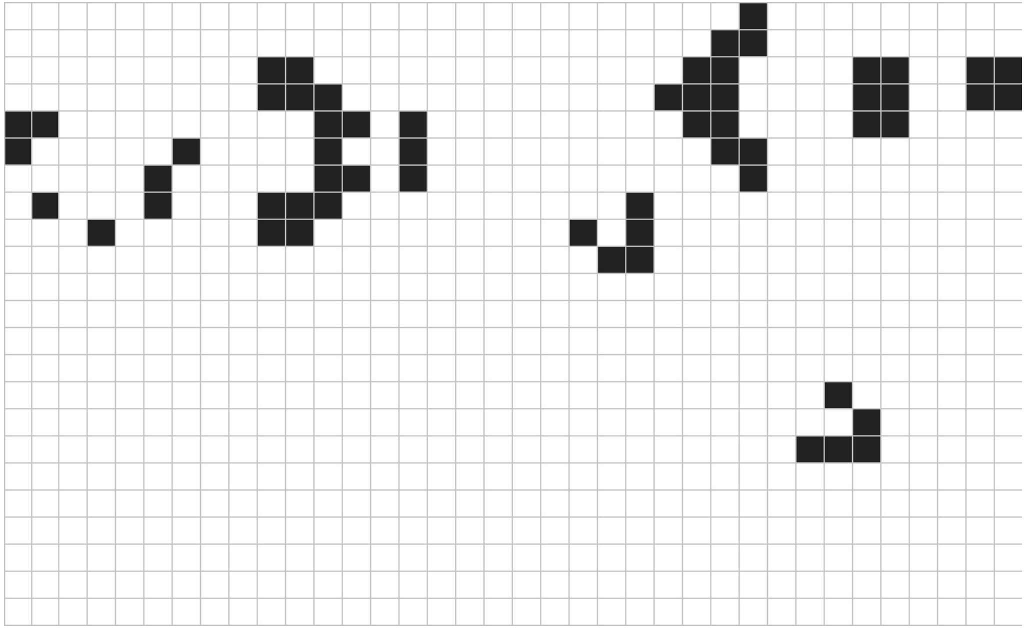
Cellular Automata

SIG Math Fall 2025

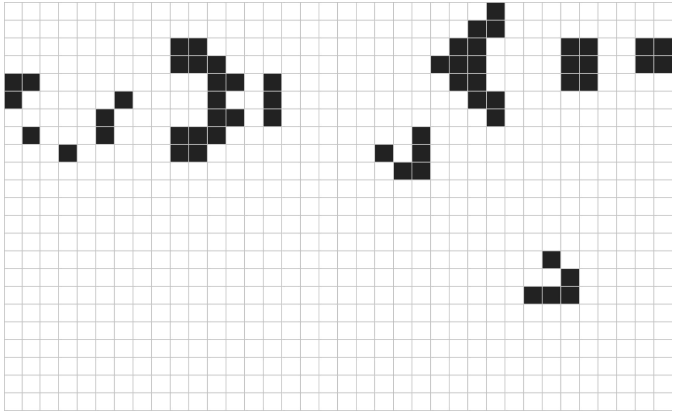
Prototypical example

- The term “cellular automata” is complex and scary sounding, but I promise most of you have seen one before…
- The Game of Life!



Traits of cellular automata

- Regular grid of cells in any finite number of dimensions
- Each cell can be in a finite number of states
- For each cell, a set of cells called its *neighborhood* is defined relative to the specified cell
- An initial state (t=0) is selected by assigning a value for each cell
- A new *generation* is created according to some fixed *rule*. All cells are updated simultaneously, and the rule does not change



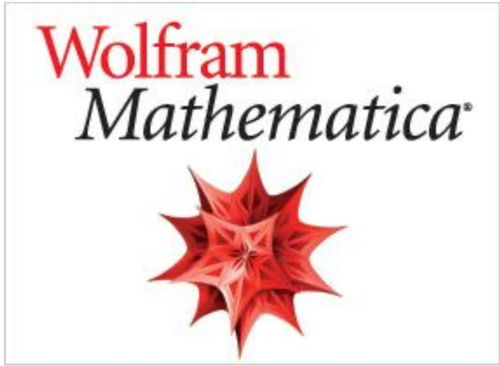
A bit of history

- The idea of CA was conceived in the 1940s by Stanisław Ulam and John von Neumann while at Los Alamos national laboratory.
- The topic was studied a bit, but interest in the topic exploded in the 1970s when Conway’s Game of Life became popular.
- In the subsequent decades, Stephen Wolfram engaged in a systematic study of 1-D cellular automata.

Elementary Cellular Automata

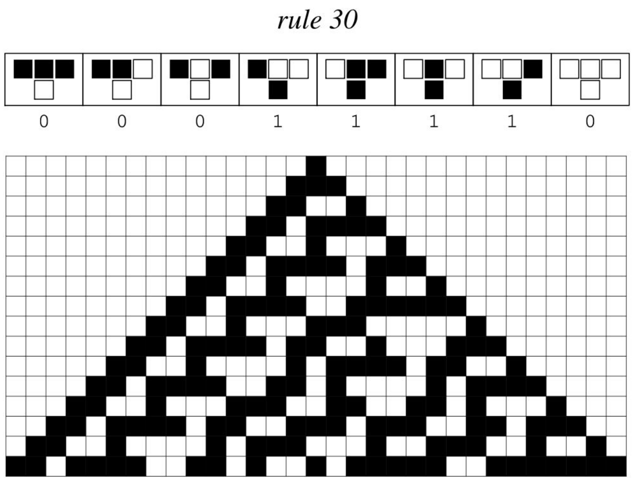
A note on Stephen Wolfram

- Stephen Wolfram is a very influential figure in the field of cellular automata and has conducted a lot of research on it over the course of several decades. He is also the creator of Wolfram Alpha and Mathematica!
- He published a famous book titled *A New Kind of Science*, though reception has been quite mixed as Wolfram is beyond egotistical and many of his claims are unsupported.
  - Wolfram believes that instead of constructing increasingly sophisticated theories to explain math and physics, we ought to focus attention on simple programs (like CA!), and instead of trying to prove theorems about them, try to learn things by looking at the output of simulations.



