

$$\#\ of\ configurations \ = \ |\mathbb{F}_2|^{|\mathbb{F}_2^3|} = 2^8 = 256.$$

We have 8 times where we have to decide if its a 1 or a 0 so that would be 2^8. □

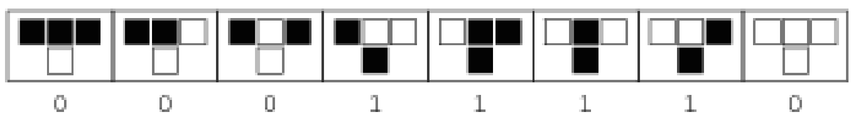


Figure 2: If you had difficulty following the proof above, here is a simple visual aid that can help. We have eight possible states of the three blocks in the top portion and then we assign an output to each of those states. This particular set is called Rule 30.

Now, is there a numbering system that can give a number to each configuration? Well, yes there is. Stephen Wolfram developed the method for this. He did this by converting the binary string of the eight new boxes into decimal. First, we define x to be the string of 3 digits in $\mathbb{F}_2^3$. Then, we let f at x be a_k for $0 \leq k \leq 7$. And finally we can define another function r that will convert f into a decimal number.

(x_{i-1}, x_i, x_{i+1})	111	110	101	100	011	010	001	000
f	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0

$$r(f) = \sum_{i=0}^{7} a_i 2^i$$

That equation he found will enumerate each of the 256 Elementary rules. Let's go through an example of this enumeration though.

Example: What is the number for the following table?

(x_{i-1}, x_i, x_{i+1})	111	110	101	100	011	010	001	000
f	0	1	1	0	1	1	1	0

For this example, all we need to do is just plug into the equation.

$$r(f) = \sum_{i=0}^{7} a_i 2^i = 0 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 110$$

And thus this is Rule 110. This rule in particular is quite famous because it is one of the well known class 4 Cellular Automata and is also known to be Turing Complete[3]. Now, what is "class 4" and why does this even matter? Well, Elementary CA and CA in general can be classified into 4 distinct categories based on complexity (Wikipedia [2]):

1. Class 1: Nearly all initial patterns evolve quickly into a stable, homogeneous state. Any randomness in the initial pattern disappears.

2. Class 2: Nearly all initial patterns evolve quickly into stable or oscillating structures. Some of the randomness in the initial pattern may filter out, but some remains. Local changes to the initial pattern tend to remain local.

3. Class 3: Nearly all initial patterns evolve in a pseudo-random or chaotic manner. Any stable structures that appear are quickly destroyed by the surrounding noise. Local changes to the initial pattern tend to spread indefinitely.

4. Class 4: Nearly all initial patterns evolve into structures that interact in complex and interesting ways, with the formation of local structures that are able to survive for long periods of time. Class 2 type stable or oscillating structures may be the eventual outcome, but the number of steps required to reach this state may be very large, even when the initial pattern is relatively simple. Local changes to the initial pattern may spread indefinitely. Wolfram has conjectured that many, if not all class 4 cellular automata are capable of universal computation. This has been proven for Rule 110 and Conway's game of Life.

Class 4 is the meat that CA research takes part in. Strange things start to arise out of patterns that logic assumes that simple patterns will eventually occur, but none never do.

With that stated, that is the end of these notes for this week. The next few pages will all contain a few images of rules and their visual outcomes when processed a several generations forward (Wolfram [5]).

References

[1] H. S. Mortveit and C. M. Reidys, *An Introduction to Sequential Dynamical Systems*, Springer Science+Business Media LLC, New York, 2008.

