Cellular Automaton

A cellular automaton is a collection of "colored" cells on a grid of specified shape that evolves through a number of discrete time steps according to a set of rules based on the states of neighboring cells. The rules are then applied iteratively for as many time steps as desired. von Neumann was one of the first people to consider such a model, and incorporated a cellular model into his "universal constructor." Cellular automata were studied in the early 1950s as a possible model for biological systems (Wolfram 2002, p. 48). Comprehensive studies of cellular automata have been performed by S. Wolfram starting in the 1980s, and Wolfram's fundamental research in the field culminated in the publication of his book *A New Kind of Science* (Wolfram 2002) in which Wolfram presents a gigantic collection of results concerning automata, among which are a number of groundbreaking new discoveries.

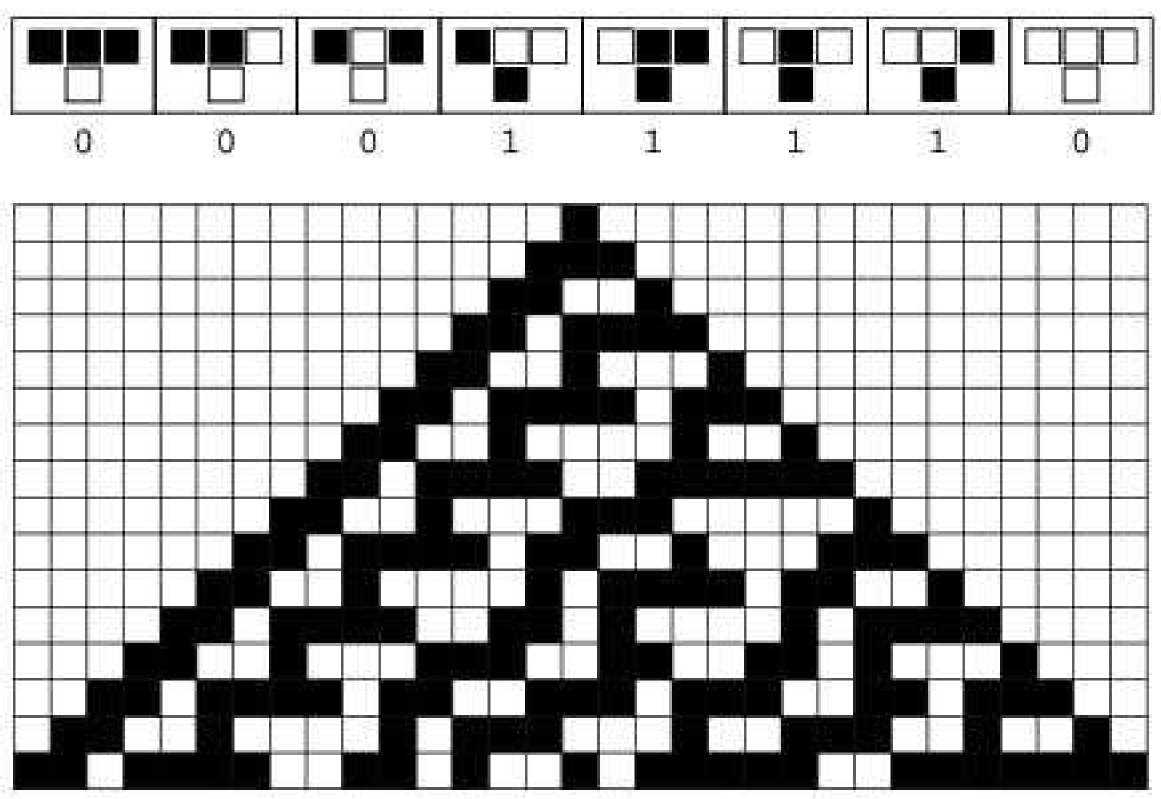
The Season 2 episode "Bettor or Worse" (2006) of the television crime drama *NUMB3RS* mentions one-dimensional cellular automata.

Cellular automata come in a variety of shapes and varieties. One of the most fundamental properties of a cellular automaton is the type of grid on which it is computed. The simplest such "grid" is a one-dimensional line. In two dimensions, square, triangular, and hexagonal grids may be considered. Cellular automata may also be constructed on Cartesian grids in arbitrary numbers of dimensions, with the d-dimensional integer lattice $\mathbb{Z}^d$ being the most common choice. Cellular automata on a d-dimensional integer lattice are implemented in the Wolfram Language as CellularAutomaton[*rule*, *init*, *steps*].