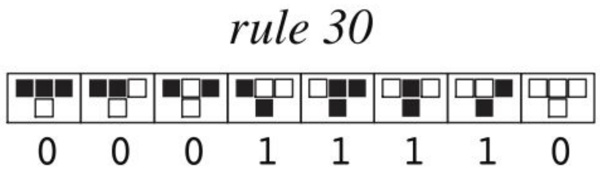
What makes elementary cellular automata elementary?

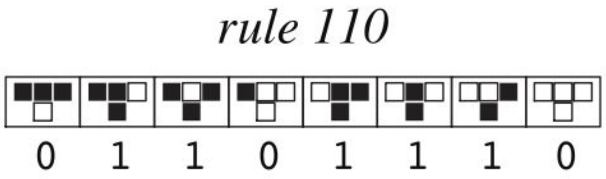
- It’s the simplest version of cellular automata that is still interesting!
- Our grid is a 1-D strip of cells, where each cell can be a 0 or a 1. The rule for each generation depends only on the cell and its immediate neighbors.
- Despite the simplicity, there is a lot of interesting behavior!

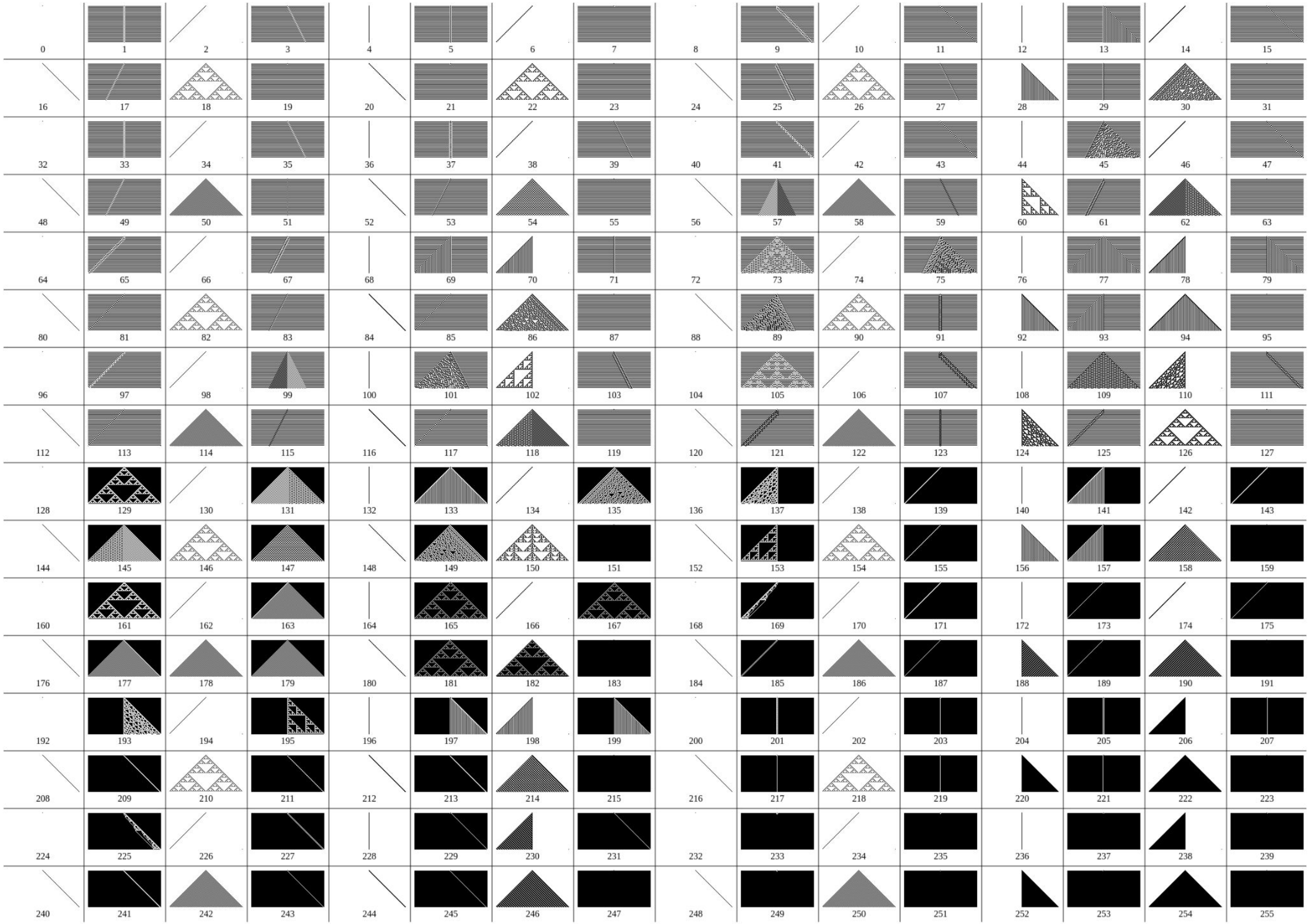


Rule numbering scheme

- For each cell and its neighbors, there are $2^3 = 8$ possible configurations
- The rule must specify the resulting state for each of those configurations, so there are 2^8 = **256 possible elementary cellular automata**.
- Stephen Wolfram proposed what is called the Wolfram code to assign each rule a number between 0 and 255.
- Each configuration is written in order: 111, 110, ... 001, 000, and the resulting state of each configuration is written in order and interpreted as the binary version of an integer.



