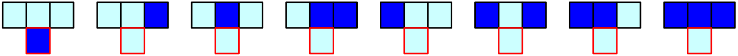
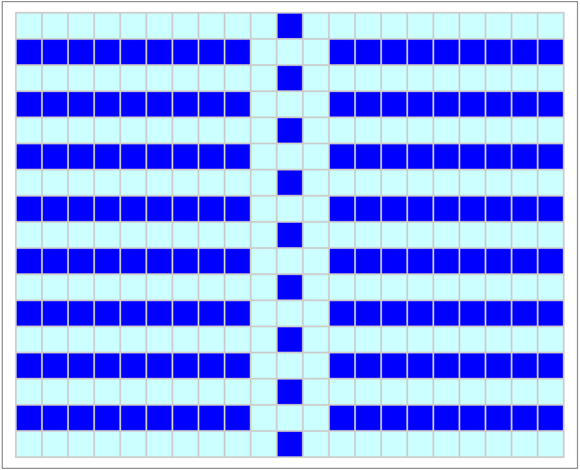
Cellular Automata and Universality

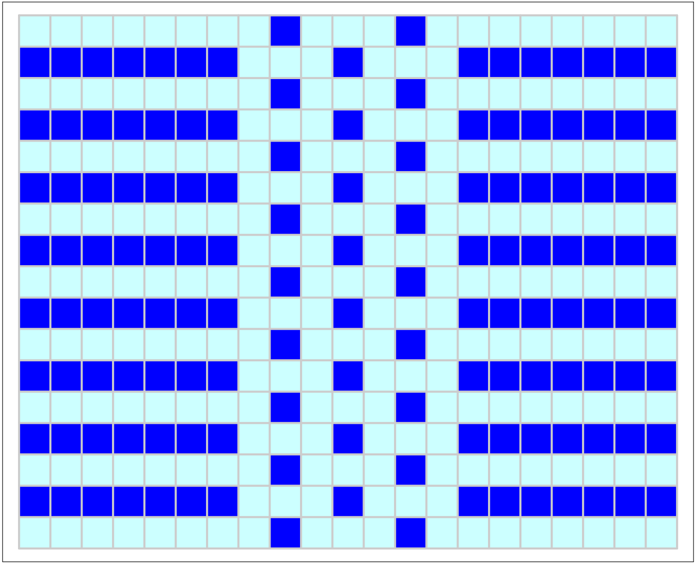
Klaus Sutner

ECA 1

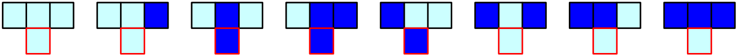


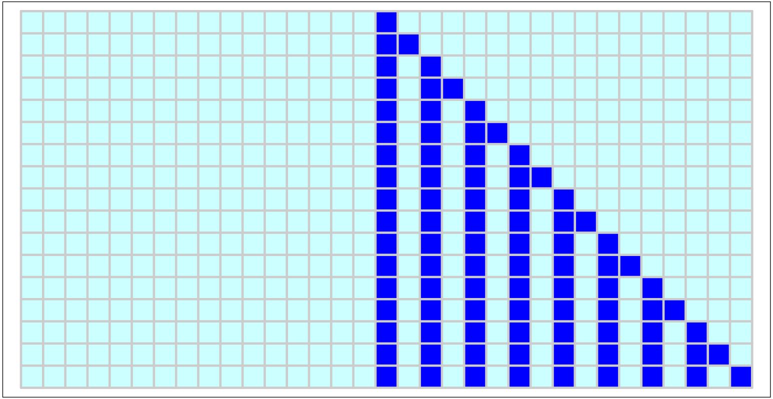


ECA 1

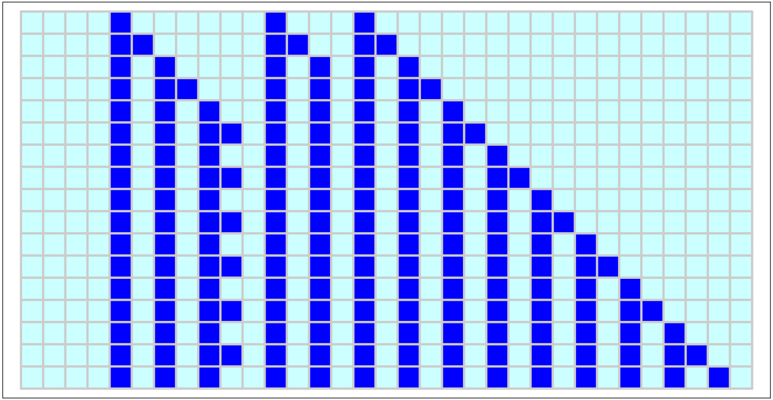


ECA 28

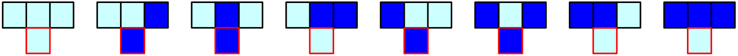


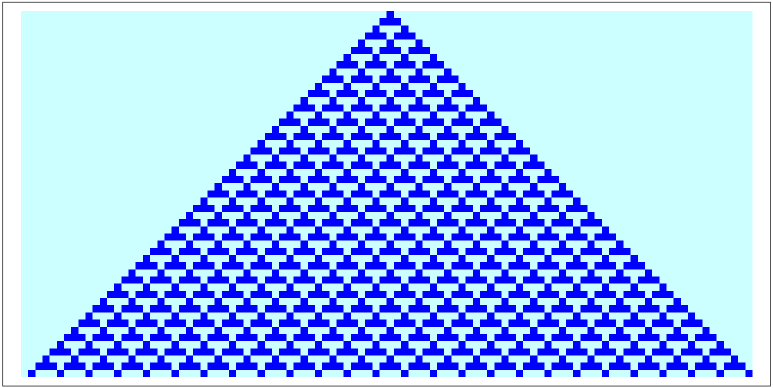