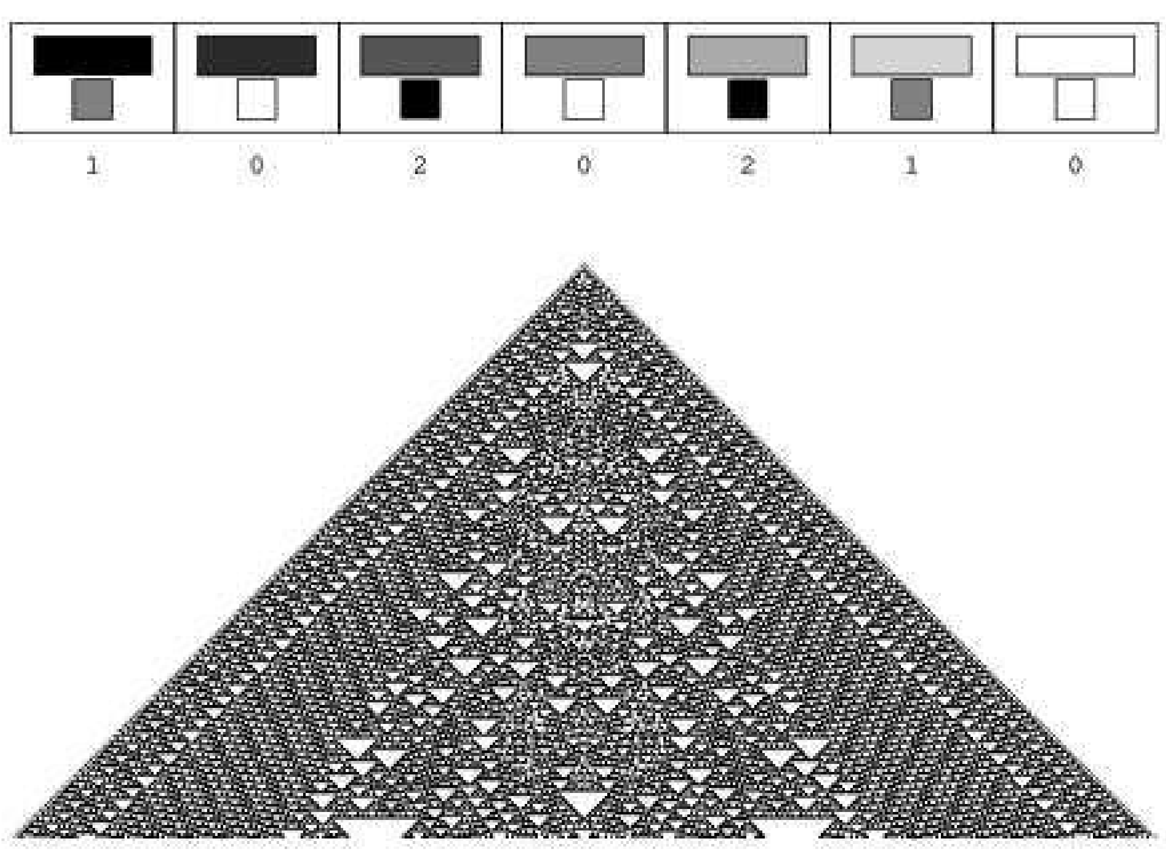
The number of colors (or distinct states) k a cellular automaton may assume must also be specified. This number is typically an integer, with $k = 2$ (binary) being the simplest choice. For a binary automaton, color 0 is commonly called "white," and color 1 is commonly called "black". However, cellular automata having a continuous range of possible values may also be considered.

In addition to the grid on which a cellular automaton lives and the colors its cells may assume, the neighborhood over which cells affect one another must also be specified. The simplest choice is "nearest neighbors," in which only cells directly adjacent to a given cell may be affected at each time step. Two common neighborhoods in the case of a two-dimensional cellular automaton on a square grid are the so-called Moore neighborhood (a square neighborhood) and the von Neumann neighborhood (a diamond-shaped neighborhood).

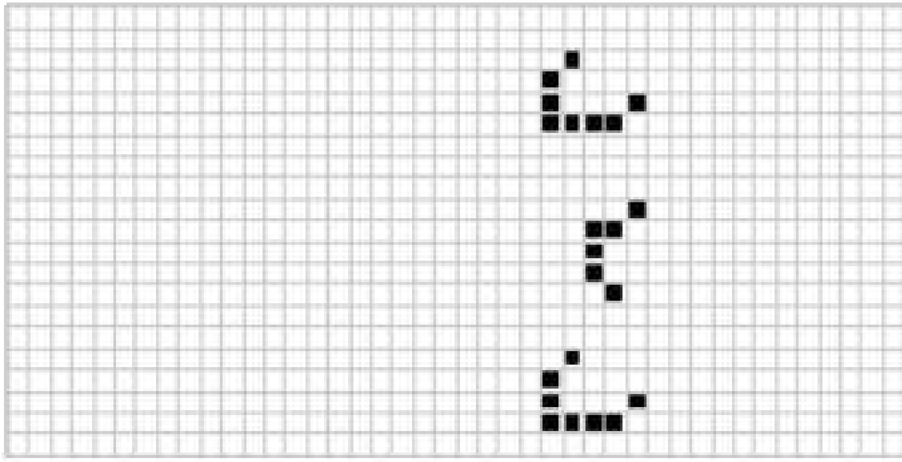
rule 30



The simplest type of cellular automaton is a binary, nearest-neighbor, one-dimensional automaton. Such automata were called "elementary cellular automata" by S. Wolfram, who has extensively studied their amazing properties (Wolfram 1983; 2002, p. 57). There are 256 such automata, each of which can be indexed by a unique binary number whose decimal representation is known as the "rule" for the particular automaton. An illustration of rule 30 is shown above together with the evolution it produces after 15 steps starting from a single black cell.



A slightly more complicated class of cellular automata are the nearest-neighbor, k-color, one-dimensional totalistic cellular automata. In such automata, it is the *average* of adjacent cells that determine the evolution, and the simplest nontrivial examples have $k = 3$ colors. For these automata, the set of rules describing the behavior can be encoded as a $(3\,k - 2)$-digit k-ary number known as a "code." The rules and 300 steps of the ternary $(k = 3)$ code 912 automaton are illustrated above.