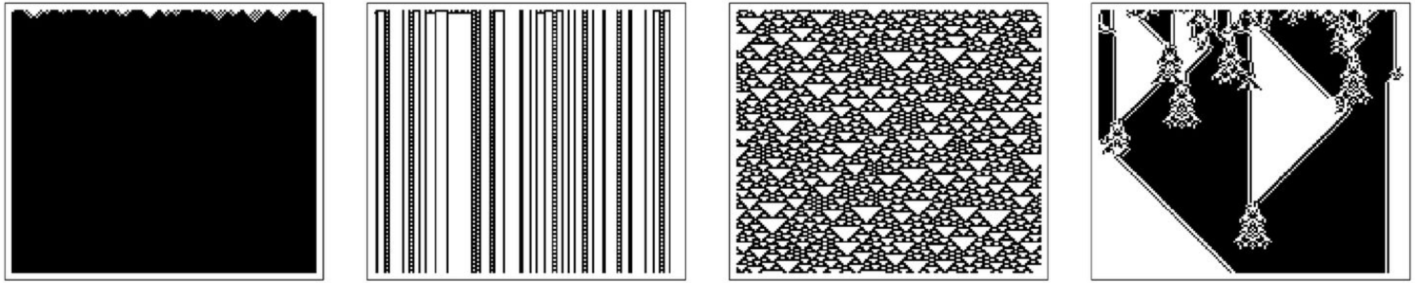




0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63
64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79
80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95
96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111
112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143
144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159
160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175
176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191
192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207
208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223
224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239
240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255

Classifying cellular automata

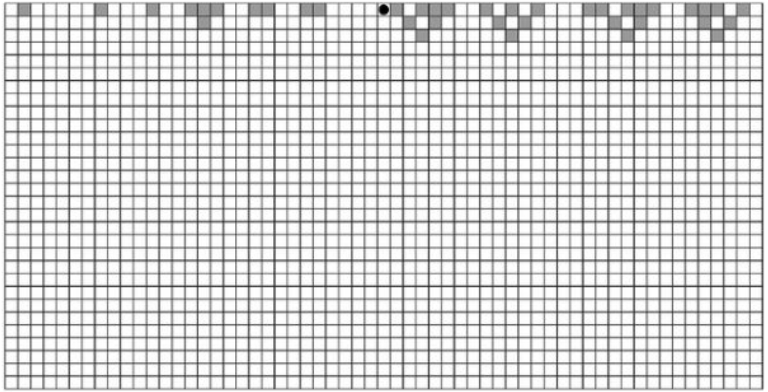
- Wolfram proposed 4 classes that all cellular automata (not just elementary) could fit into.
  - Class 1 – Reaches some fixed state and stays there
  - Class 2 – Reaches some repetitive or stable state
  - Class 3 – Remains in a random state
  - Class 4 – Forms areas of stable or repetitive states, but also forms structures that interact with each other in complicated ways



class 1 class 2 class 3 class 4

Class 1

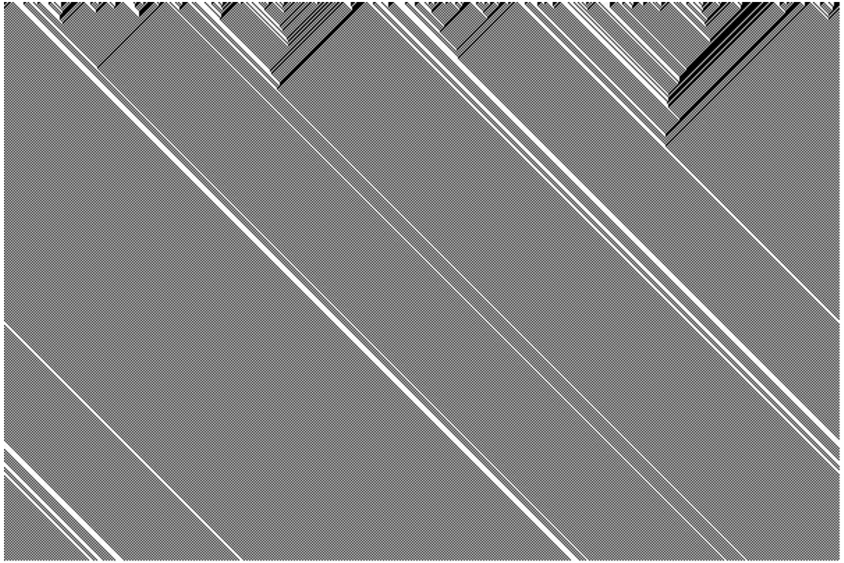
- Class 1 automata are quite uninteresting because they reach a fixed state. Any changes in the initial conditions will always die out, and the same final state is almost always reached regardless of initial conditions.
- As a result, there is nothing to study about class 1 automata (they are very boring)



rule 160

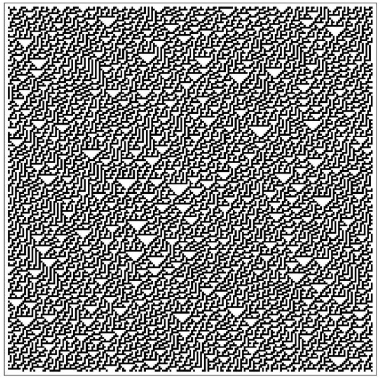
Class 2

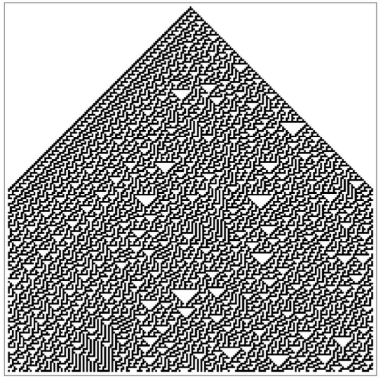
- In class 2, the initial conditions do matter somewhat, though the potential for computation is generally very limited.
- Because class 2 automata are very predictable and not random, they are generally used for real-world modeling rather than computational research.
- One example of a class 2 automaton is rule 34 which you can look up.



