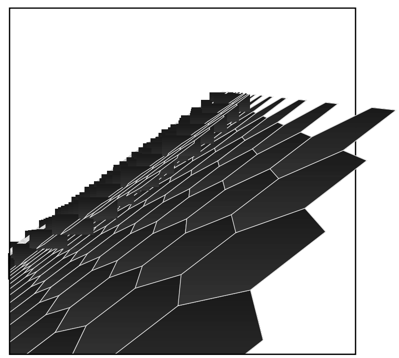
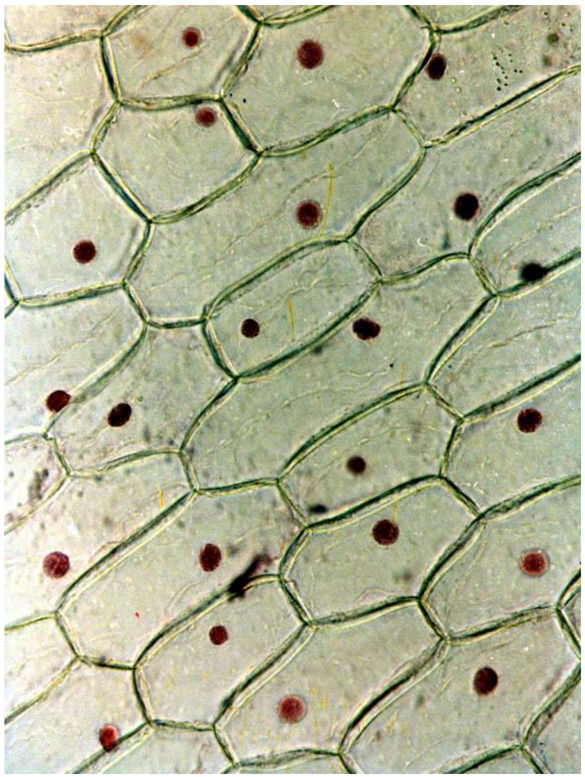
Cellular Systems



Motivation

Evolution has rediscovered several times multicellularity as a way to build complex living systems



- Multicellular systems are composed by many copies of a unique fundamental unit - the cell
- The local interaction between cells influences the fate and the behavior of each cell
- The result is an heterogeneous system composed by differentiated cells that act as specialized units, even if they all contain the same genetic material and have essentially the same structure

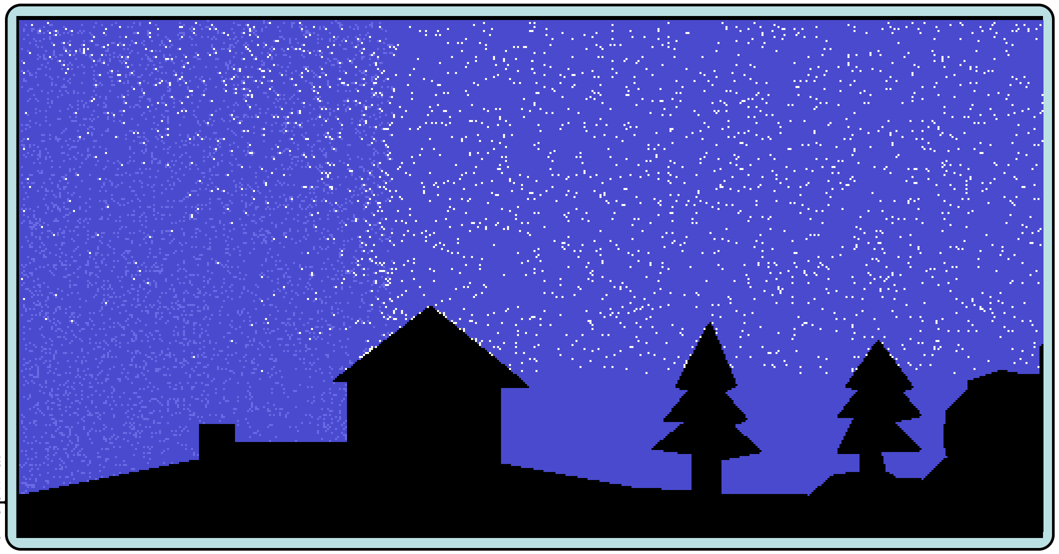
Fields of Application

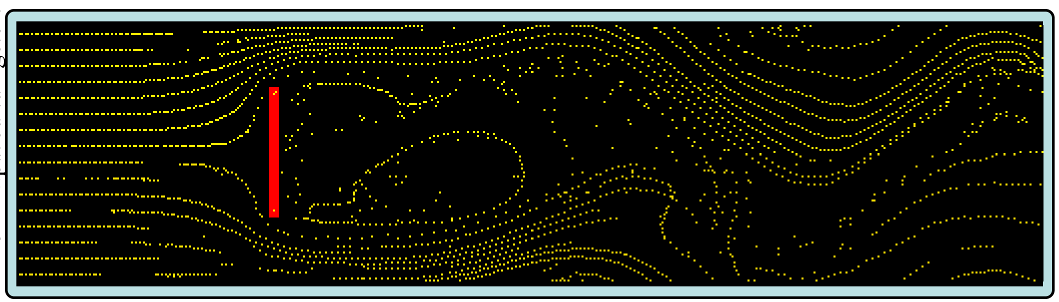
The concept of “many simple systems with (geometrically structured) local interaction” is relevant to:

- **Artificial Life** and **Evolutionary Experiments**, where it allows the definition of arbitrary “synthetic universes”.
- **Computer Science** and **Technology** for the implementation of parallel computing engines and the study of the rules of emergent computation.
- **Physics**, **Biology,** and other sciences, for the modeling and simulation of complex biological, natural, and physical systems and phenomena, and research on the rules of structure and pattern formation.
  - More generally, the study of **complex systems**, i.e., systems composed by many simple units that interact non-linearly
- **Mathematics**, for the definition and exploration of complex space-time dynamics and of the behavior of dynamical systems.

Modeling complex phenomena

Many complex phenomena are the result of the collective dynamics of a very large number of parts obeying simple rules.





from http://cui.unige.ch/~chopard/

Unexpected global behaviors and patterns can emerge from the interaction of many systems that “communicate” only locally.

Modeling cellular systems

We want to define the simplest nontrivial model of a cellular system. We base our model on the following concepts:

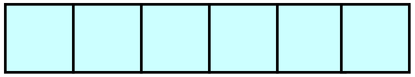
- *Cell* and *cellular space*
- *Neighborhood* (local interaction)
- Cell *state*
- *Transition rule*

We do not model all the details and characteristics of biological multicellular organisms but we obtain simple models where many interesting phenomena can still be observed

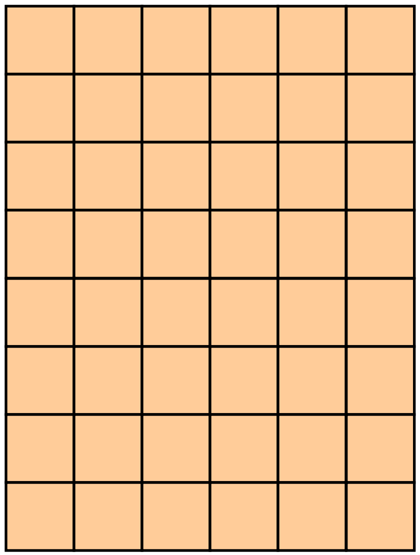
- There are many kinds of cellular system models based on these concepts
- The simplest model is called *Cellular Automaton* (CA)