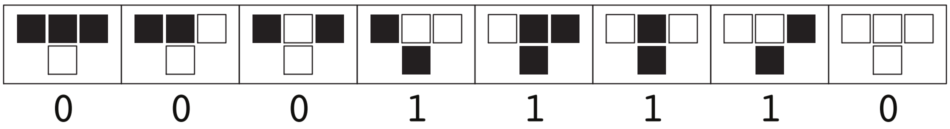
In two dimensions, the best-known cellular automaton is Conway's game of life, discovered by J. H. Conway in 1970 and popularized in Martin Gardner's *Scientific American* columns. The game of life is a binary $(k = 2)$ totalistic cellular automaton with a Moore neighborhood of range $r = 1$. Although the computation of successive game of life generations was originally done by hand, the computer revolution soon arrived and allowed more extensive patterns to be studied and propagated. An animation of the game of life construction known as a puffer train is illustrated above.

WireWorld is another common two-dimensional cellular automaton.

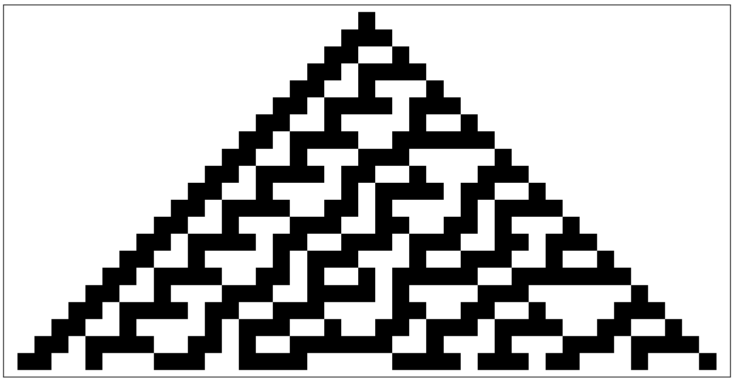
The theory of cellular automata is immensely rich, with simple rules and structures being capable of producing a great variety of unexpected behaviors. For example, there exist universal cellular automata that are capable of simulating the behavior of any other cellular automaton or Turing machine. It has even been proved by Gacs (2001) that there exist fault-tolerant universal cellular automata, whose ability to simulate other cellular automata is not hindered by random perturbations provided that such perturbations are sufficiently sparse.