Class 3

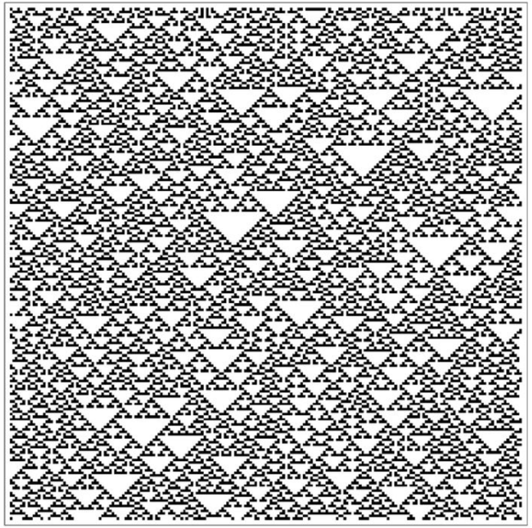
- When one looks at class 3 systems the most obvious feature of their behavior is its apparent randomness.
- What’s very interesting is that this randomness does not depend on the initial conditions – compare rule 30 with the initial conditions being random and the initial conditions being one black cell.

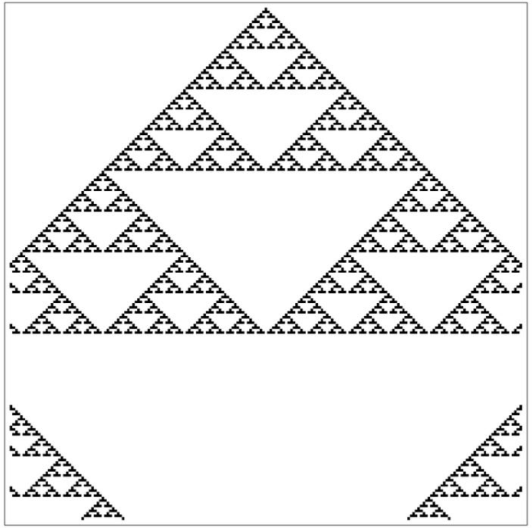




Class 3 (cont.)

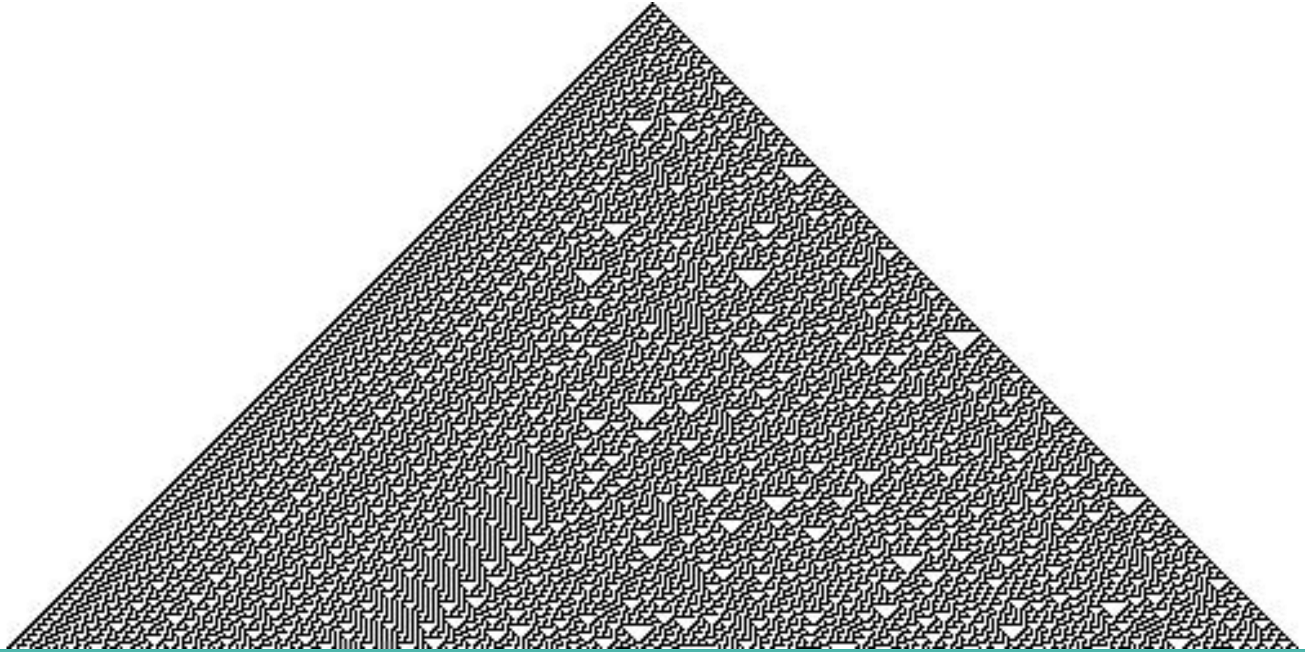
- Sometimes the randomness does come from the initial conditions. For rule 22, it looks random when starting from random initial conditions, but a Sierpinski triangle when started from one black cell:





Rule 30

- It’s hard to cover cellular automata without mentioning the arguably most significant one – rule 30!

