

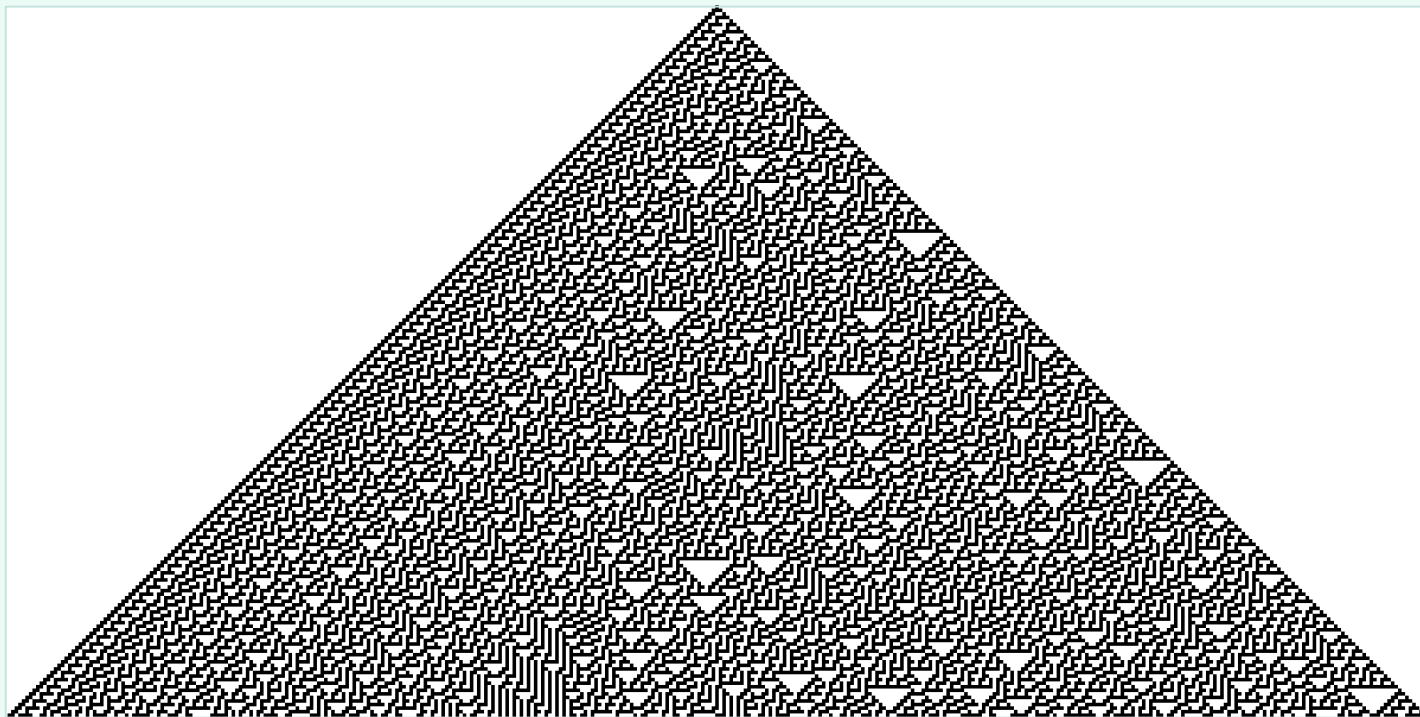
Classic Algorithms in Computer Science: 1D Cellular Automata

Intricate Patterns in Nature



<https://plus.maths.org/cellular-automata>

What simple rule made this?



Each row is computed from the row above it — nothing else

One row, three neighbours

time t



time $t + 1$



Cells

A line of squares, each 0 (off) or 1 (on)

Neighbourhood

A cell, plus the cell to its left and right

Rule

The same lookup table applies to every cell, all at once

Eight patterns make one rule number

							
							
0	0	0	1	1	1	1	0
128	64	32	16	8	4	2	1

00011110 in binary = $16 + 8 + 4 + 2 =$ **Rule 30**

Two outputs for each of 8 patterns gives $2^8 = 256$ possible rules, numbered 0 to 255

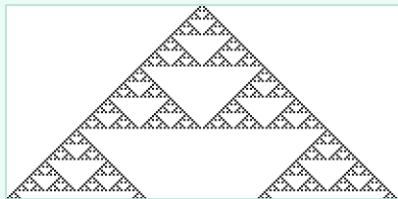
Your turn: work out Rule 90

							
							
?	?	?	?	?	?	?	?
128	64	32	16	8	4	2	1

1. Write 90 in binary (8 bits)
2. Fill in the eight outputs
3. Start from one black cell and draw five rows

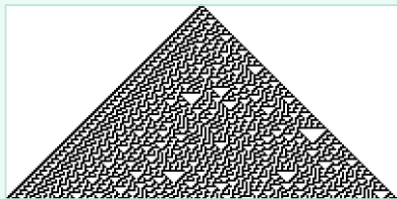
Bonus: can you describe the rule in one sentence, without the table?