Elementary Cellular Automaton

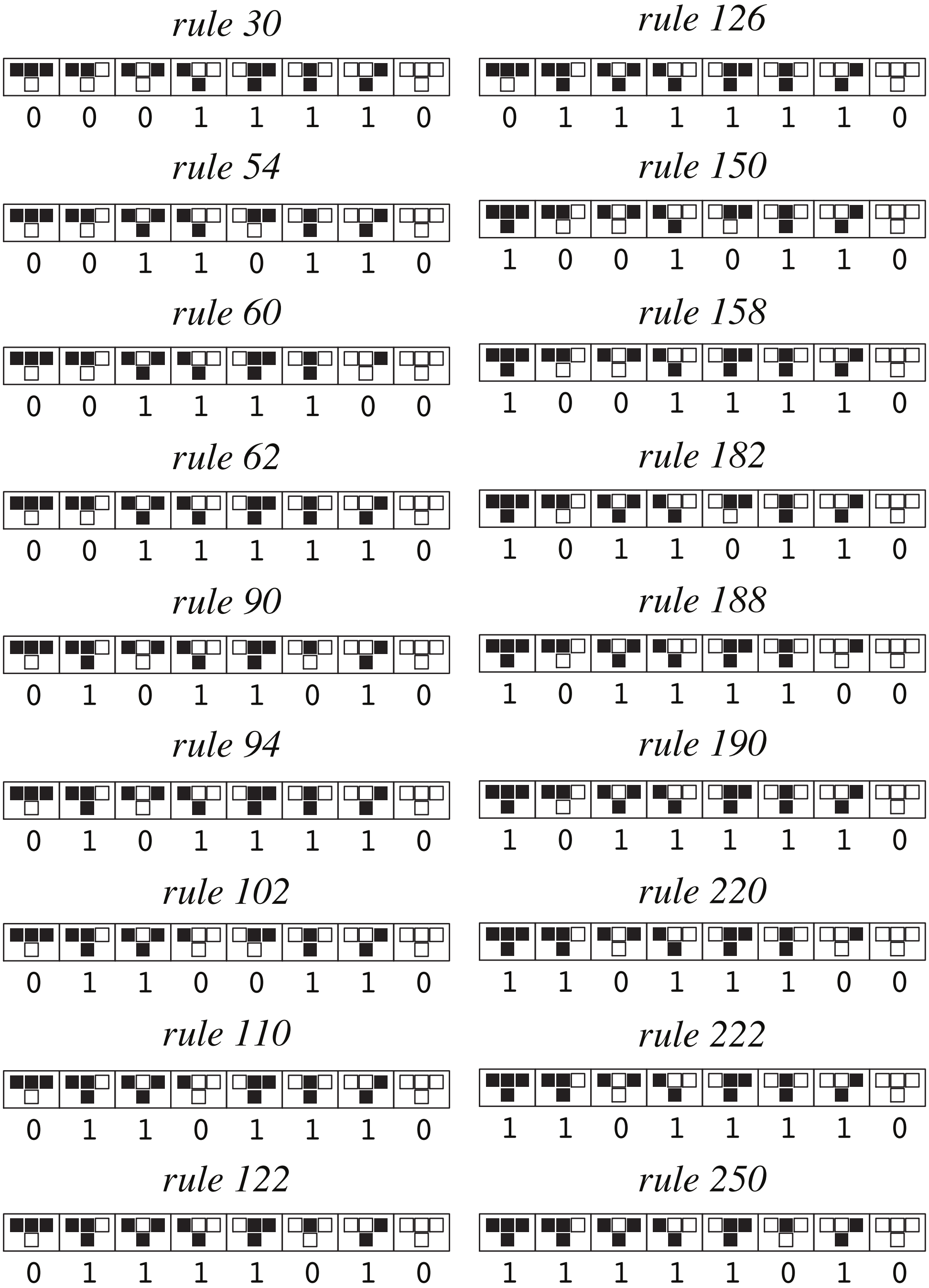
Download Wolfram Notebook

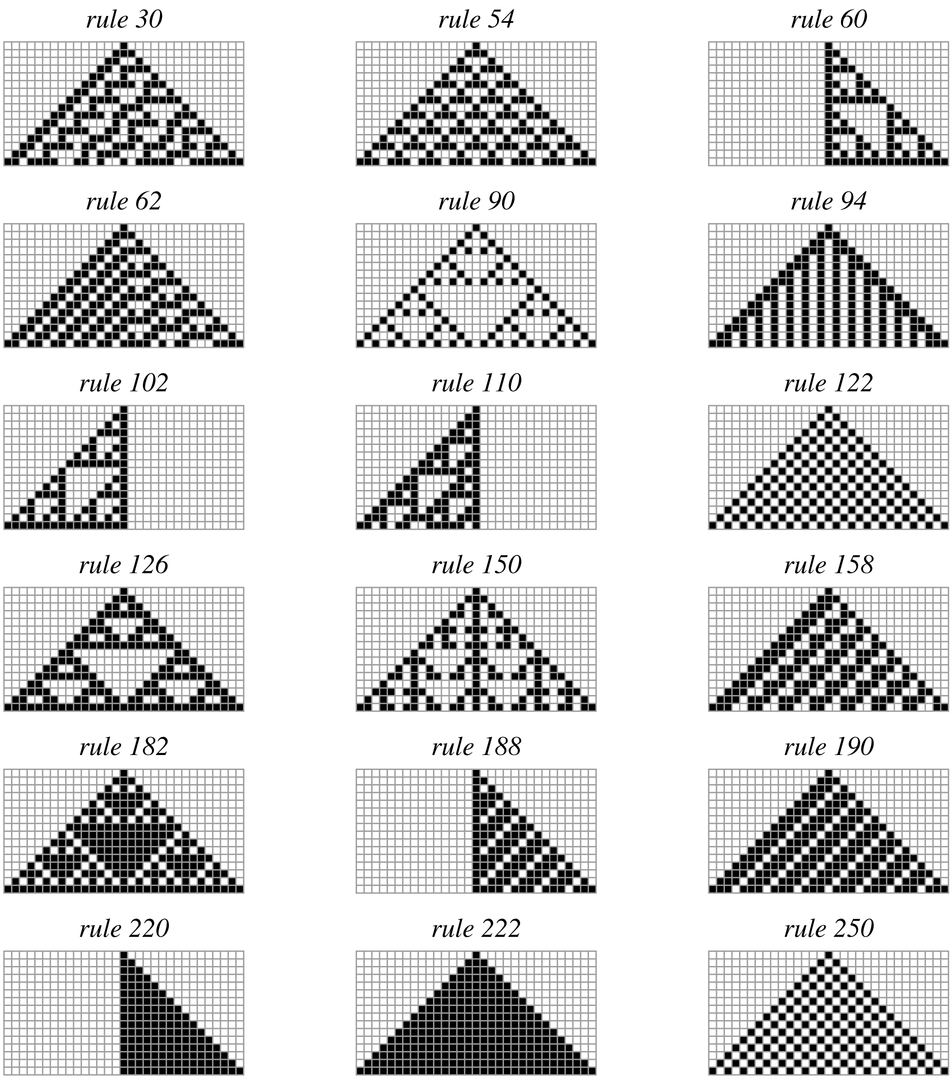


The simplest class of one-dimensional cellular automata. Elementary cellular automata have two possible values for each cell (0 or 1), and rules that depend only on nearest neighbor values. As a result, the evolution of an elementary cellular automaton can completely be described by a table specifying the state a given cell will have in the next generation based on the value of the cell to its left, the value the cell itself, and the value of the cell to its right. Since there are $2 \times 2 \times 2 = 2^3 = 8$ possible binary states for the three cells neighboring a given cell, there are a total of $2^8 = 256$ elementary cellular automata, each of which can be indexed with an 8-bit binary number (Wolfram 1983, 2002). For example, the table giving the evolution of rule 30 ($30 = 00\,011\,110_2$) is illustrated above. In this diagram, the possible values of the three neighboring cells are shown in the top row of each panel, and the resulting value the central cell takes in the next generation is shown below in the center. n generations of elementary cellular automaton rule r are implemented as CellularAutomaton[*r*, {{1}, 0}, *n*].