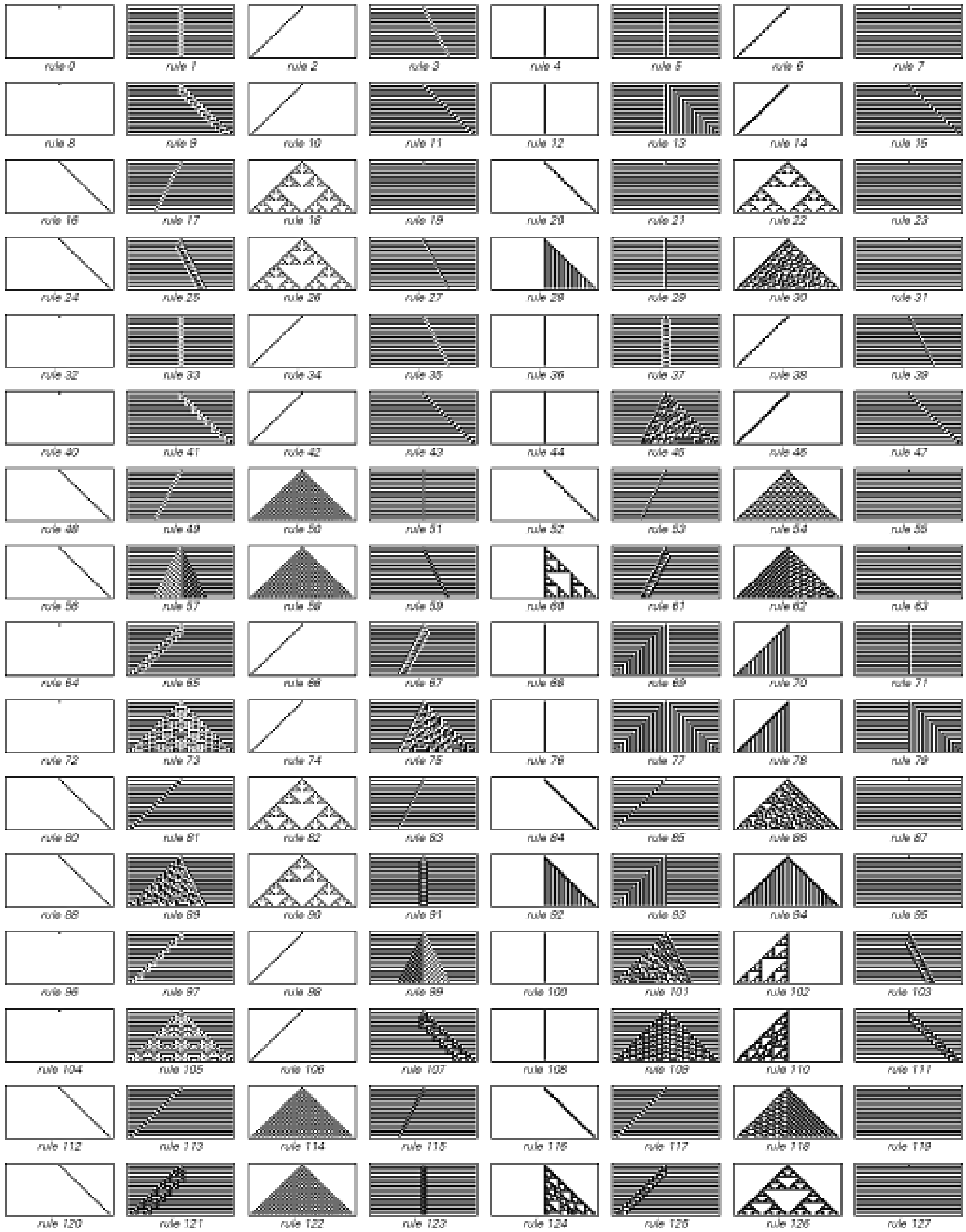
[2] http://en.wikipedia.org/wiki/Cellular_automaton. Wikipedia, January 20, 2015.

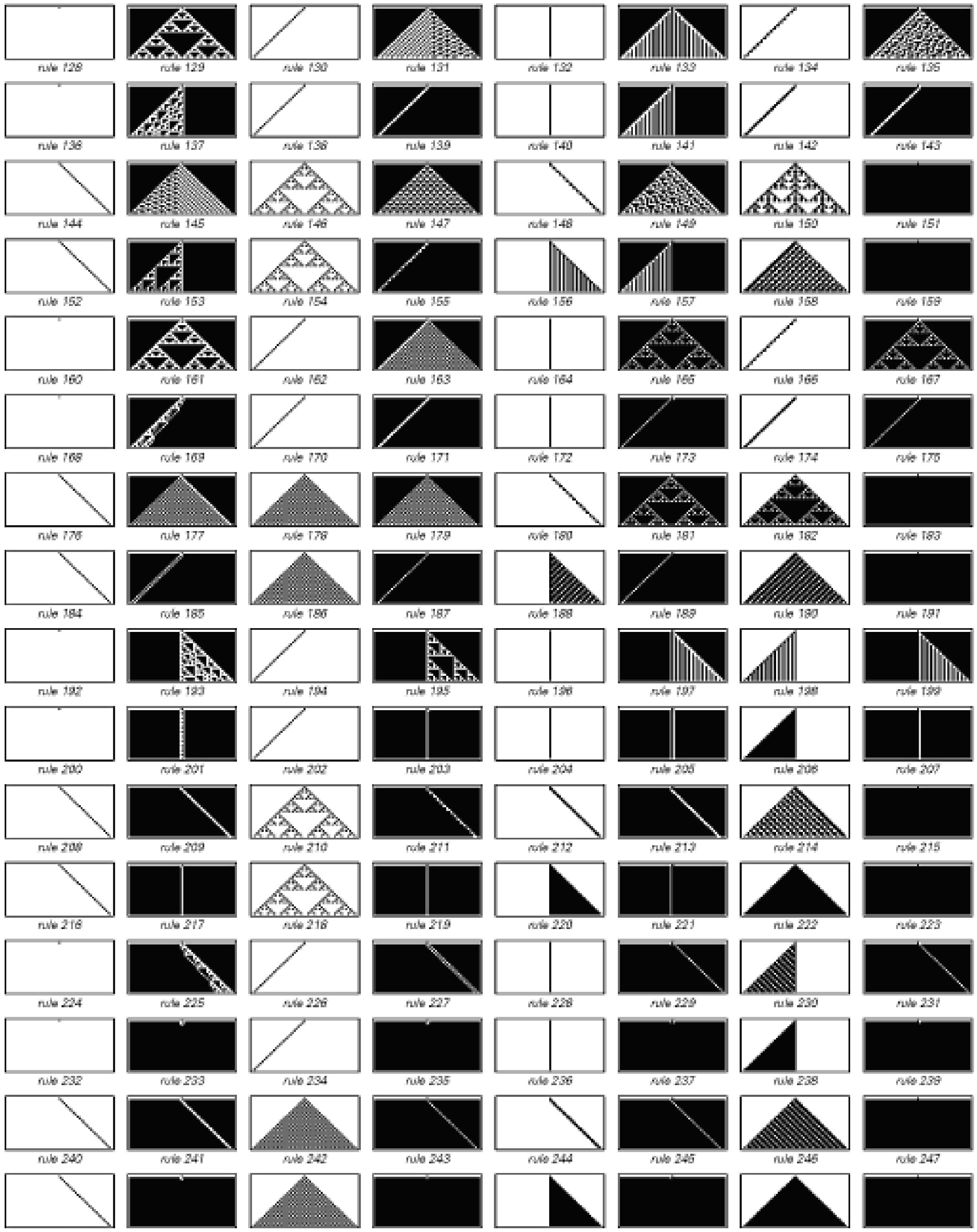
[3] http://en.wikipedia.org/wiki/Rule_110. Wikipedia, January 20, 2015.