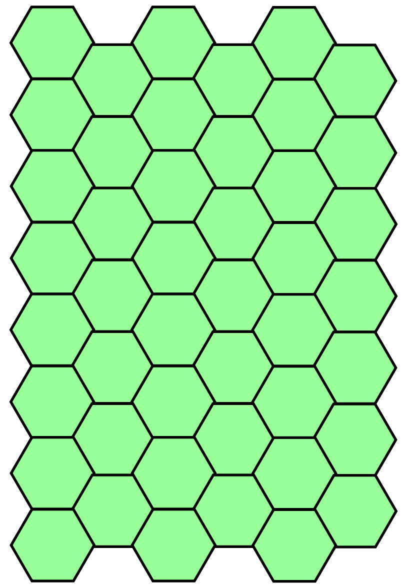
Cellular space

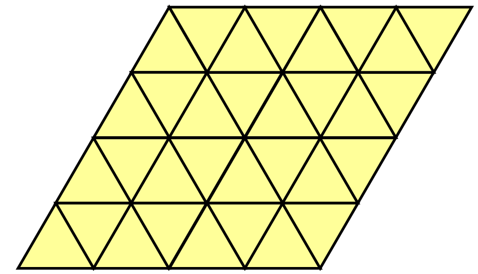
1D



2D

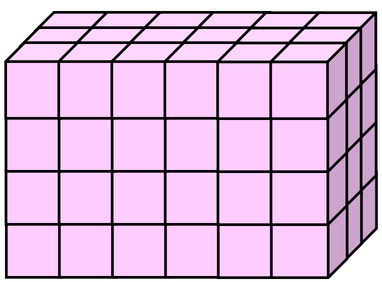






...

3D

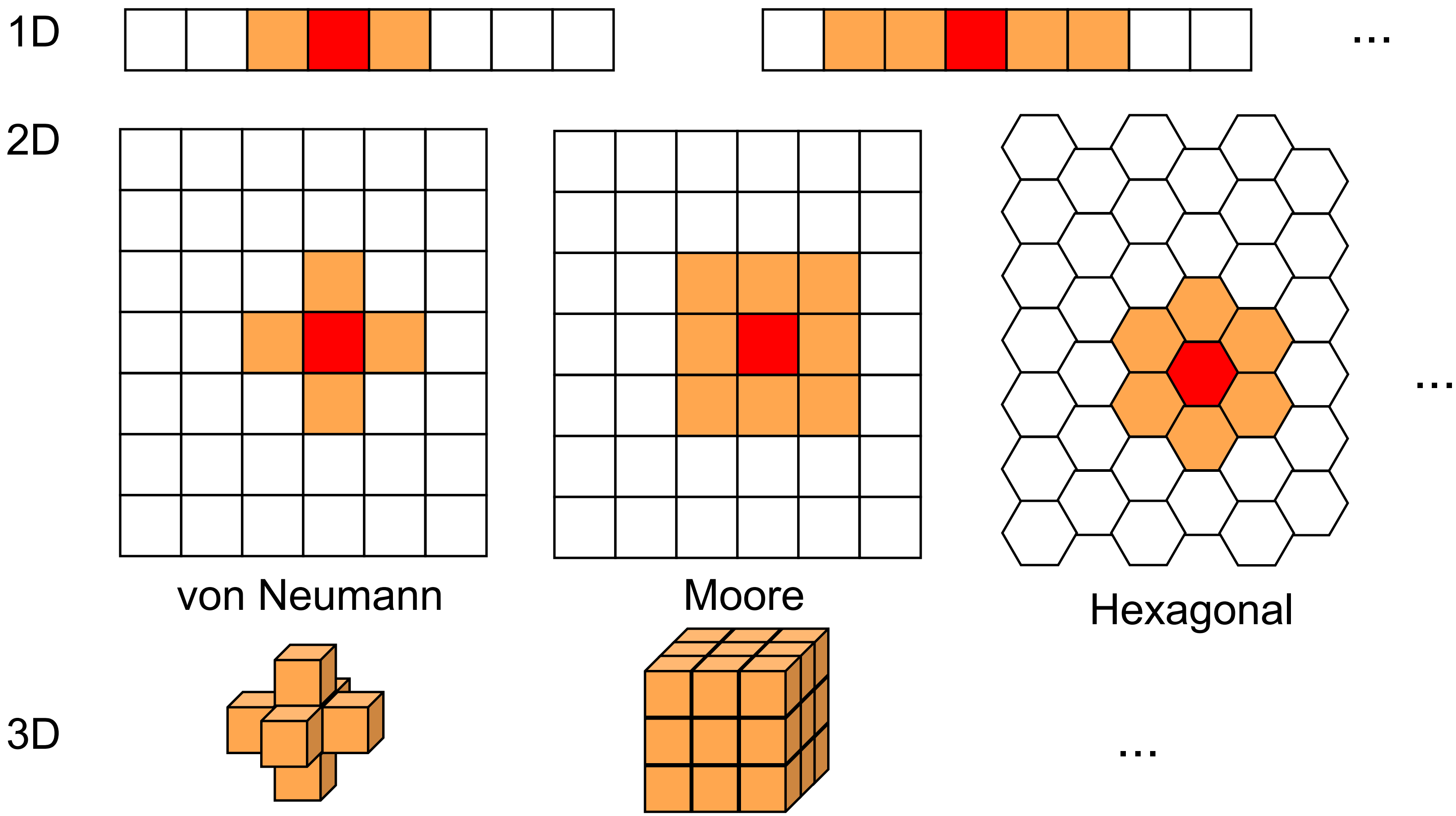


...

and beyond...

Neighborhood

- Informally, it is the set of cells that can influence *directly* a given cell
- In *homogeneous* cellular models it has the same shape for all cells



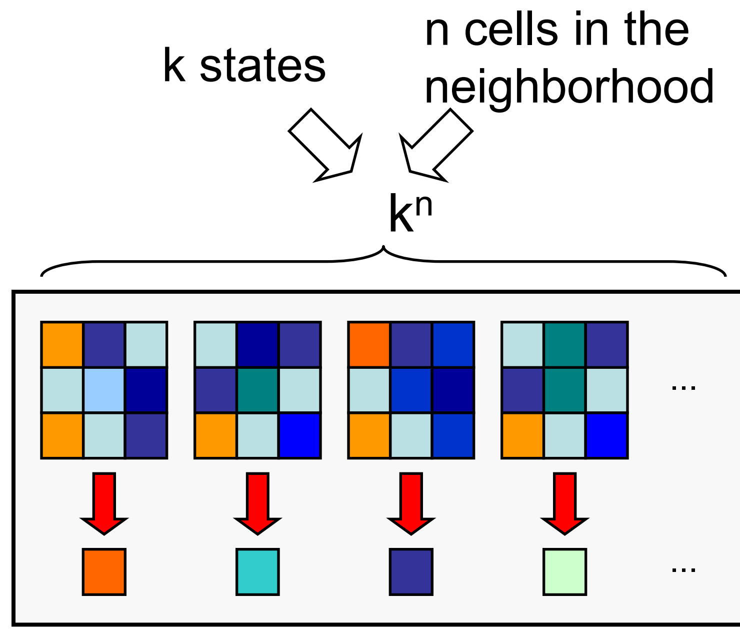
State Set and Transition Rule

The value of the **state** of each cell belong to a finite set, whose elements we can assume as being numbers. The value of the state is often represented by cell colors. There can be a special **quiescent state** s_0.

$$
\begin{aligned}
S &= \{s_0, \ldots, s_{k-1}\} \\
&= \{0, \ldots, k-1\} \\
&= \{\bullet, \ldots, \bullet\}
\end{aligned}
$$

The **transition rule** is the fundamental element of the CA. It must specify the new state corresponding to each possible configuration of states of the cells in the neighborhood.