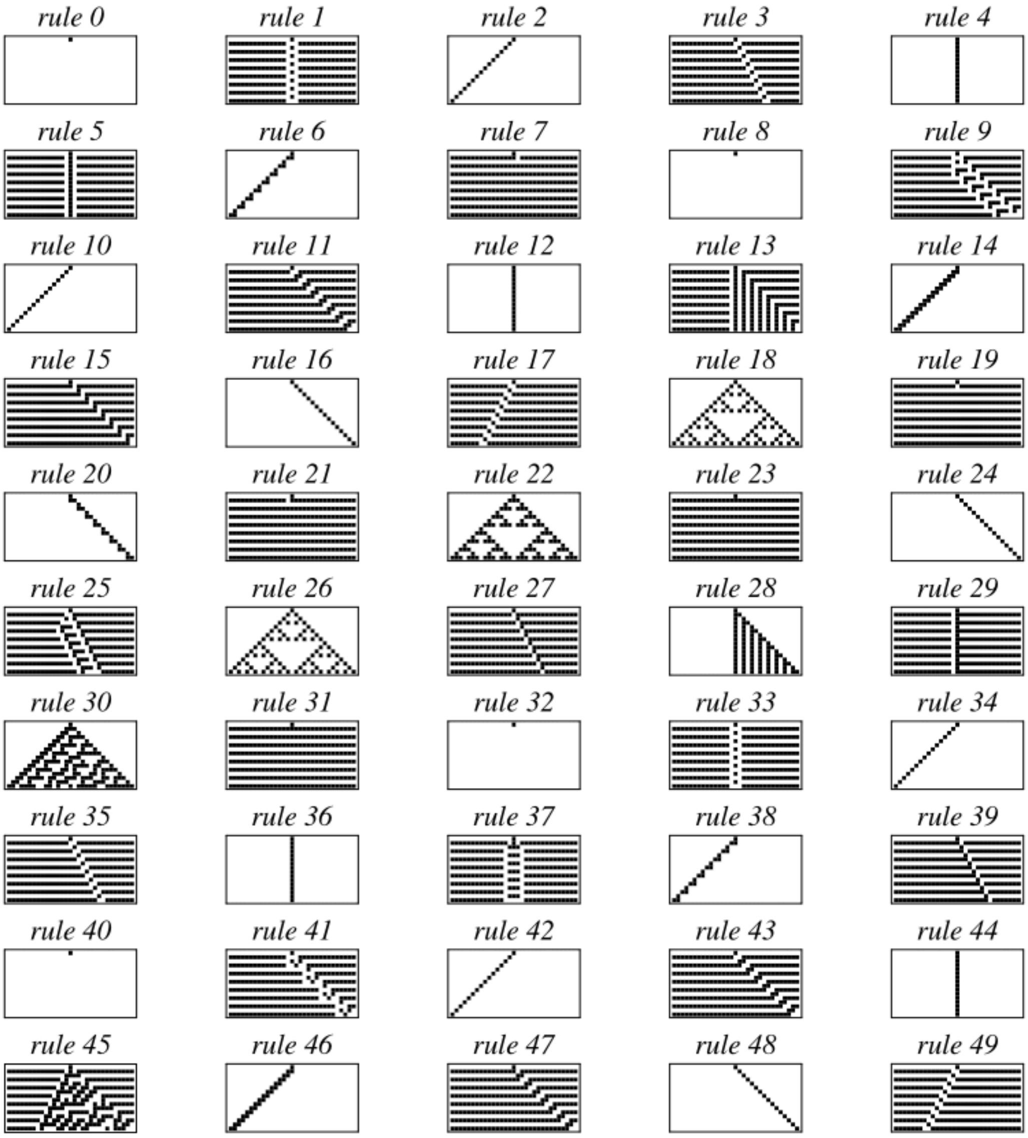
The evolution of a one-dimensional cellular automaton can be illustrated by starting with the initial state (generation zero) in the first row, the first generation on the second row, and so on. For example, the figure above illustrated the first 20 generations of the rule 30 elementary cellular automaton starting with a single black cell.