Four rules, four personalities



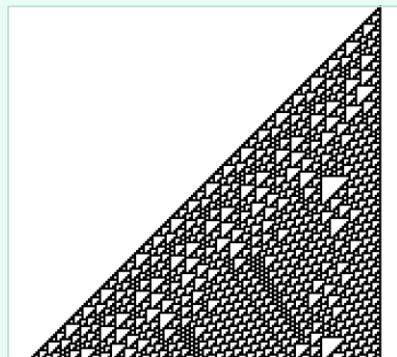
Rule 90

A perfect fractal: the Sierpiński triangle



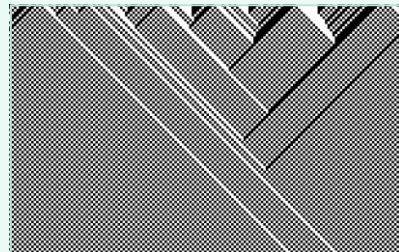
Rule 30

Looks random; Wolfram used it to generate random numbers



Rule 110

Proven capable of universal computation



Rule 184

Cars on a road: traffic jams form and dissolve

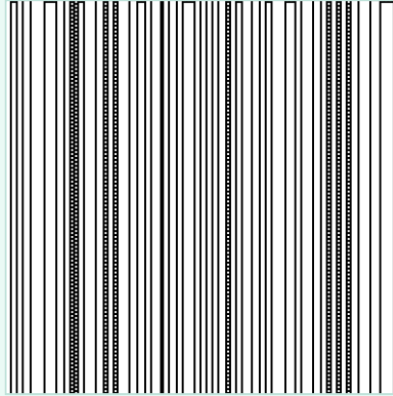
Wolfram's four classes



I • Uniform

Every cell ends up in the same state

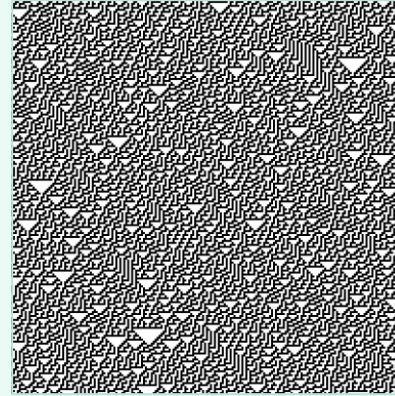
e.g. Rule 254



II • Periodic

Settles into stable or repeating stripes

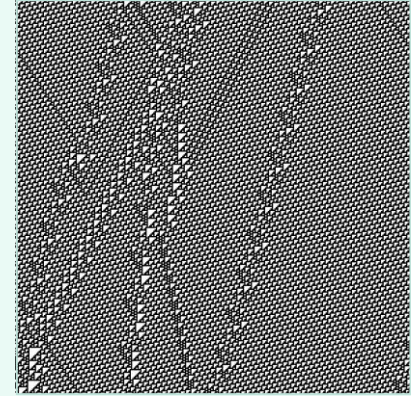
e.g. Rule 108



III • Chaotic

Random-looking forever

e.g. Rule 30



IV • Complex

Structures that move and interact

e.g. Rule 110

All four start from random rows; Class IV is shown after 300 steps

From toy model to living skin

Are cellular automata only abstract mathematics?



A lizard whose scales compute



Ocellated lizard: each scale is a single colour

The animal

Ocellated lizard (*Timon lepidus*), whose back has a green-and-black maze pattern

The finding

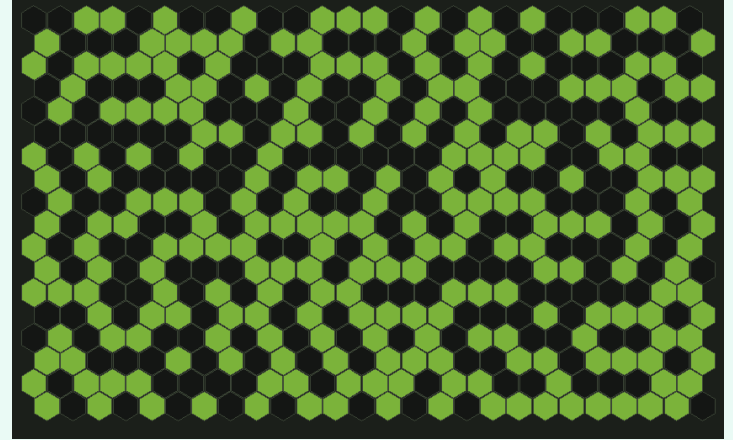
Each scale's colour is set by its neighbours, so the scale lattice acts as a (stochastic) cellular automaton