[4] http://en.wikipedia.org/wiki/Translational_symmetry. Wikipedia, January 20, 2015.

[5] Wolfram, Stephen. *A New Kind of Science*. Champaign: Wolfram Media, Inc., 2002

[3] This was discovered by Matthew Cook and published in 2004, and means that it is capable of supporting universal computation (Wikipedia [3]).


rule 0
rule 1
rule 2
rule 3
rule 4
rule 5
rule 6
rule 7
rule 8
rule 9
rule 10
rule 11
rule 12
rule 13
rule 14
rule 15
rule 16
rule 17
rule 18
rule 19
rule 20
rule 21
rule 22
rule 23
rule 24
rule 25
rule 26
rule 27
rule 28
rule 29
rule 30
rule 31
rule 32
rule 33
rule 34
rule 35
rule 36
rule 37
rule 38
rule 39
rule 40
rule 41
rule 42
rule 43
rule 44
rule 45
rule 46
rule 47
rule 48
rule 49
rule 50
rule 51
rule 52
rule 53
rule 54
rule 55
rule 56
rule 57
rule 58
rule 59
rule 60
rule 61
rule 62
rule 63
rule 64
rule 65
rule 66
rule 67
rule 68
rule 69
rule 70
rule 71
rule 72
rule 73
rule 74
rule 75
rule 76
rule 77
rule 78
rule 79
rule 80
rule 81
rule 82
rule 83
rule 84
rule 85
rule 86
rule 87
rule 88
rule 89
rule 90
rule 91
rule 92
rule 93
rule 94
rule 95
rule 96
rule 97
rule 98
rule 99
rule 100
rule 101
rule 102
rule 103
rule 104
rule 105
rule 106
rule 107
rule 108
rule 109
rule 110
rule 111
rule 112
rule 113
rule 114
rule 115
rule 116
rule 117
rule 118
rule 119
rule 120
rule 121
rule 122
rule 123
rule 124
rule 125
rule 126
rule 127


rule 128
rule 129
rule 130
rule 131
rule 132
rule 133
rule 134
rule 135
rule 136
rule 137
rule 138
rule 139
rule 140
rule 141
rule 142
rule 143
rule 144
rule 145
rule 146
rule 147
rule 148
rule 149
rule 150
rule 151
rule 152
rule 153
rule 154
rule 155
rule 156
rule 157
rule 158
rule 159
rule 160
rule 161
rule 162
rule 163
rule 164
rule 165
rule 166
rule 167
rule 168
rule 169
rule 170
rule 171
rule 172
rule 173
rule 174
rule 175
rule 176
rule 177
rule 178
rule 179
rule 180
rule 181
rule 182
rule 183
rule 184
rule 185
rule 186
rule 187
rule 188
rule 189
rule 190
rule 191
rule 192
rule 193
rule 194
rule 195
rule 196
rule 197
rule 198
rule 199
rule 200
rule 201
rule 202
rule 203
rule 204
rule 205
rule 206
rule 207
rule 208
rule 209
rule 210
rule 211
rule 212
rule 213
rule 214
rule 215
rule 216
rule 217
rule 218
rule 219
rule 220
rule 221
rule 222
rule 223
rule 224
rule 225
rule 226
rule 227
rule 228
rule 229
rule 230
rule 231
rule 232
rule 233
rule 234
rule 235
rule 236
rule 237
rule 238
rule 239
rule 240
rule 241
rule 242
rule 243
rule 244
rule 245
rule 246
rule 247
rule 248
rule 249
rule 250
rule 251
rule 252
rule 253
rule 254
rule 255