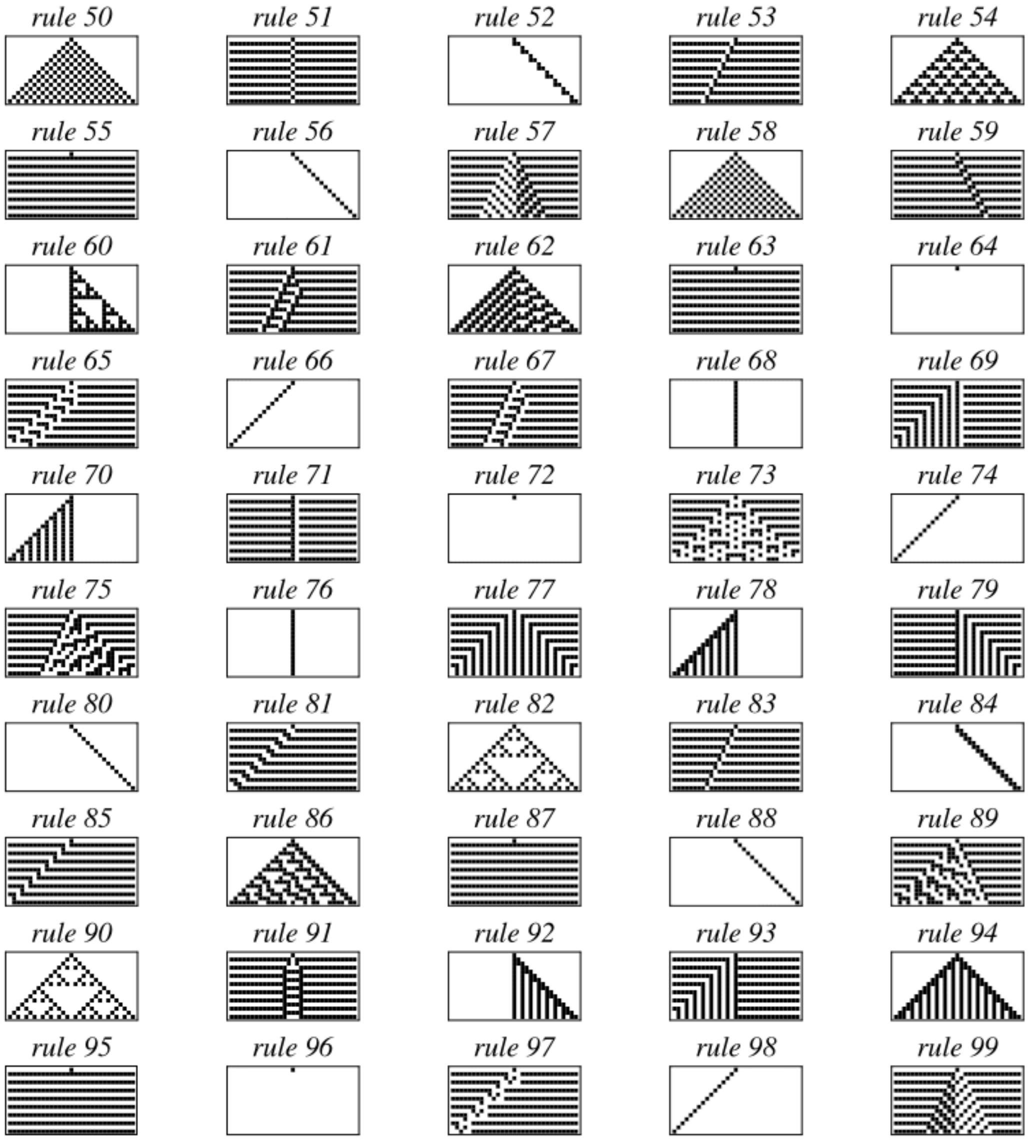

rule 30
0 0 0 1 1 1 1 0
rule 126
0 1 1 1 1 1 1 0
rule 54
0 0 1 1 0 1 1 0
rule 150
1 0 0 1 0 1 1 0
rule 60
0 0 1 1 1 1 0 0
rule 158
1 0 0 1 1 1 1 0
rule 62
0 0 1 1 1 1 1 0
rule 182
1 0 1 1 0 1 1 0
rule 90
0 1 0 1 1 0 1 0
rule 188
1 0 1 1 1 1 0 0
rule 94
0 1 0 1 1 1 1 0
rule 190
1 0 1 1 1 1 1 0
rule 102
0 1 1 0 0 1 1 0
rule 220
1 1 0 1 1 1 0 0
rule 110
0 1 1 0 1 1 1 0
rule 222
1 1 0 1 1 1 1 0
rule 122
0 1 1 1 1 0 1 0
rule 250
1 1 1 1 1 0 1 0