The evidence

Scale colours were tracked as the lizards grew, over about four years

Manukyan et al., Nature 544, 173–179 (2017)

Smooth chemistry, discrete scales



Turing reaction-diffusion: a continuous pattern

Scale lattice: one colour per scale

Skin is thinner at scale borders. That alone splits the smooth pattern scale by scale, turning it into a cellular automaton.

A computer lizard that regrows itself

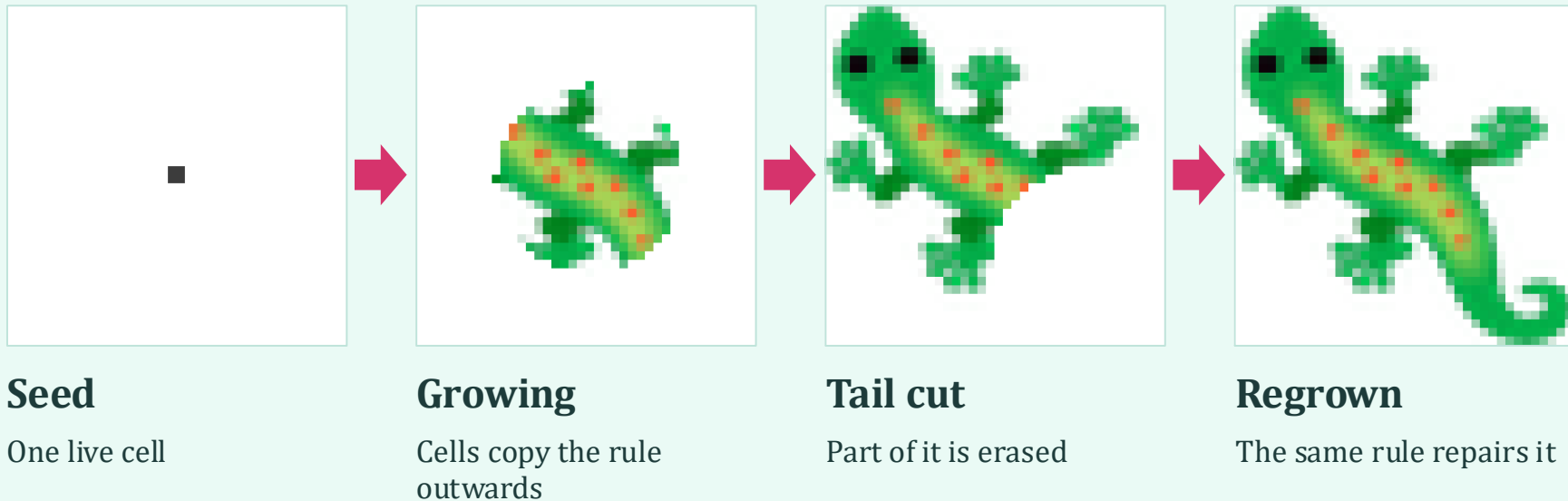


Illustration of the sequence, drawn from the pixel-art lizard emoji the researchers trained on

Mordvintsev, Randazzo, Niklasson & Levin, "Growing Neural Cellular Automata", Distill (2020) · try it live at distill.pub/2020/growing-ca

Wolfram's question, run in reverse

Wolfram (today's lesson)

Write the rule → see what pattern appears

Neural cellular automata

Choose the pattern → learn a rule that builds it

How it works

- Cell states are smooth numbers from 0 to 1, not just on or off
- Each cell also carries hidden values that guide growth
- The rule is a neural network of about 8,000 parameters
- Cells update at random moments, not all at once