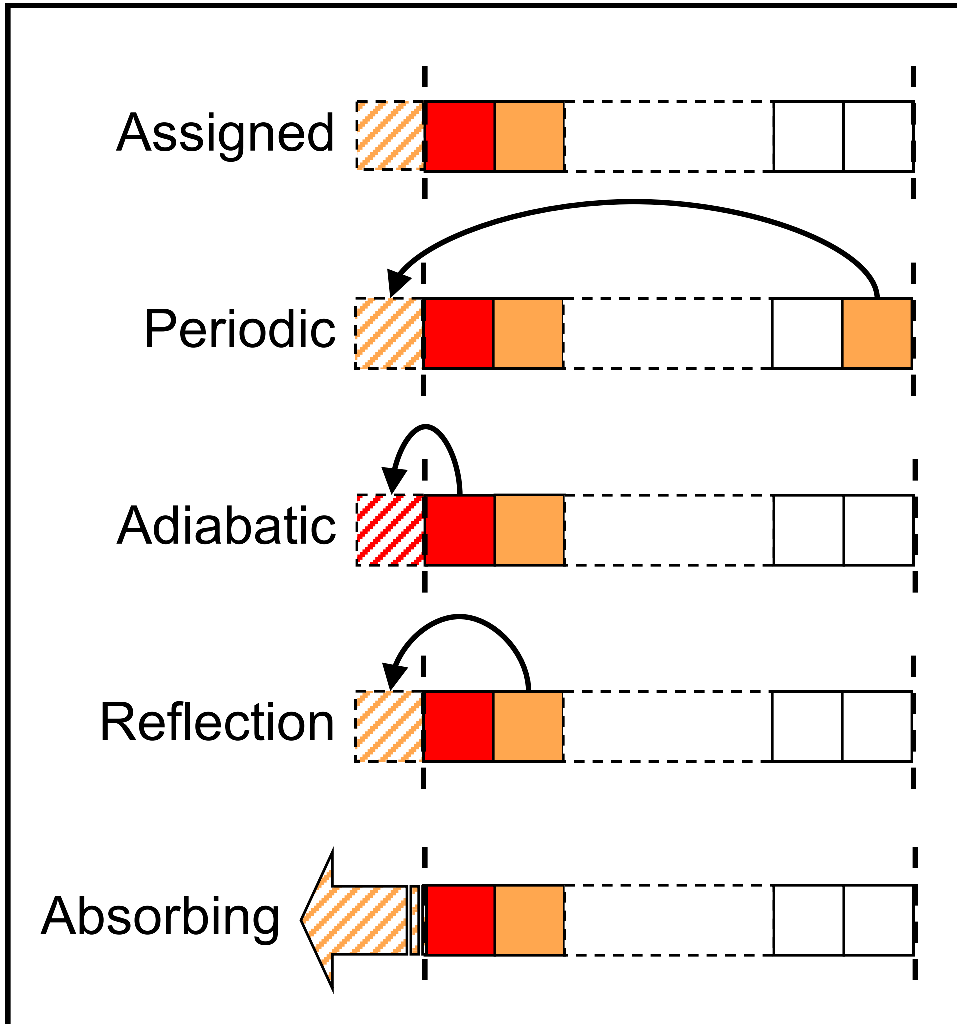
The transition rule can be represented as a **transition table**, although this becomes rapidly impractical.



transition table

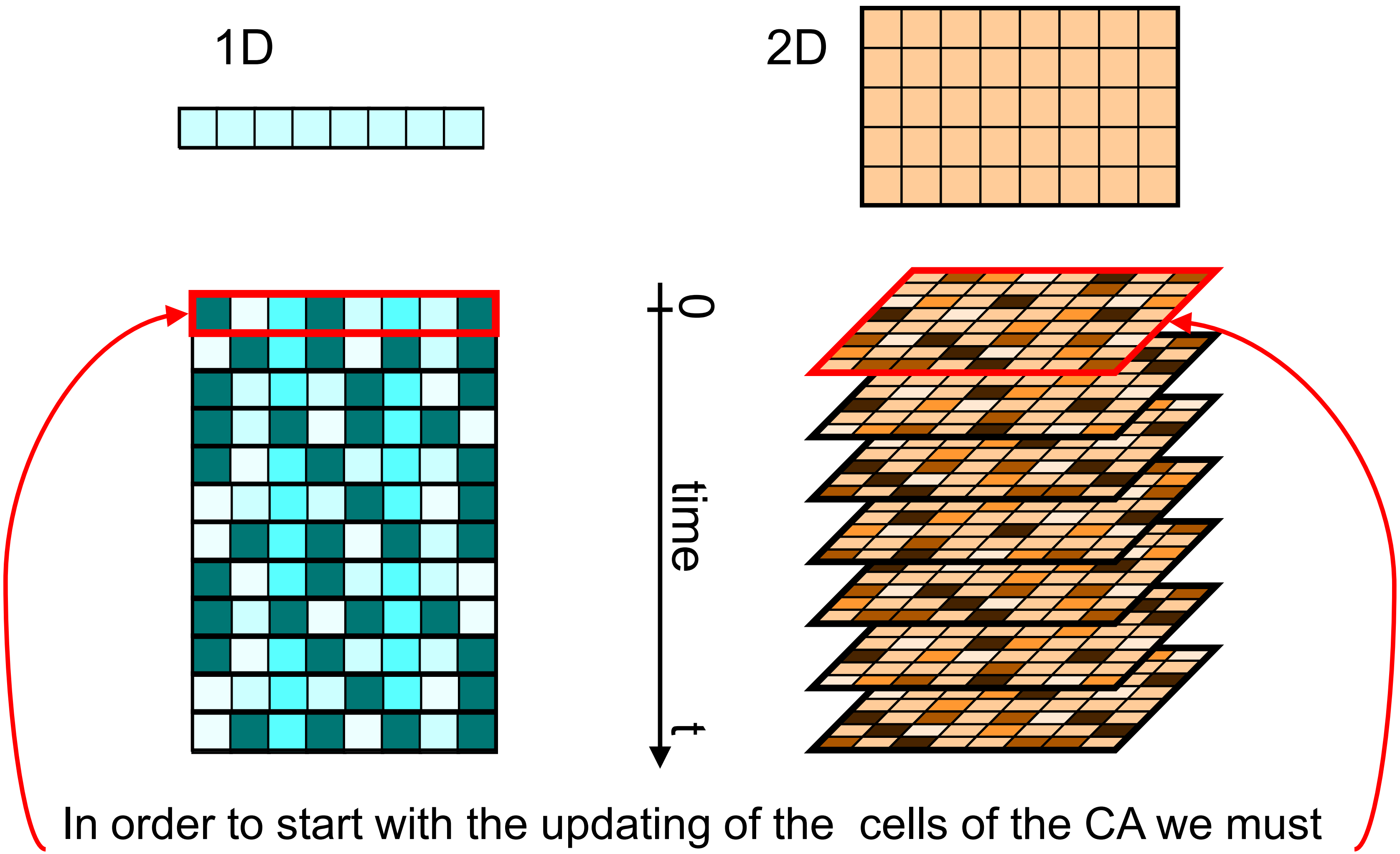
Boundary Conditions

- If the cellular space has a boundary, cells on the boundary may lack the cells required to form the prescribed neighborhood
- *Boundary conditions* specify how to build a "virtual" neighborhood for boundary cells