ECA 28

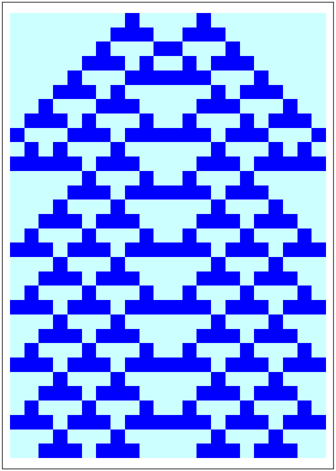


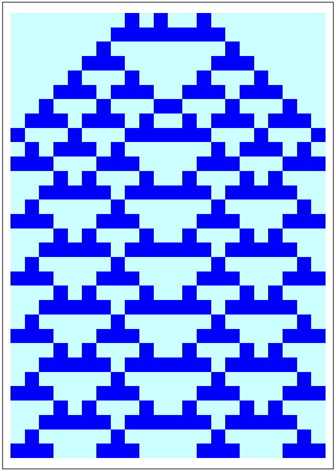
ECA 54



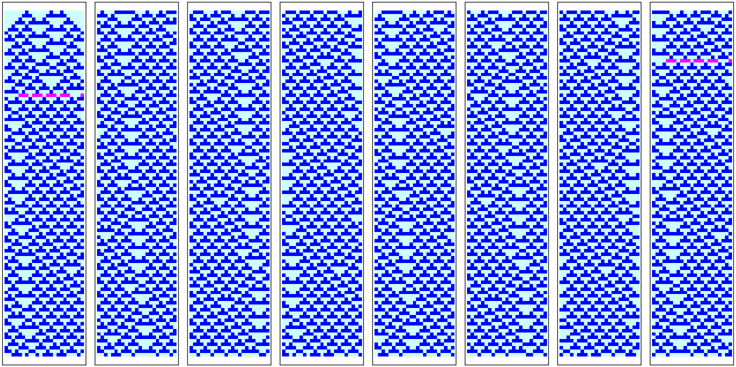


Multiple Seeds



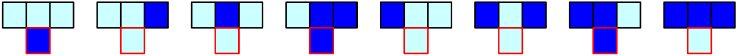


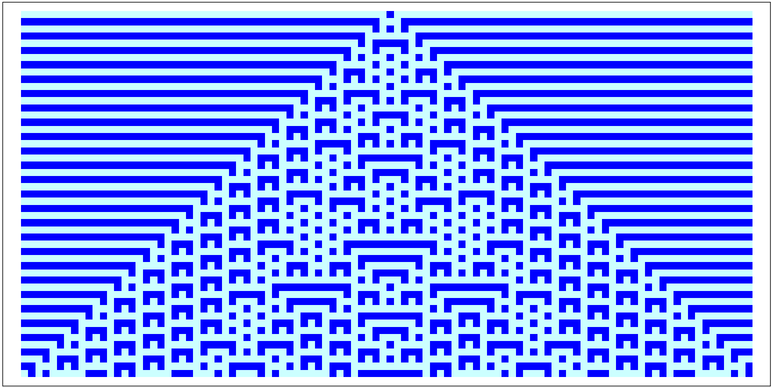
Disaster



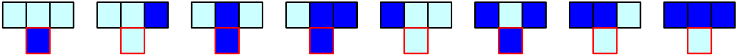
$n = 23,\ t = 24,\ p = 690$

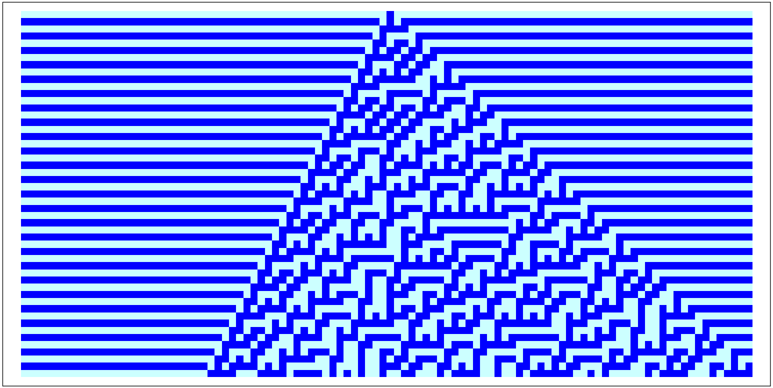
ECA 73



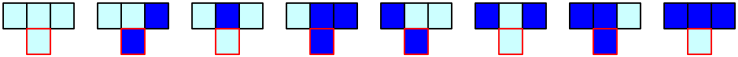