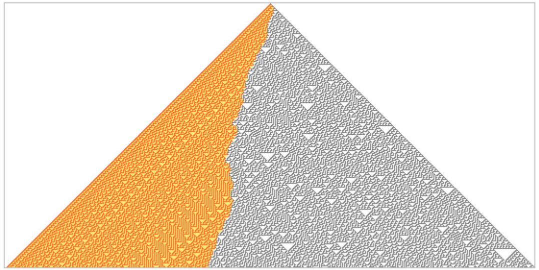


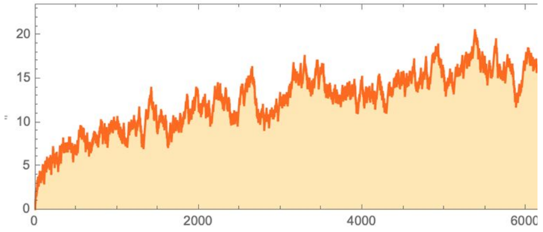
Rule 30’s center column

- There are a lot of questions we can ask about rule 30. For example, there is an obvious repetitive pattern on the left half. What is the boundary of that pattern defined by? Does it cross the middle axis? Nobody knows.
- Rule 30 is so chaotic that the **bits of the center column were used as the random number generator in Mathematica** for years!
- Stephen Wolfram uses rule 30 as the basis for his idea of computational irreducibility – a program so simple that the only way to evaluate it is to run it step by step.
- In 2019, Wolfram announced the rule 30 prizes, which ask for solutions to 3 questions relating to rule 30’s center column.

Rule 30 problem 1

- **Does the center column ever become periodic?**
- People have ran simulations for billions of iterations, and so far the center column is definitely not periodic.
  - It's also easy to prove that no *two* columns can both become periodic.
- There seems to be a boundary that separates the ordered part from the disorder. Does the ordered part always stay on the left?
- Transients do happen: consider Li(n) - pi(n). This looks like it will always stay positive, but mathematicians have proved that it becomes negative some time before 10^{317}.

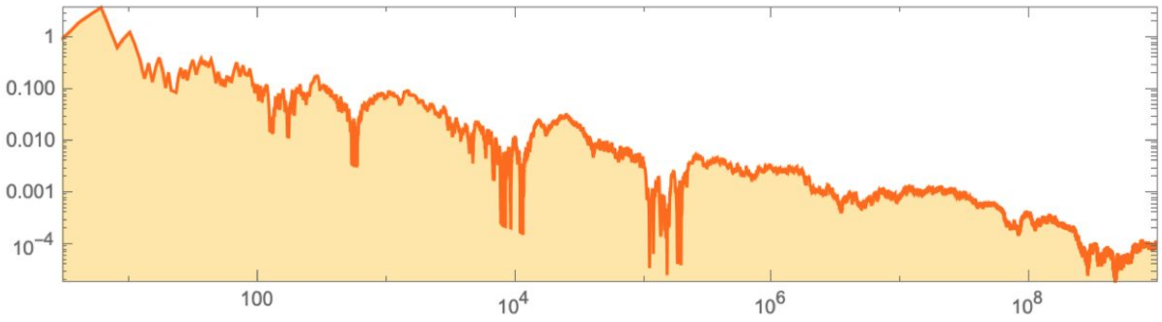




Rule 30 problem 2

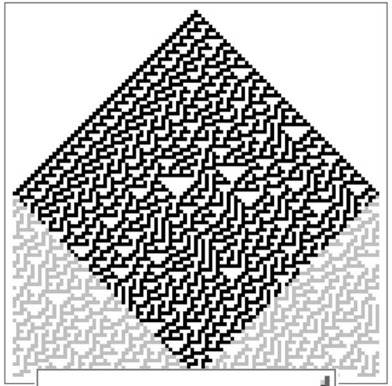
- **Does each color of cell occur on average equally often?**
- The chart displays the ratio between white and black cells in rule 30. It’s certainly very close, but does the limit actually approach 1?
- It certainly seems like the difference is getting smaller. There is a lot of empirical evidence, but unfortunately nothing concrete.

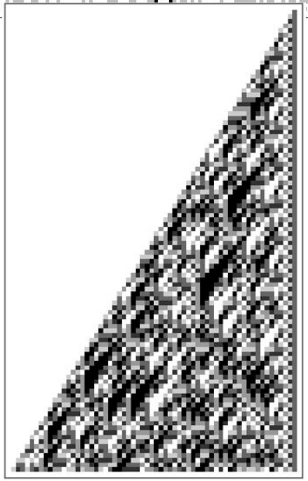
steps	■	□	ratio
1	1	0	
10	7	3	2.33333
100	52	48	1.08333
1000	481	519	0.92678
10 000	5032	4968	1.01288
100 000	50 098	49 902	1.00393
1 000 000	500 768	499 232	1.00308
10 000 000	5 002 220	4 997 780	1.00089
100 000 000	50 009 976	49 990 024	1.00040
1 000 000 000	500 025 038	499 974 962	1.00010