Some common kinds of boundary conditions

Initial Conditions

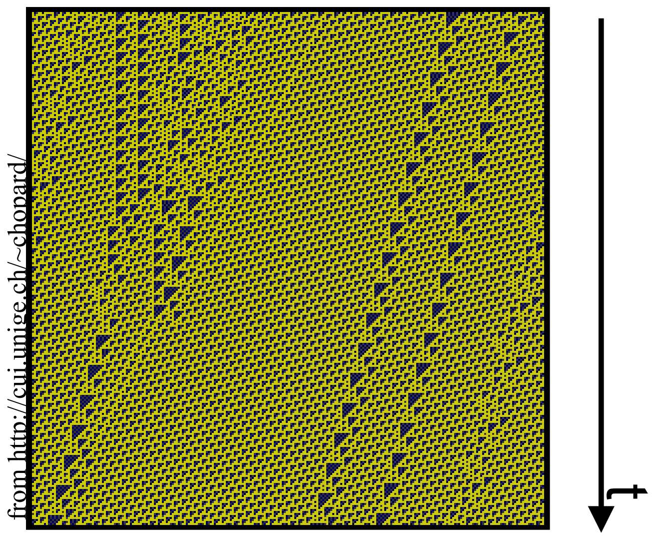


In order to start with the updating of the cells of the CA we must specify the initial state of the cells (**initial conditions** or **seed**)

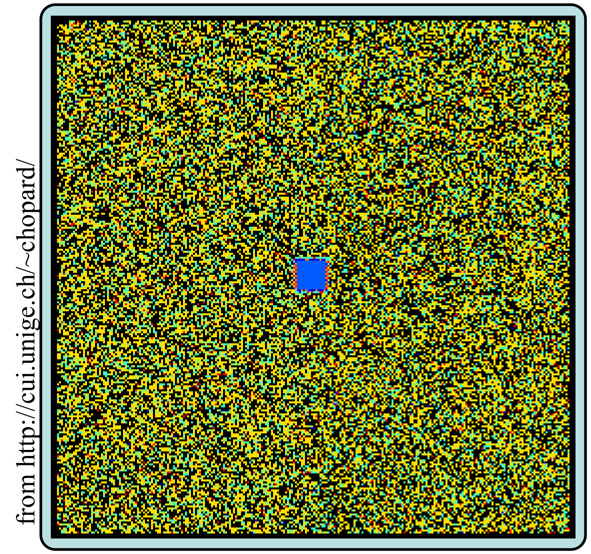
Displaying CA dynamics

1D

Space-time animation
(or static plot)

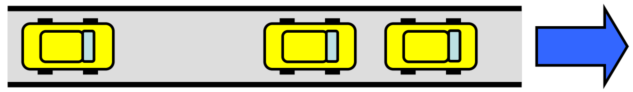


2D

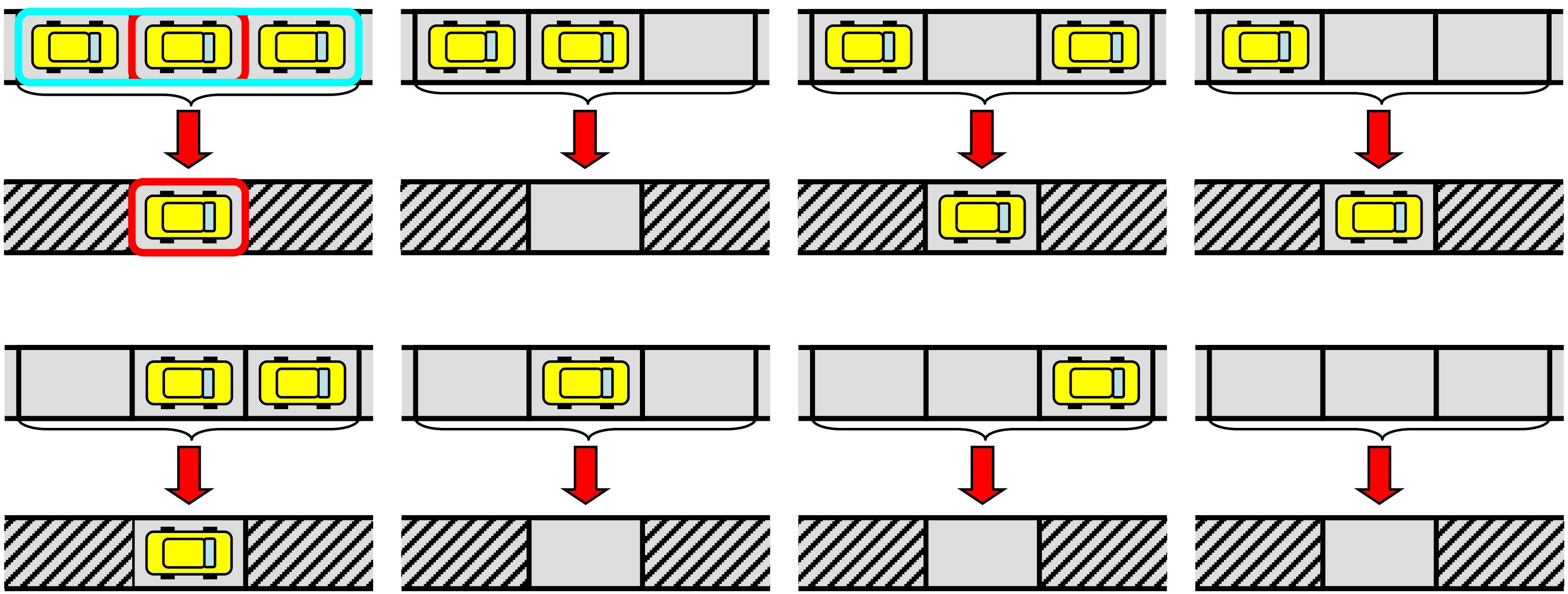


animation of spatial plot
(signaled by the border
in this presentation)

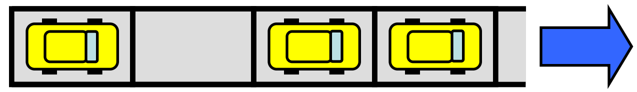
Example: Modeling Traffic



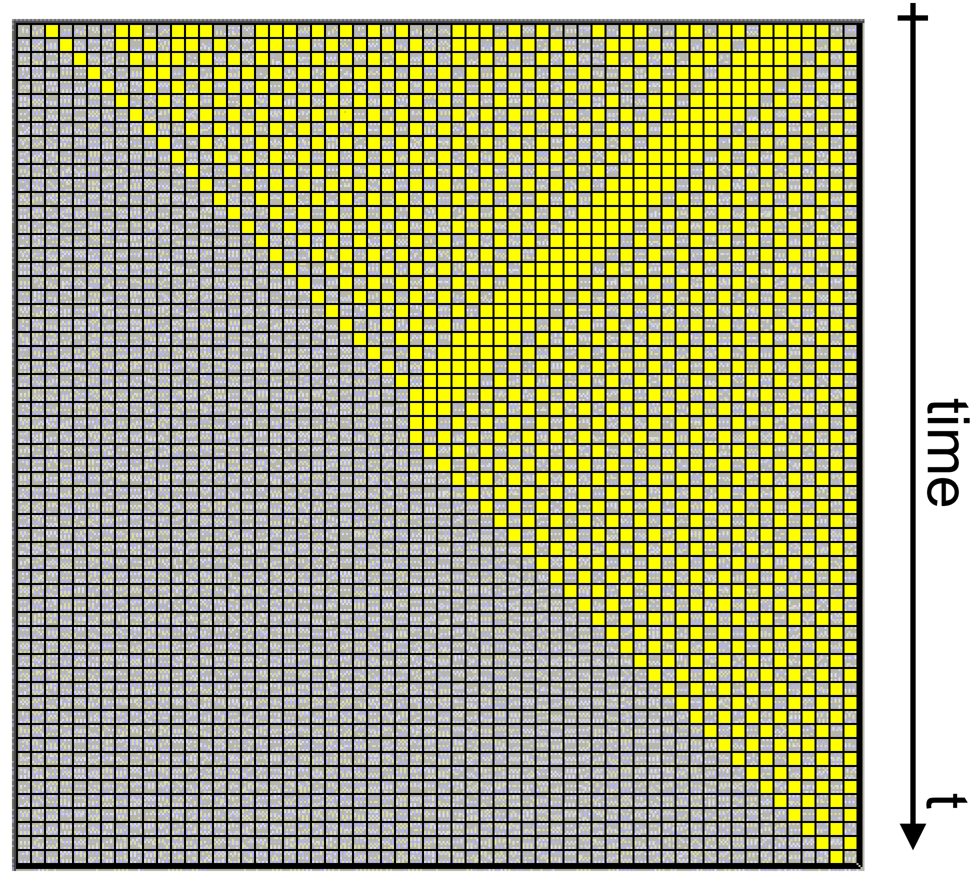
We construct an elementary model of car motion in a single lane, based only on the **local** traffic conditions. The cars advance at discrete time steps and at discrete space intervals. A car can advance (and must advance) only if the destination interval is free.