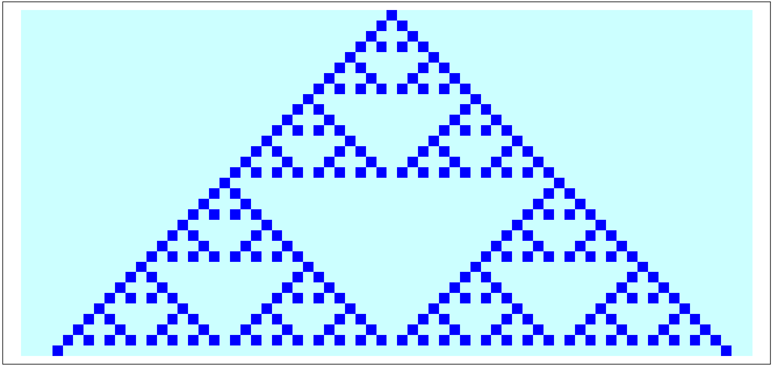
ECA 45



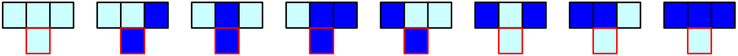


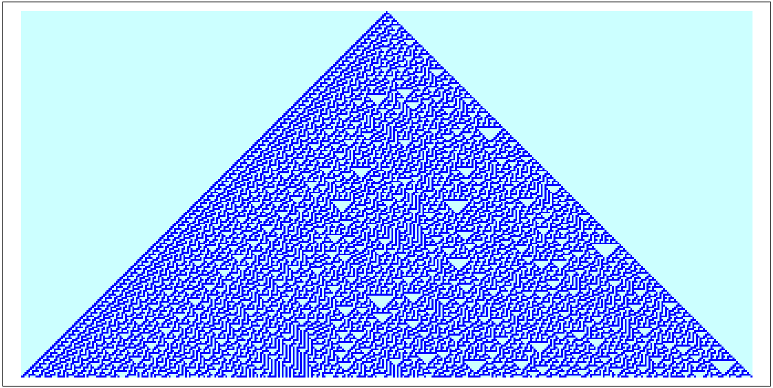
ECA 90



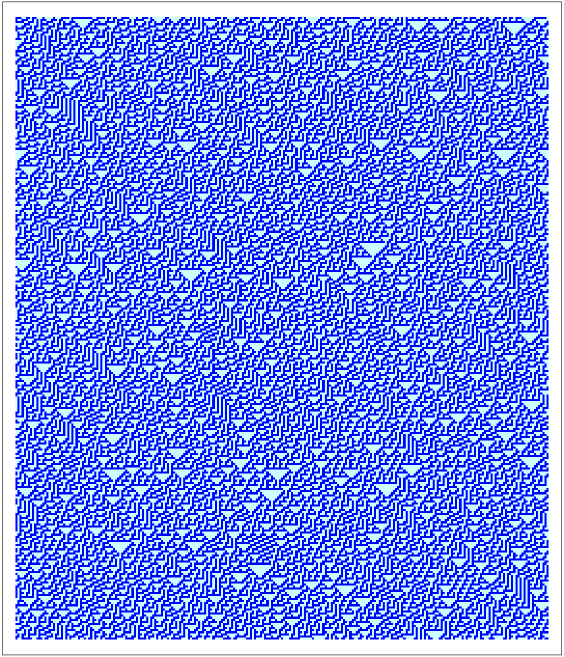


ECA 30

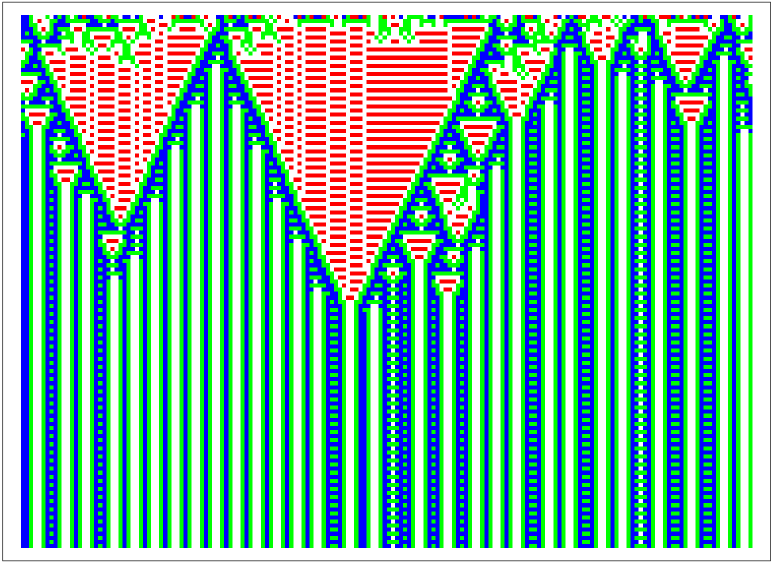




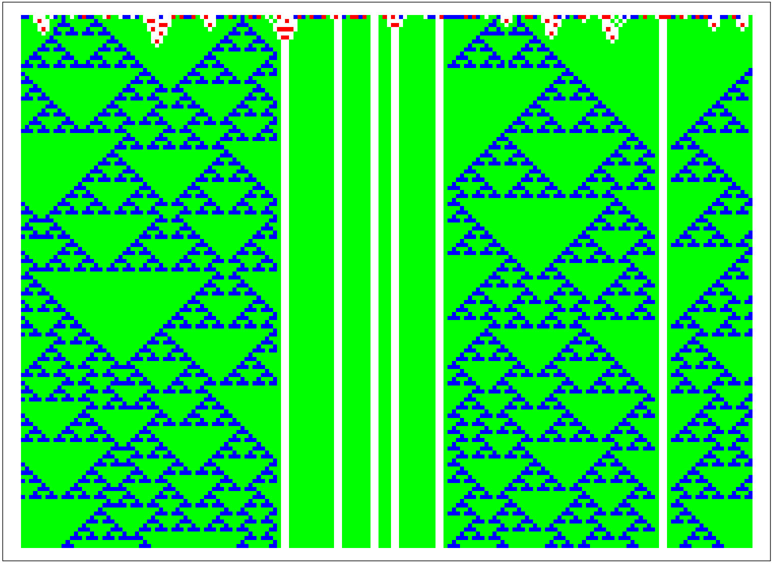
ECA 30



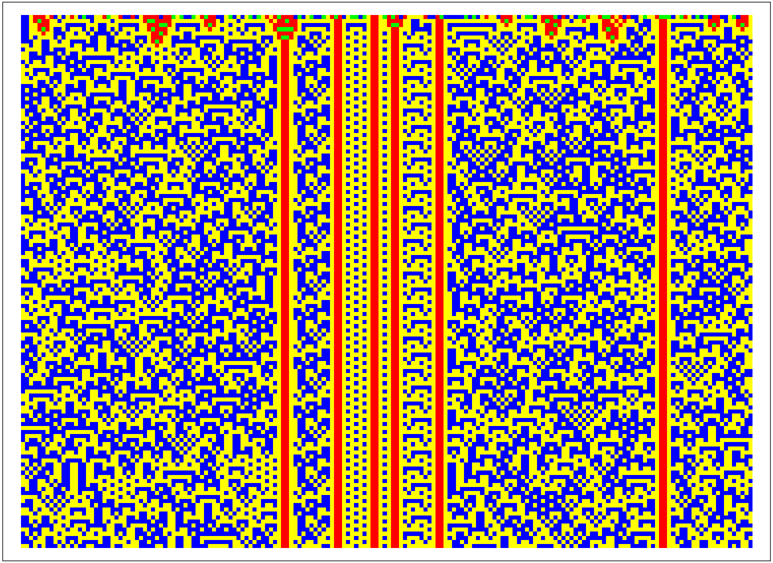
782359



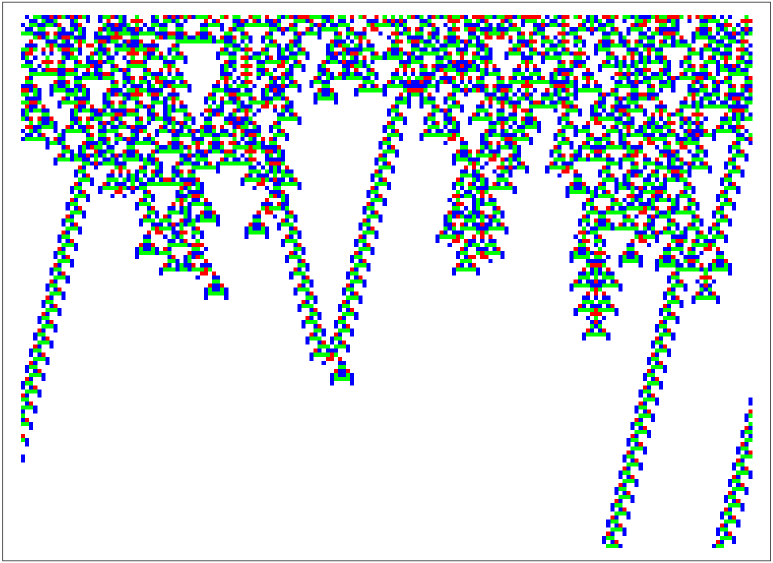
782340