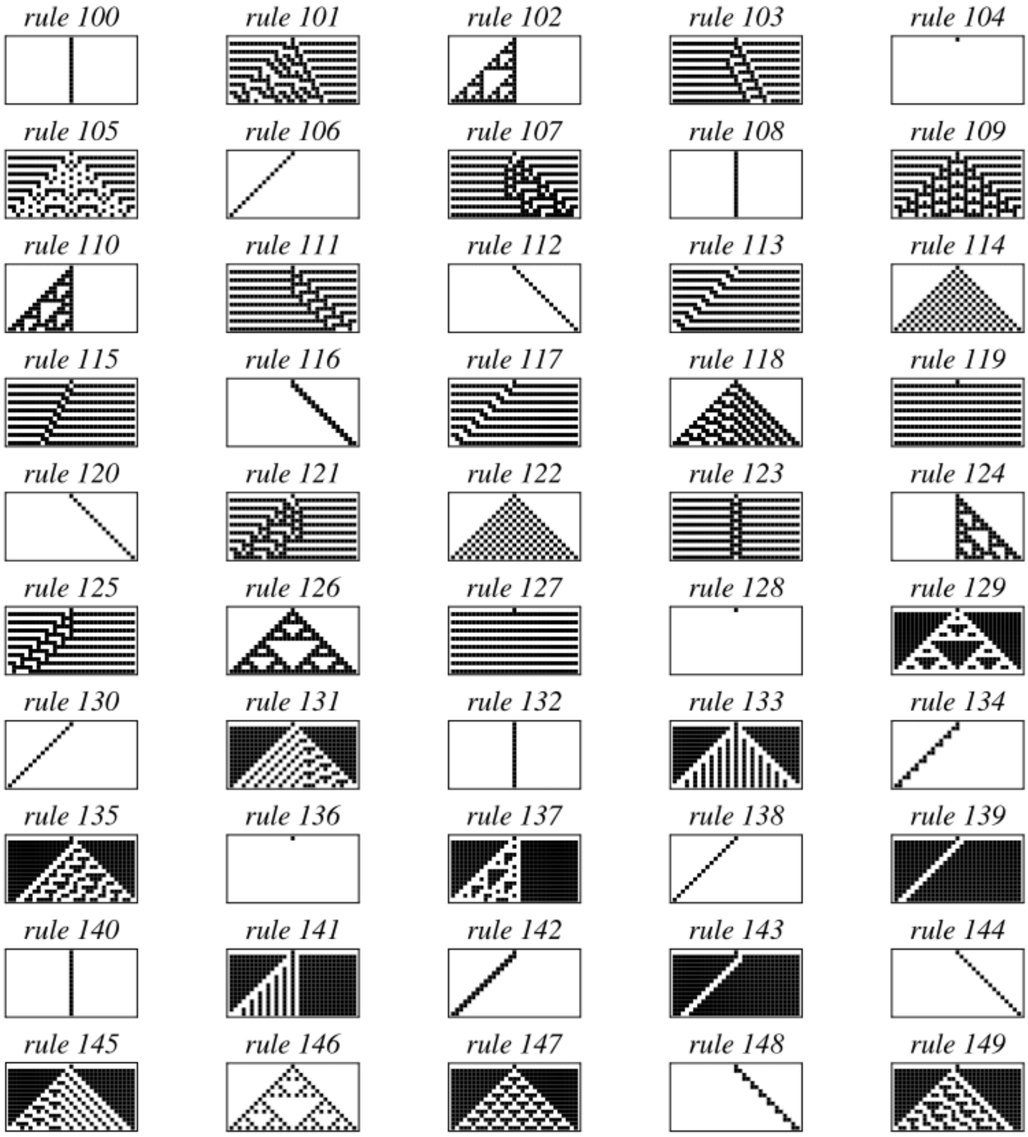


The illustrations above show some automata numbers that give particularly interesting pattern propagated for 15 generations starting with a single black cell in the initial iteration. Rule 30 is of special interest because it is chaotic (Wolfram 2002, p. 871), with central column given by 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, ... (OEIS A051023). In fact, this rule is used as the random number generator used for large integers in the Wolfram Language (Wolfram 2002, p. 317).

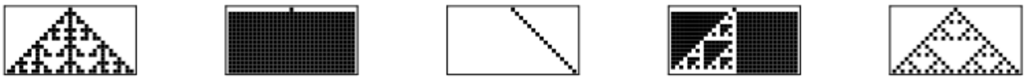
The complete set of 256 (rules 0-255) elementary cellular automata are illustrated below for a starting condition consisting of a single black cell.


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

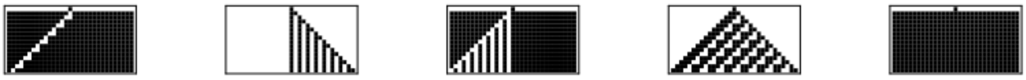

rule 50
rule 51
rule 52
rule 53
rule 54
rule 55
rule 56
rule 57
rule 58
rule 59
rule 60
rule 61
rule 62
rule 63
rule 64
rule 65
rule 66
rule 67
rule 68
rule 69
rule 70
rule 71
rule 72
rule 73
rule 74
rule 75
rule 76
rule 77
rule 78
rule 79
rule 80
rule 81
rule 82
rule 83
rule 84
rule 85
rule 86
rule 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

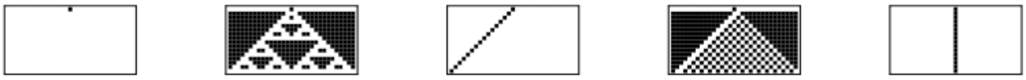
rule 150 *rule 151* *rule 152* *rule 153* *rule 154*



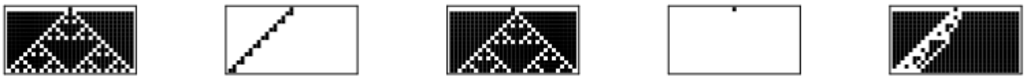
rule 155 *rule 156* *rule 157* *rule 158* *rule 159*



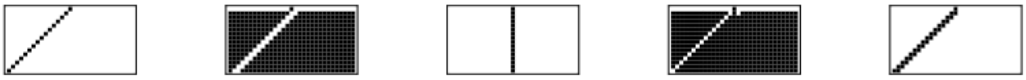
rule 160 *rule 161* *rule 162* *rule 163* *rule 164*



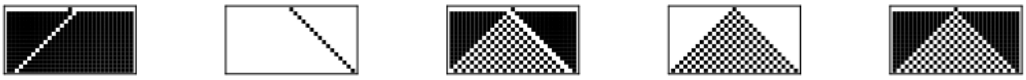
rule 165 *rule 166* *rule 167* *rule 168* *rule 169*



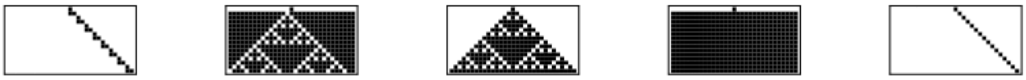
rule 170 *rule 171* *rule 172* *rule 173* *rule 174*



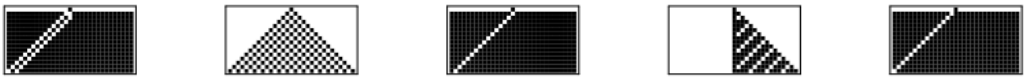
rule 175 *rule 176* *rule 177* *rule 178* *rule 179*



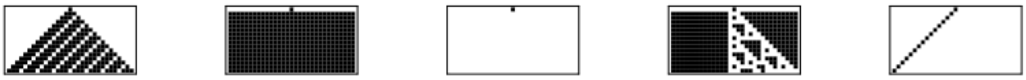
rule 180 *rule 181* *rule 182* *rule 183* *rule 184*



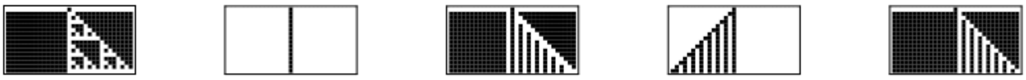
rule 185 *rule 186* *rule 187* *rule 188* *rule 189*