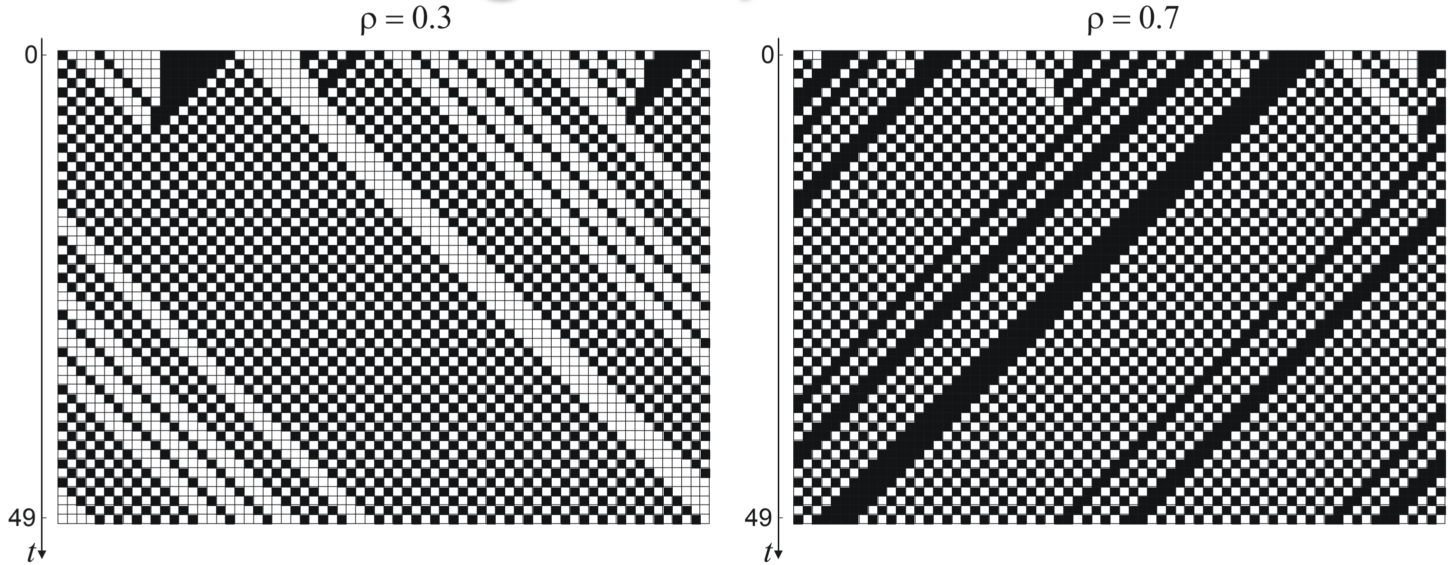
Example: Traffic Jam

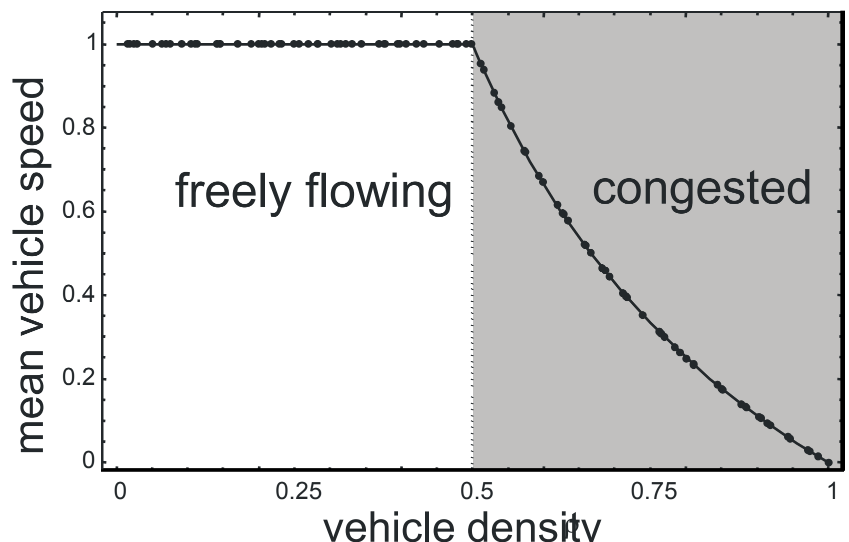


Running the *traffic CA* with a high-density random initial distribution of cars we observe a phenomenon of backward propagation of a region of extreme traffic congestion (traffic jam).



Emergent phenomena





There is a qualitative change of behavior for ρ = 0.5. In the language of physics there is a *phase transition* between the two regimes at the *critical density* ρ = 0.5

In practice...

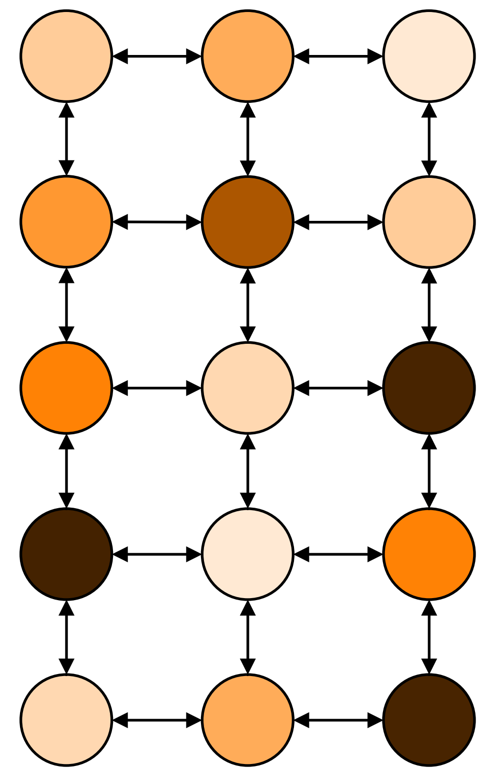
To implement and run a CA experiment

1. Assign the geometry of the CA space
2. Assign the geometry of the neighborhood
3. Define the set of states of the cells
4. Assign the transition rule
5. Assign the boundary conditions
6. Assign the initial conditions of the CA
7. Repeatedly update all the cells of the CA, until some stopping condition is met (for example, a pre-assigned number of steps is attained, or the CA is in a quiescent state, or cycles in a loop,...).