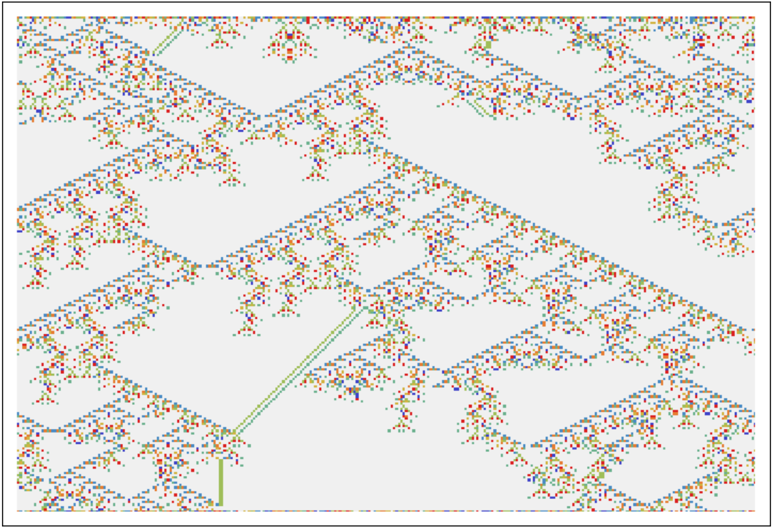
782353



722797



Langton's CA



- Cellular Automata

2. CA and Correctness Proofs

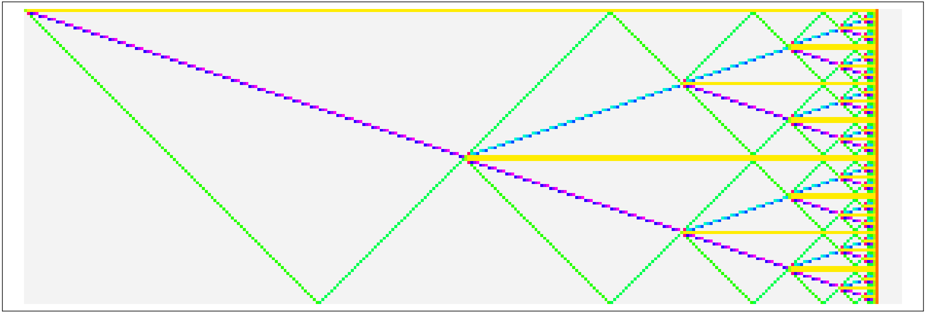
A Hard Problem

How does one prove properties of cellular automata?

The short term behavior (first-order logic) is decidable via automata theory.

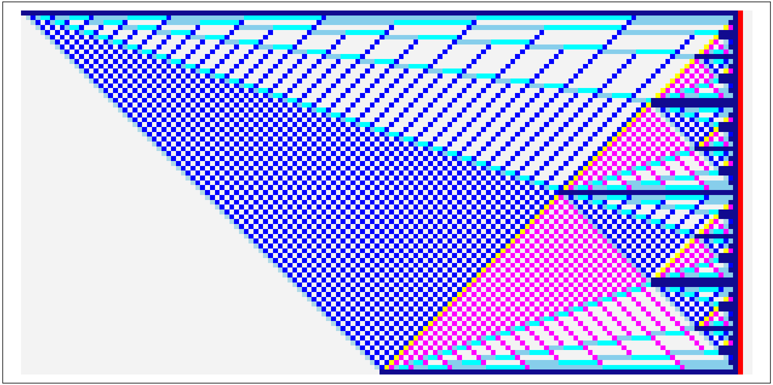
Almost any question about the long term behavior is undecidable; there is no standardized argument.