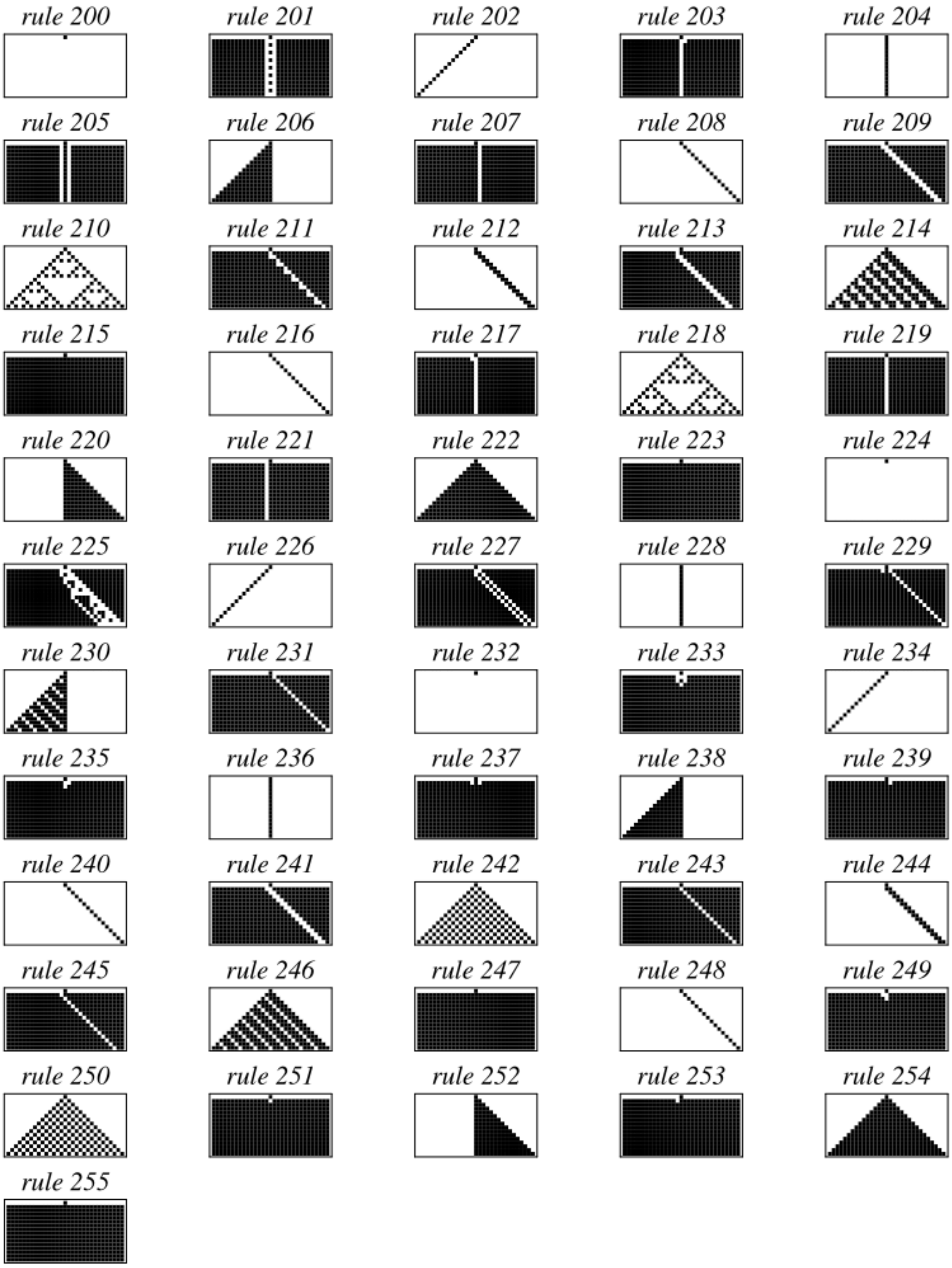


rule 190 *rule 191* *rule 192* *rule 193* *rule 194*



rule 195 *rule 196* *rule 197* *rule 198* *rule 199*





Letting $a_i(t)$ denote the state of the ith cell (for i running from $-\infty$ to ∞) at time $t=0$, 1, ..., its value can be written explicitly in terms of the adjacent cells from the previous generation as a trivariate function

$$a_i(t) = f(a_{i-1}(t-1), a_i(t-1), a_{i+1}(t-1)). \tag{1}$$

If the values $a_i(t)$ are represented by Boolean values, then the functions may have particularly simple forms for certain rules. In particular,

$$f_{30}(p, q, r) = \mathrm{Xor}[p, \mathrm{Or}[q, r]] \tag{2}$$
$$f_{90}(p, q, r) = \mathrm{Xor}[p, r] \tag{3}$$
$$f_{110}(p, q, r) = \mathrm{Xor}[\mathrm{Or}[p, q], \mathrm{And}[p, q, r]] \tag{4}$$
$$f_{250}(p, q, r) = \mathrm{Or}[p, r] \tag{5}$$
$$f_{254}(p, q, r) = \mathrm{Or}[p, q, r] \tag{6}$$

(Wolfram 2002, p. 869).

Of the $2^8 = 256$ elementary cellular automata, there are 88 fundamentally inequivalent rules (Wolfram 2002, p. 57).

The amphichiral elementary cellular automata are 0, 1, 4, 5, 18, 19, 22, 23, 32, 33, 36, 37, 50, 51, 54, 55, 72, 73, 76, 77, 90, 91, 94, 95, 104, 105, 108, 109, 122, 123, 126, 127, 128, 129, 132, 133, 146, 147, 150, 151, 160, 161, 164, 165, 178, 179, 182, 183, 200, 201, 204, 205, 218, 219, 222, 223, 232, 233, 236, 237, 250, 251, 254, and 255.