Informal definition of CA

A Cellular Automaton is

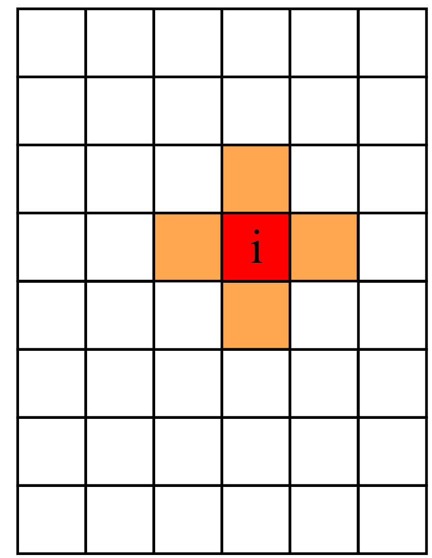
- a **geometrically structured** and
- **discrete** collection of
- **identical** (simple) systems called **cells**
- that **interact** only **locally**
- with each cell having a local **state** (memory) that can take a **finite** number of values
- and a (simple) **rule** used to **update** the state of all cells
- at **discrete time** steps
- and **synchronously** for all the cells of the automaton (global "signal")



Formal definition of CA

A Cellular Automaton is

- an **n-dimensional lattice** of
- **identical** and **synchronous** finite state machines
- whose state s is updated (synchronously) following a **transition function** (or transition rule) ϕ
- that takes into account the state of the machines belonging to a **neighborhood** N of the machine, and whose geometry is the same for all machines



$$s_i(t+1) = \phi(\ s_j(t)\ ;\ j \in N_i\ )$$

Special Rules

The transition table of a generic CA can have an enormous number of entries. Special rules can have more compact definitions.

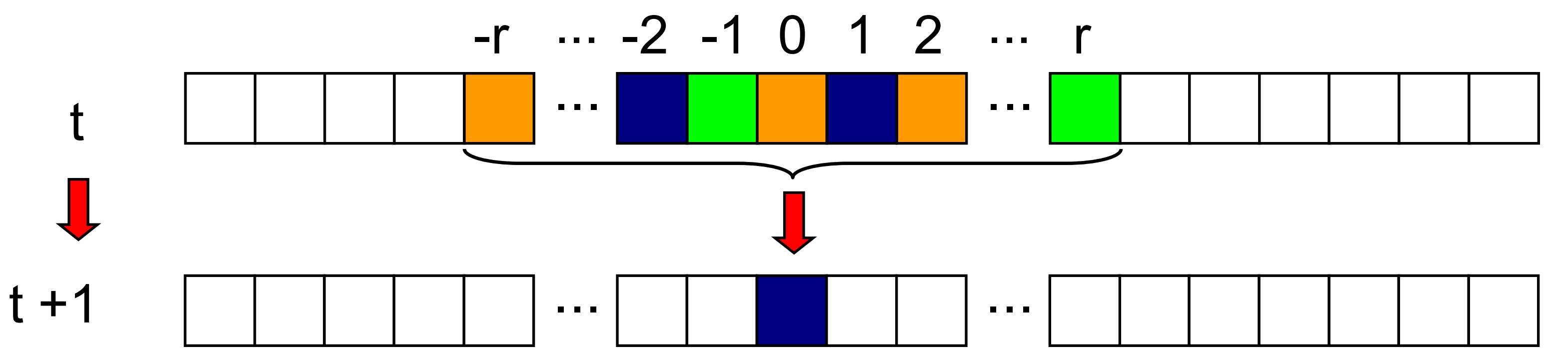
A rule is **totalistic** if the new value of the state depends only on the **sum** of the values of the states of the cells in the neighborhood

$$s_i(t+1) = \phi(\Sigma_j \, s_j(t) \; ; \; j \in N_i)$$

A rule is **outer totalistic** if the new value of the state depends on the value of the state of the updated cell and on the sum of the values of the states of the other cells in the neighborhood

$$s_i(t+1) = \phi(s_i(t) \, , \Sigma_j \, s_j(t) \; ; \; j \in N_i \, , j \neq i)$$

Rules for 1D CA



k states (colors ● , ● , ● , ...), range (or radius) r

$k^{k^{2r+1}}$ possible rules

e.g.: k=2 , r=1 $\rightarrow$ 256

k=3 , r=1 $\rightarrow \approx 8 \cdot 10^{12}$

$k^{(2r+1)(k-1)+1}$ totalistic rules

e.g.: k=2 , r=1 $\rightarrow$ 16 totalistic

k=3 , r=1 $\rightarrow$ 2187 totalistic

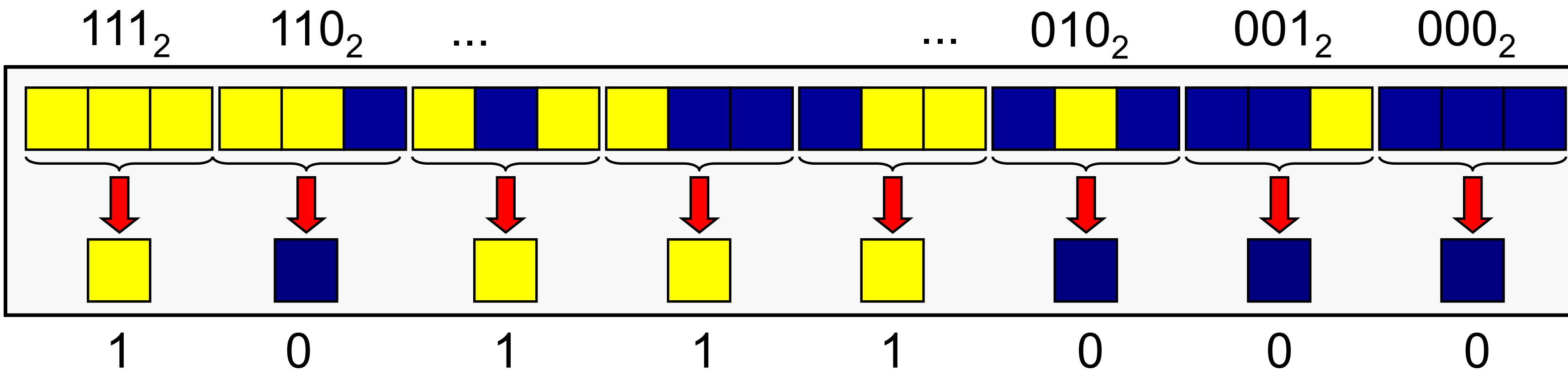
The number of possible rules grows very rapidly with k and r

Rule Code for Elementary CA

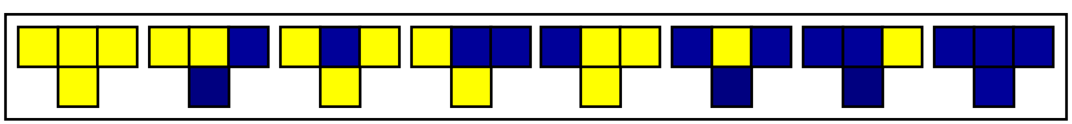
Elementary CA

256 1D binary CA (k=2) with minimal range (r=1)

Wolfram's Rule Code (here, ■ = 0 , ■ = 1)



$10111000_2 = 1 \cdot 2^7 + 0 \cdot 2^6 + \ldots + 0 \cdot 2^0 = 184_{10}$ ⇨ Rule 184