Moews



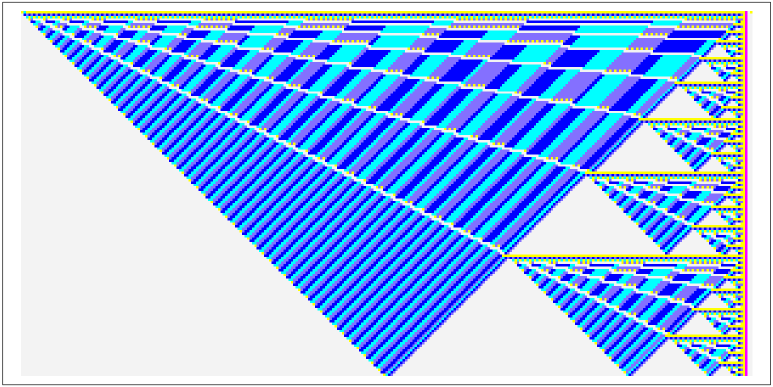
13 states

Waksman



9 states

Mazoyer



6 states

Mazoyer's Proof

The paper in TCS is 54 pages long and uses lots of diagrams.

To be sure, the proof is almost certainly correct, but it contains lots of little, unrelated combinatorial facts that are difficult for a human to check – eyes glaze over very quickly.