Where it's going

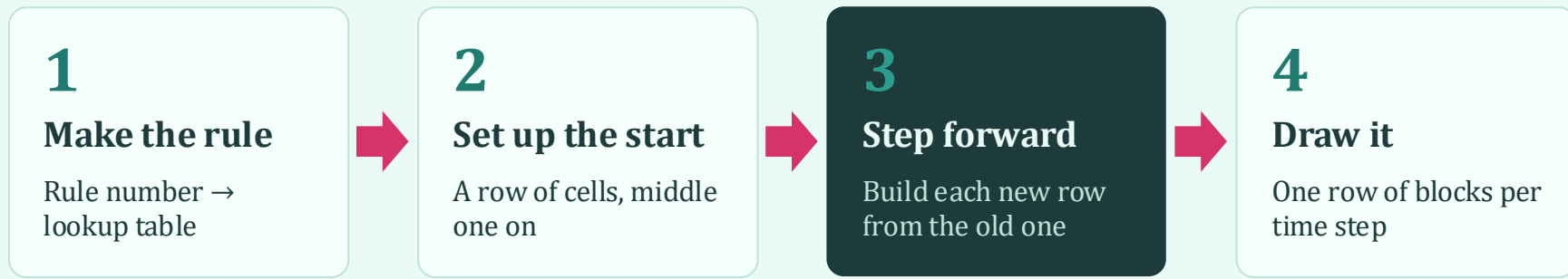
- Clues for regenerating limbs
- Energy-efficient computing with only local connections
- Robot swarms that act as one body
- Caveat: evolution has no target shape to train towards

Source: G. Musser, "Self-Assembly Gets Automated in Reverse of 'Game of Life'", Quanta Magazine, 10 Sept 2025

Implementing 1D Cellular Automata

The steps we need to think about before writing any code

The plan: split it into four jobs



Jobs 3 and 4 repeat once per time step

A main program ties the jobs together

- It decides three settings: the rule number, the width (space) and the number of steps (time)
- Then it runs the jobs in order, so each job can be written and tested on its own

Job 1: turn the rule number into a lookup table

1

Turn the rule number into 8 binary digits

Add zeros at the front if it is shorter than 8 digits

2

List the 8 possible inputs in Wolfram's order

Every neighbourhood, from 111 down to 000

3

Pair each input with its output digit

First digit with 111, second with 110, and so on

The table maps every possible input to an output. Mathematicians call this a Boolean function.

Worked example: Rule 30

30 in binary is 11110

Padded to 8 digits: 00011110

Input → output

111 → 0 011 → 1

110 → 0 010 → 1

101 → 0 001 → 1

100 → 1 000 → 0

Jobs 2 and 3: set up, then step forward

1

Set up the first row

Choose a width; every cell off except the middle one

2

For each time step, start a fresh empty row

The old row stays untouched while we read it

3

For every cell, read left, centre and right

Taken from the old row; the ends wrap round (see right)

4

Look up the pattern and write the answer

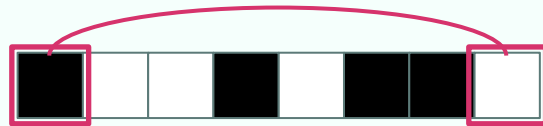
The output goes into the same position in the new row

5

Only when the row is finished, swap

The new row becomes the current row; then draw it

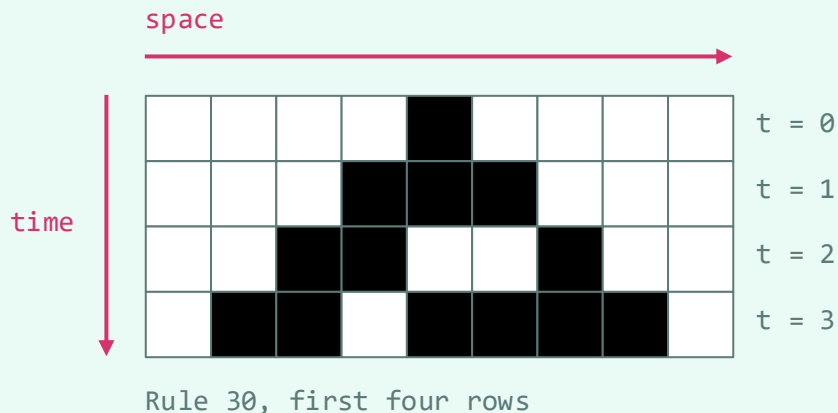
Two key ideas



Wrap-around: the first and last cells are neighbours, so the row behaves like a ring

Parallel update: every cell changes at the same moment, so all of them must read the same old row

Job 4: draw space and time



- Each cell becomes a block: black if on, white if off
- Make each block two characters wide so the cells look square
- Print every new row under the last one, labelled with its time step
- The finished picture is a space-time diagram

Then let the user explore

Show Rule 30 first, then offer a choice: watch all 256 rules one after another, with a short pause between each, or see one chosen at random