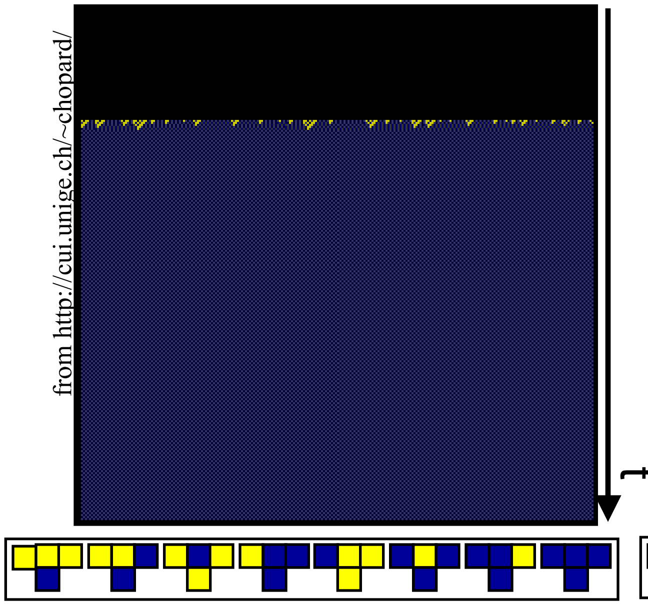


$R184 = R1,WB8_{16}$

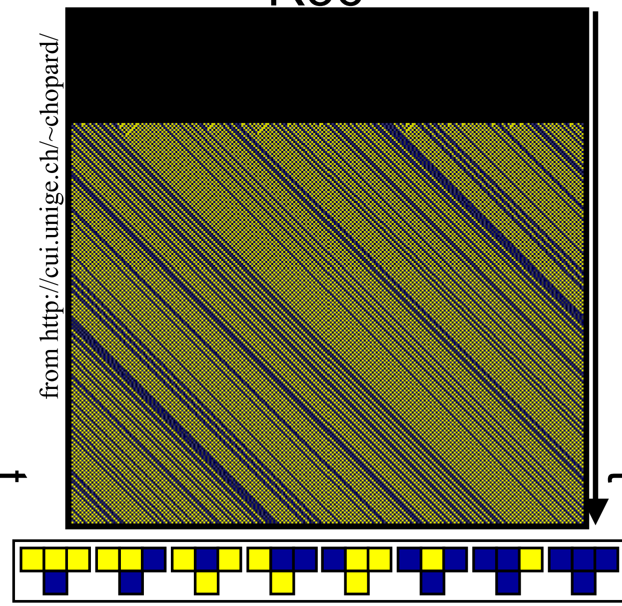
(the "car traffic" rule!)