The proof should be machine checked.

Universality of ECA 110

ECA 110 is given by the following local rule:

$$\begin{array}{ll} 000 \to 0 & 100 \to 0 \\ 001 \to 1 & 101 \to 1 \\ 010 \to 1 & 110 \to 1 \\ 011 \to 1 & 111 \to 0 \end{array}$$

As a Boolean function this comes down to

$$\rho(x, y, z) = (\overline{x} \wedge y) \vee (y \oplus z)$$

Another Look

Here is the table again, but ordered differently:

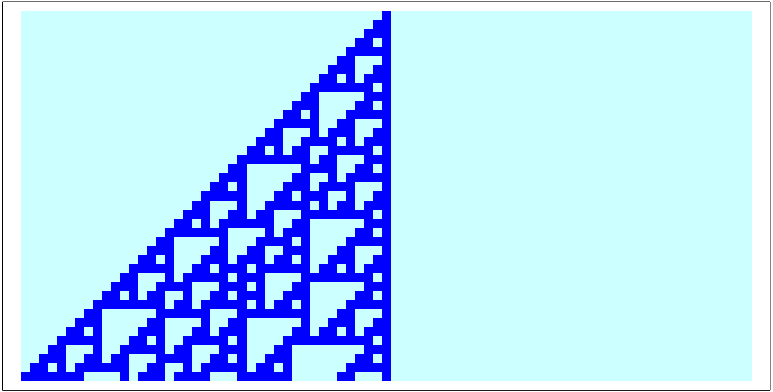
$$\begin{array}{ll} 000 \to 0 & 010 \to 1 \\ 001 \to 1 & 011 \to 1 \\ 100 \to 0 & 110 \to 1 \\ 101 \to 1 & 111 \to 0 \end{array}$$

The left column has "control bit" $y = 0$ and amounts to a left shift of z.

The right column has "control bit" $y = 1$ and amounts to $\mathrm{NAND}(x, z)$.

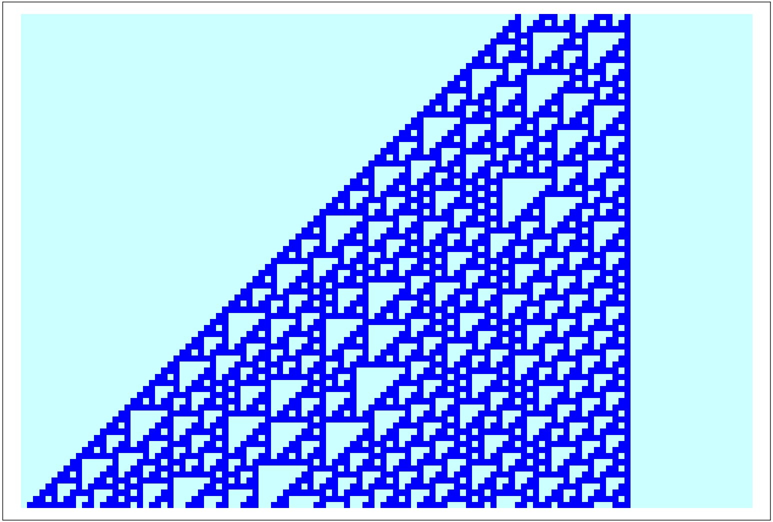
Since NAND is a functionally complete set of Boolean operations ...

One-Point Seed

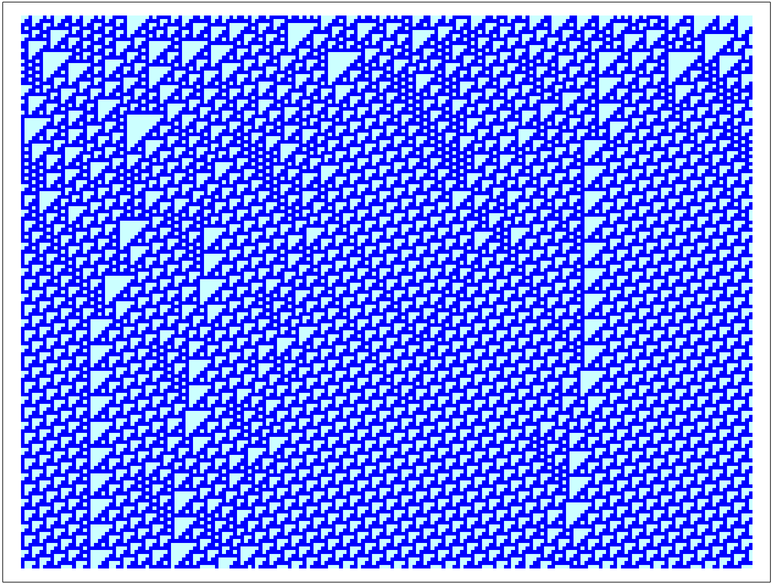


We get a half-light-cone: the configuration grows to the left only, at speed 1. Does not look particularly complicated.