Rule 30 problem 3

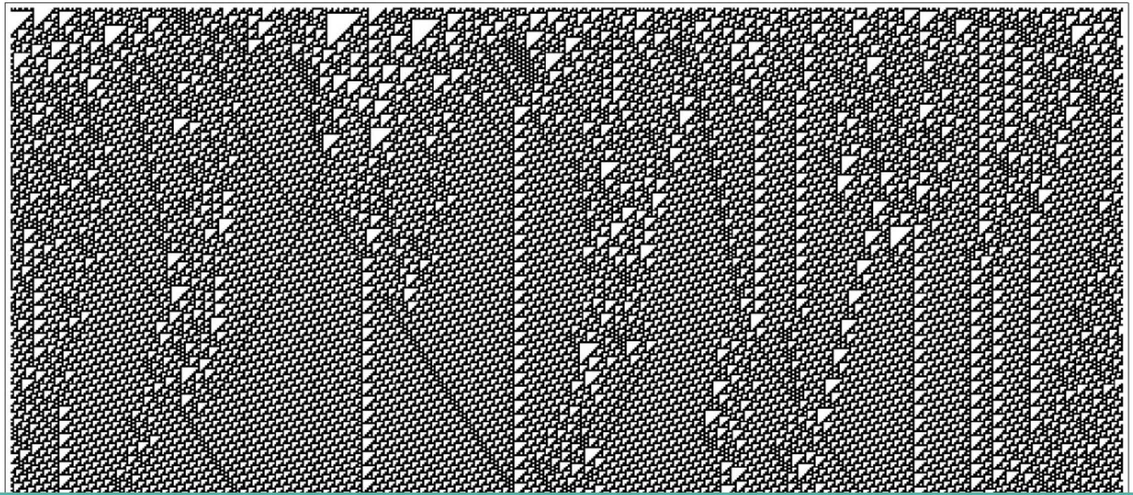
- **Does computing the nth cell require at least O(n) computational effort?**
- It's easy to find the nth cell by running rule 30 for n steps, computing the value of the center diamond. This is $O(n^2)$.
- There are problems in real life that become incomputable when tweaked slightly – compare the two-body problem with the three-body problem.
- This problem is in some sense similar to the P vs NP problem, since they both rely on proving lower bounds of computation.
- Maybe there are connections to other fields of math?





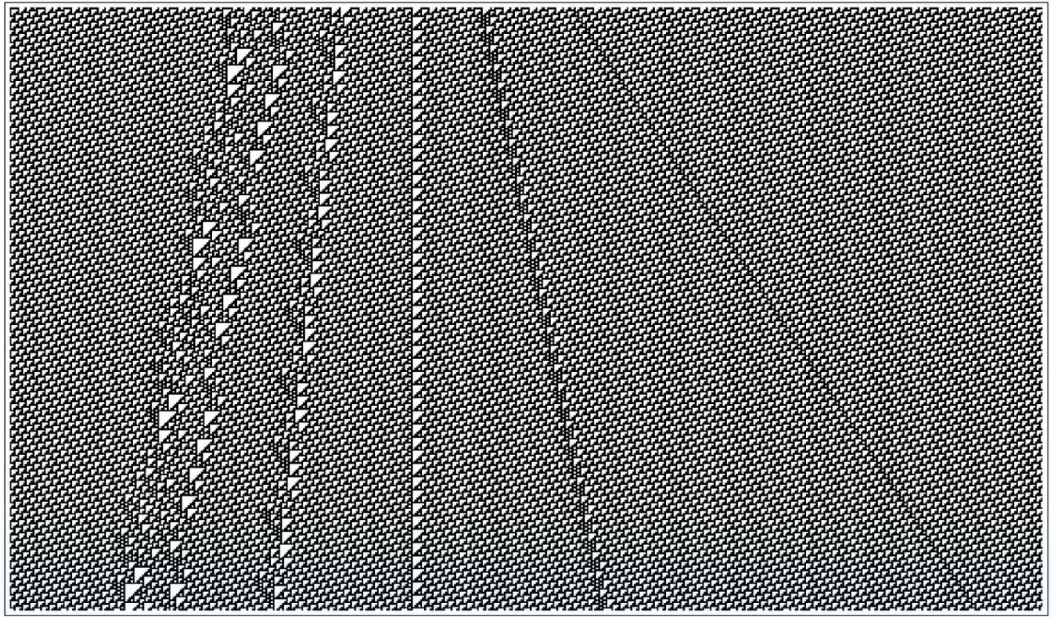
Class 4

- Class 4 systems exhibit arguably the most complex behavior. Even with random initial conditions, the systems organize themselves such that complex behavior becomes visible.
    - Sometimes these organized behaviors die out, but sometimes they persist.



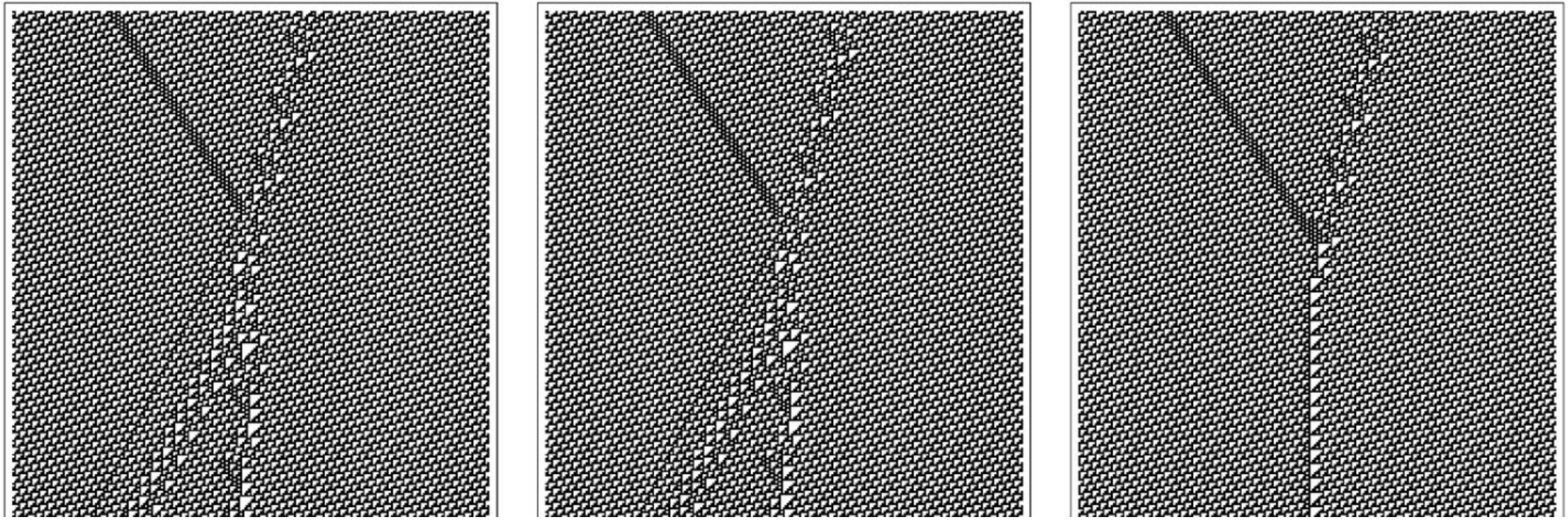
Rule 110

- The most famous example of a class 4 automaton is Rule 110. There are many types of structures, large, small, stationary, moving:



Rule 110 (cont.)