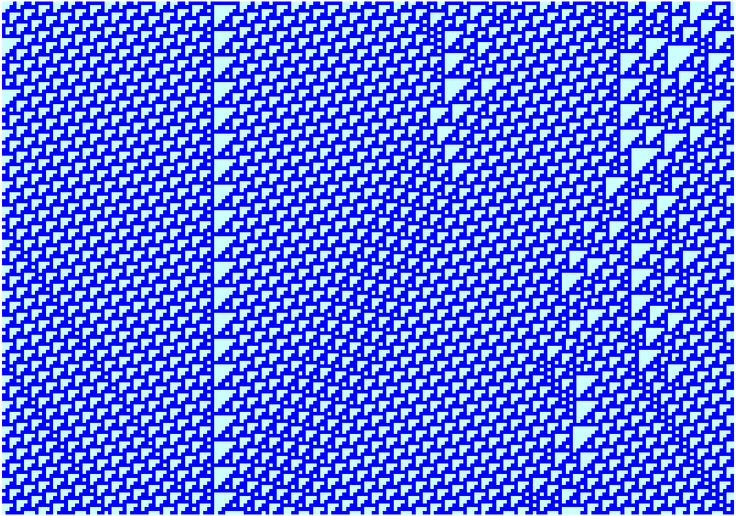
Finite Seed



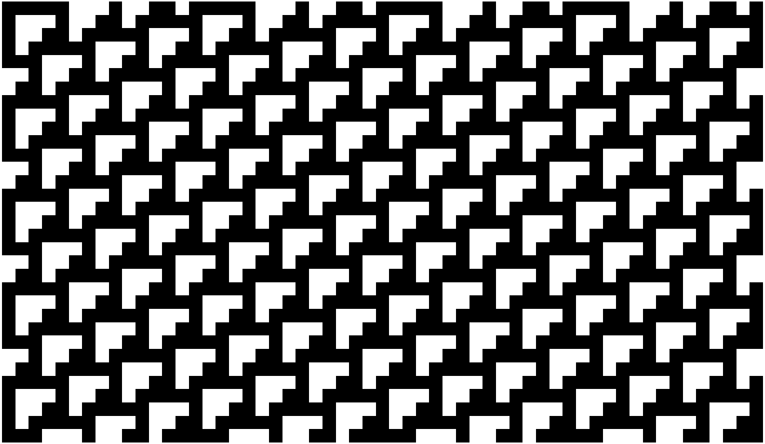
Random Seed



After 1000 Steps

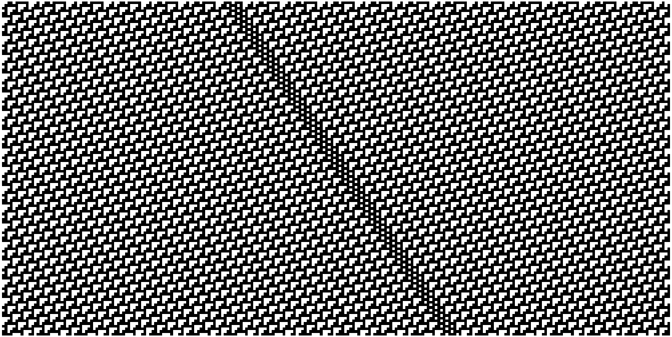


Background



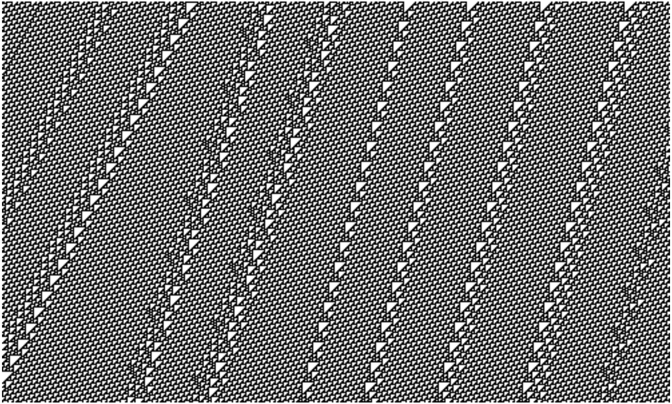
A background pattern naturally evolves; think of this as vacuum.

Particles

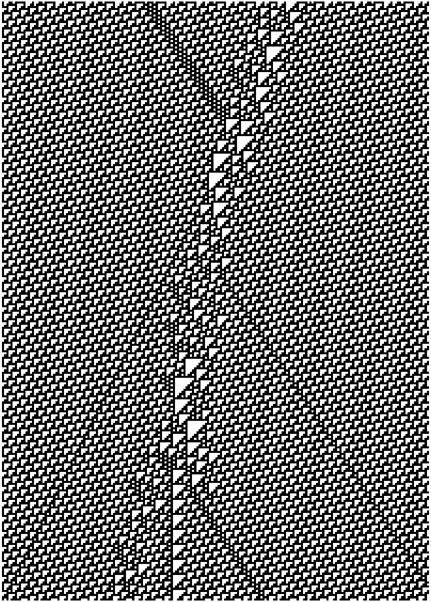


Particles can move in this vacuum.