- What's fascinating is that the structures can also collide, which can lead to more complex behavior:



Rule 110 (cont.)

- These structures interact in complex ways and it's natural to ask what their nature is.
- Matthew Cook proved in 2004 that **rule 110 is Turing complete** – that is, any computer program can be run by setting the initial conditions of rule 110!
- Think about how simple our cellular automaton rule is, it's just a set of 8 rules updating a strip of cells, but it can run any program we want!
- Naturally, the question extends to: what other systems are universal?
  - We actually don't know! Not much research has been done, even on other class 4 automata. Rule 54 is a promising case, but little research has been done.

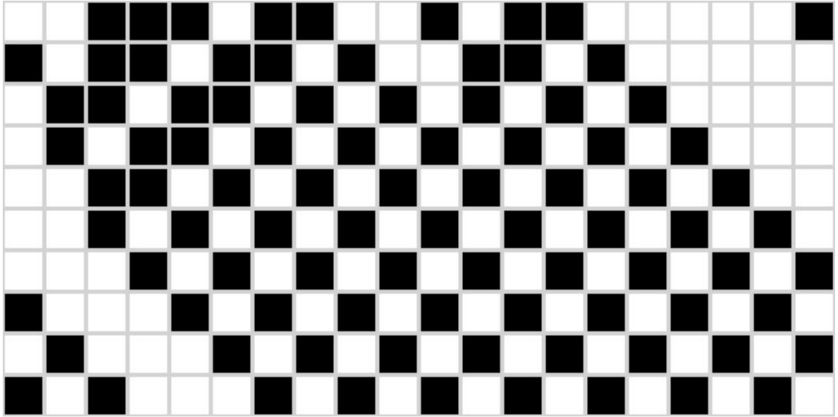
Applications of cellular automata

Elementary cellular automata

- Elementary cellular automata are very simple and as such they are more suited towards analyzing abstract models of computation than they are for practical applications.
- In particular, the one-dimensionality makes it very hard to simulate anything that happens in the real world.
- However, there is a common example of a situation where one dimensionality is the primary constraint – traffic flow!

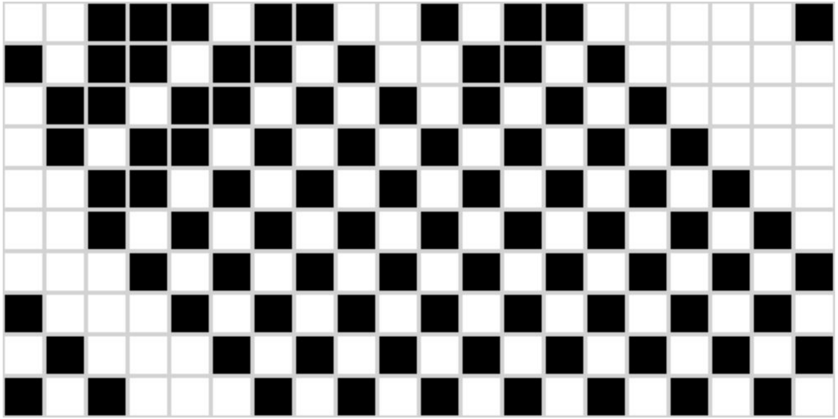
Rule 184

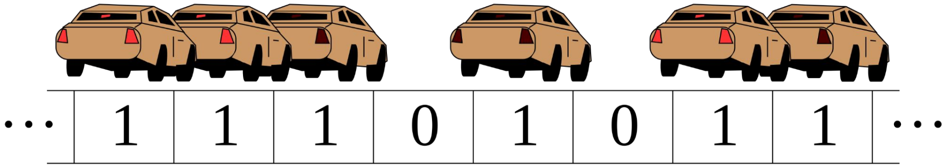
- Rule 184 is an ECA that can describe several notably different systems:
  - Rule 184 can be used as a simple model for traffic flow in a single lane of a highway, and forms the basis for many more complex models of traffic flow
  - It also models a form of deposition of particles onto an irregular surface, in which each local minimum of the surface is filled with a particle in each.
  - It can also model ballistic particle systems where colliding particles annihilate each other.



Rule 184 for traffic flow

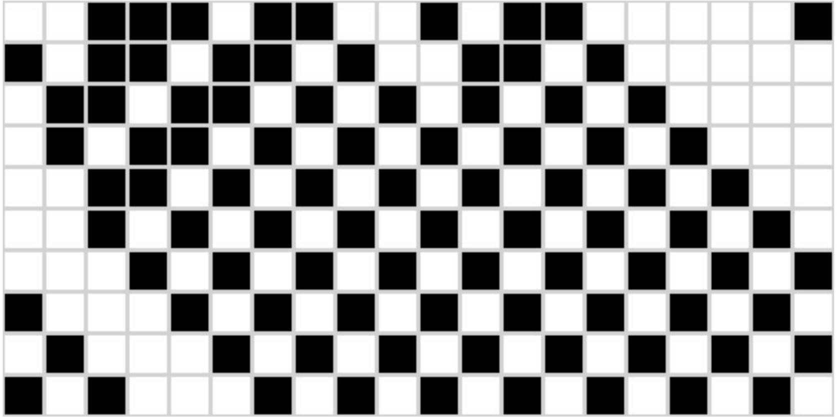
- Traffic flow – each black cell is a car. They move forwards if there is an open space ahead of them, otherwise they stop. There are emergent properties of real world traffic, such as clusters of freely moving cars when traffic is light, and stop-and-go traffic when it is heavy.

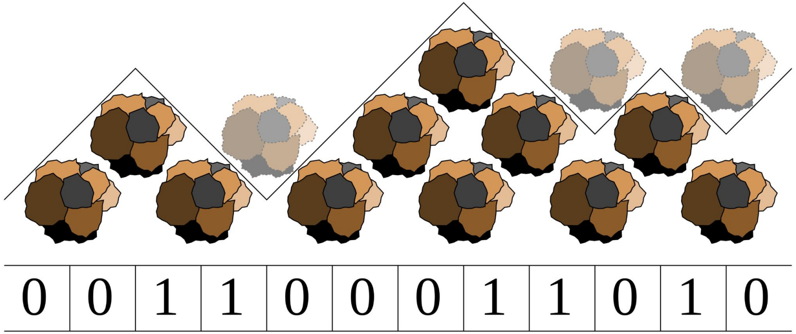




Rule 184 for surface deposition

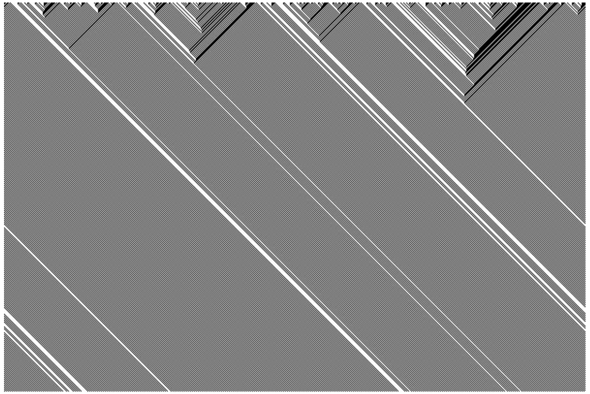
- The boundary between filled and unfilled regions is interpreted as a surface in which more particles can be deposited. At each time step, the surface grows by the deposition of new particles in each local minimum of the surface; that is, at each position where it is possible to add one new particle that has existing particles below it on both sides.

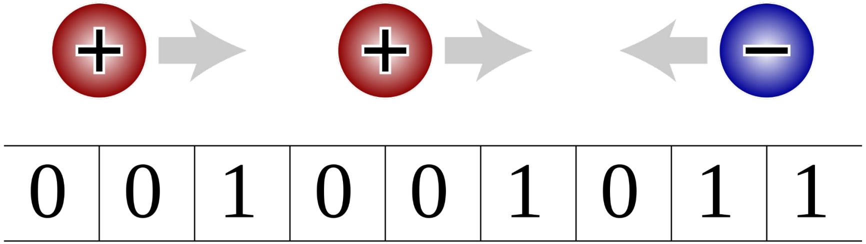




Rule 184 for ballistic annihilation

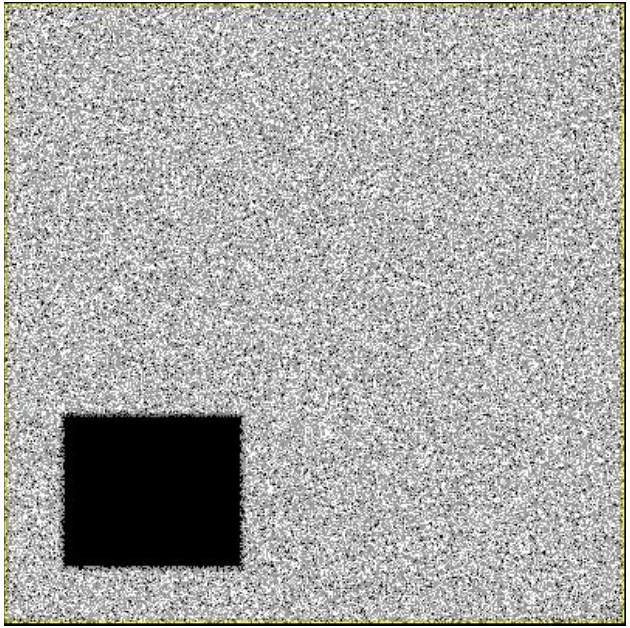
- In the simplest version of ballistic annihilation, the system consists of a single type of particle and antiparticle, moving at equal speeds in opposite directions in a one-dimensional medium.
  - Two consecutive black cells are a particle that moves rightwards
  - Two consecutive white cells are a particle that moves leftwards
  - Alternating states is the medium in which particles move





Lattice gas automata

- ECA are quite limited in their applications due to their very simple nature. However, considering more complex CA's can lead to interesting results.
- The LGA is performed on a 2D grid. Each cell has 4 bits, one per direction (up, down, left, right).
    - Each iteration, the model first performs **collision**, using conservation of momentum formulas
    - Then, the model does **propagation**, which just moves cell values based on their direction.
- LGA's are typically done on a hexagonal grid!



Lattice gas automata in practice

- This cellular automaton is still quite simple. However, it's actually **possible to use them to derive the Navier-Stokes equations**, which are the equations that model all of fluid dynamics!
- However, another model known as the lattice Boltzmann became popular in the 1990s and made LGA less relevant for simulation.
- A modified version of LGA is still used in biology! BIO-LGCA is a more complex version of LGA that simulate (literal) cells, showing bacterial colonies, tumor growth, wound healing, and more.
  - The normal technique for this is to solve very complex PDE's. Being able to simulate things can be very useful!

https://upload.wikimedia.org/wikipedia/commons/c/c0/AdhLGCA.webm

Thanks for listening!