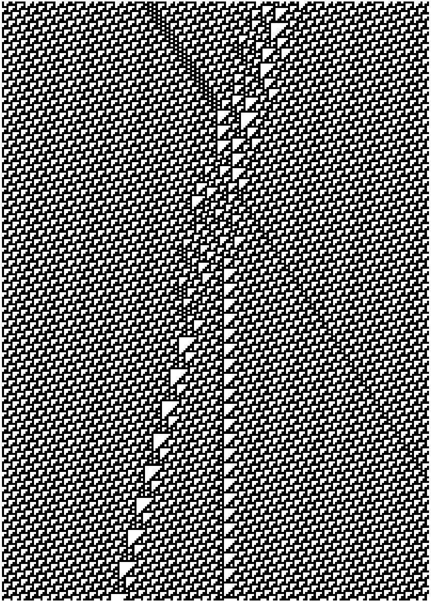
More Particles



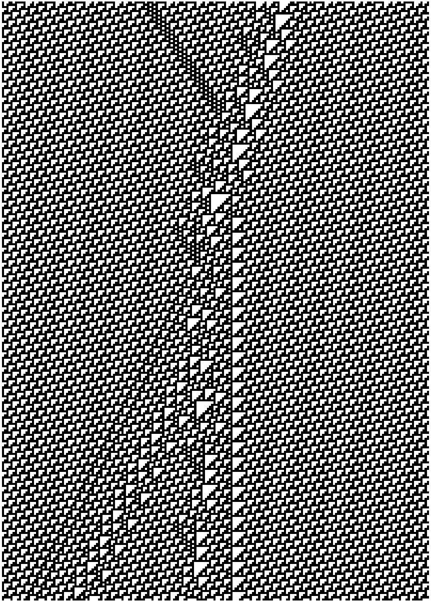
Interactions



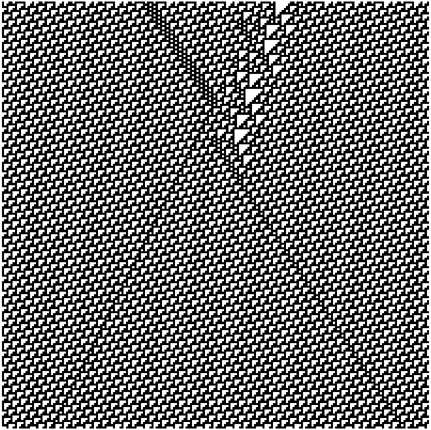
Interactions



Interactions

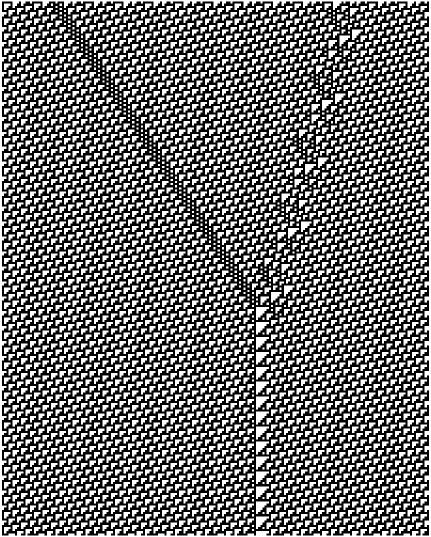


Interactions



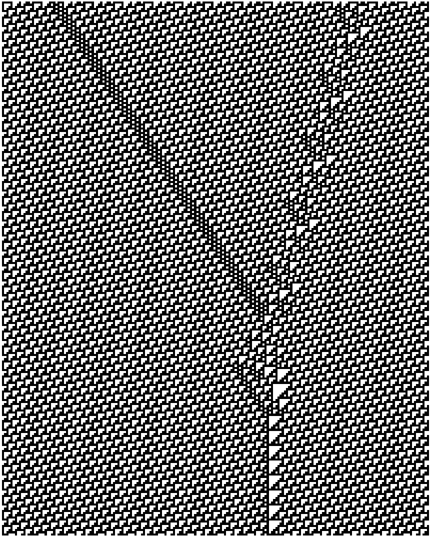
and interact . . .

Interactions



and interact . . .

Interactions



and interact . . .

Putting it Together

In the mid-90's, Matthew Cook, working at WRI, developed a fancy simulation system that allowed him to design and experiment with large configurations on ECA 110.

Based on his observations, he was able to show universality assuming almost periodic configurations by simulating cyclic tag systems, a variant of Post tag systems.

This is arguable the most interesting result in the study of computational universality in the last two or three decades. It's also the only result that lead to a law suit.

Verification

A number of people have checked the proof in great detail and there are even attempts at producing a more formalized versions of it.

Again, the proof is probably correct but it is really impossible to check all the details by hand – we need a computer-checkable version.