Examples of Elementary CA

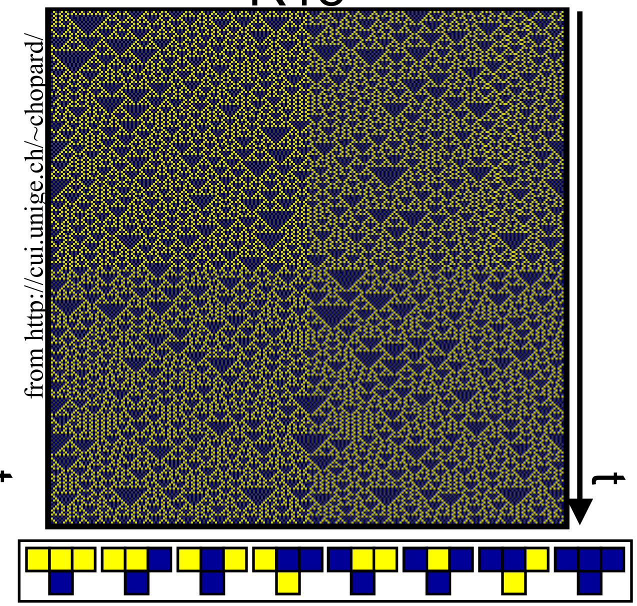
R40



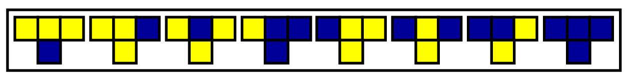
R56

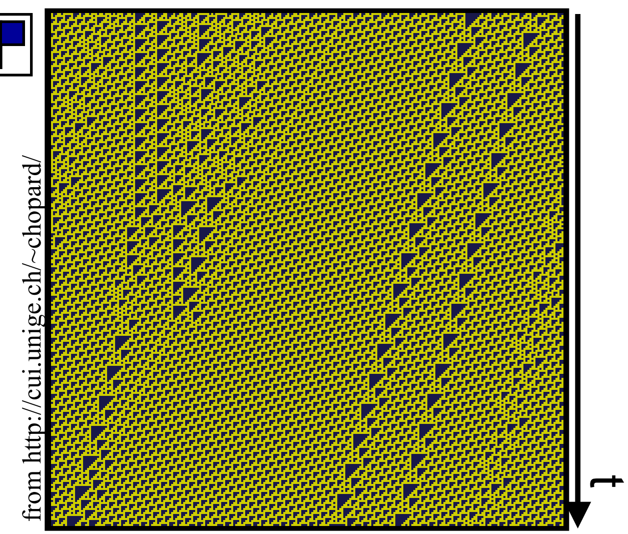


R18



R110



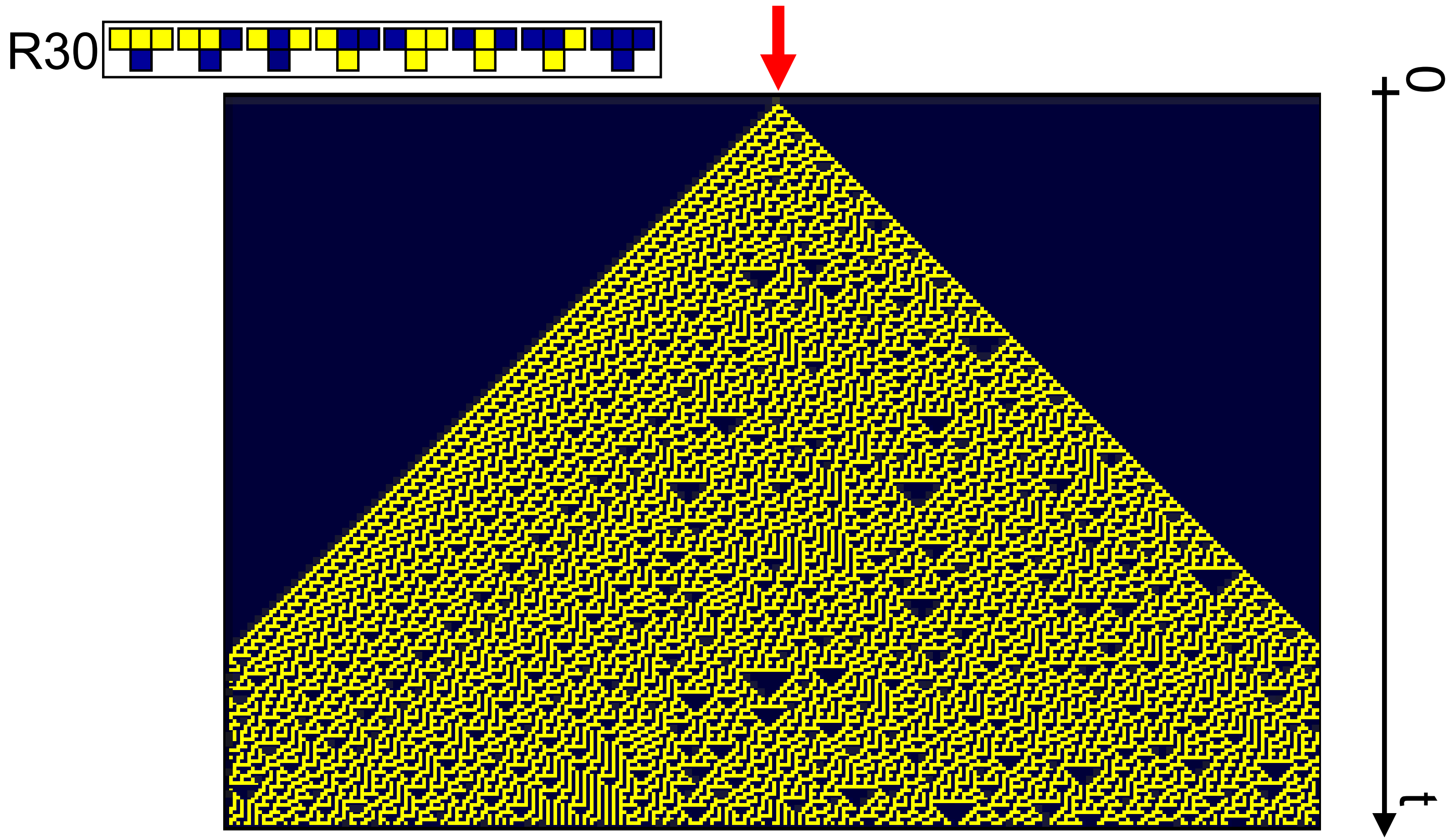


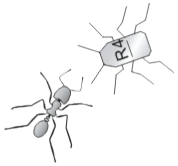
There are four *qualitative* behavioral classes:

1. Uniform final state
2. Simple stable or periodic final state
3. Chaotic, random, nonperiodic patterns
4. Complex, localized, propagating structures

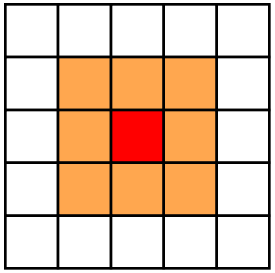
Example of application: RNG

Rule 30 is used by Mathematica as its Random Number Generator (RNG are ubiquitous in bio-inspired experiments).

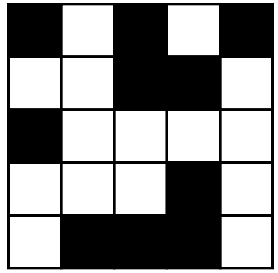




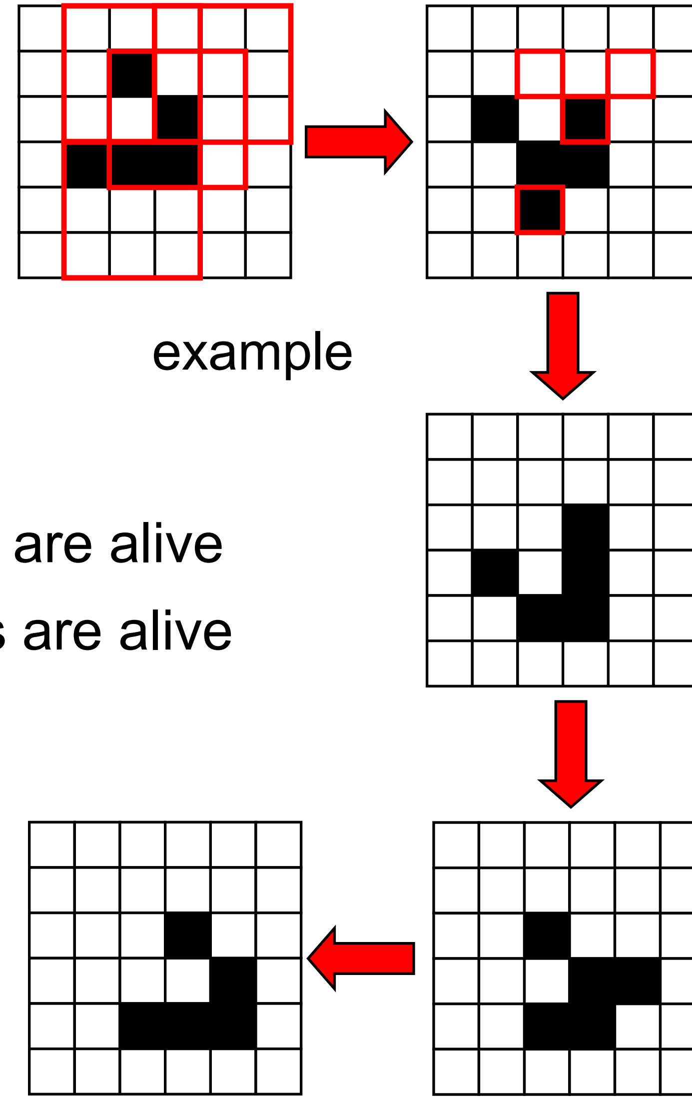
The classical 2D CA: *Life*



Moore neighborhood



two states
dead alive



example

Outer totalistic rule (John Conway)

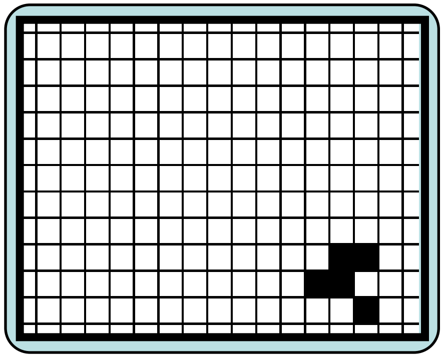
- **Birth** if exactly 3 neighbors are alive
- **Survival** if 2 or 3 neighbors are alive
- **Death**

from "isolation" if 0 or 1 n. a. a

from "overcrowding" if more than 3 neighbors are alive

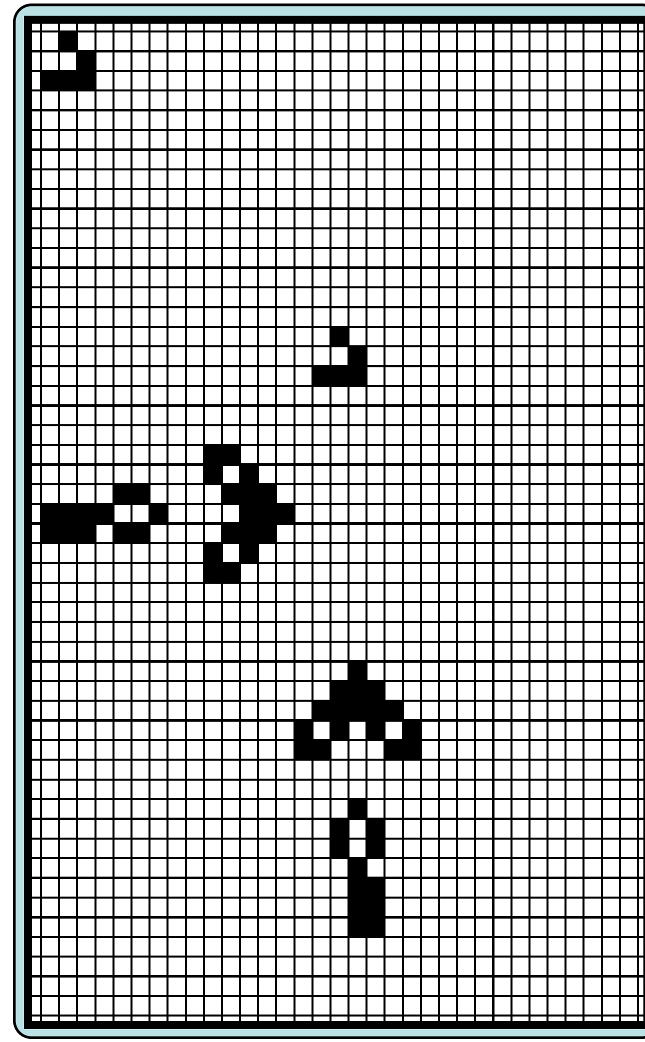
(often coded as "rule 23/3")

Computation in *Life*

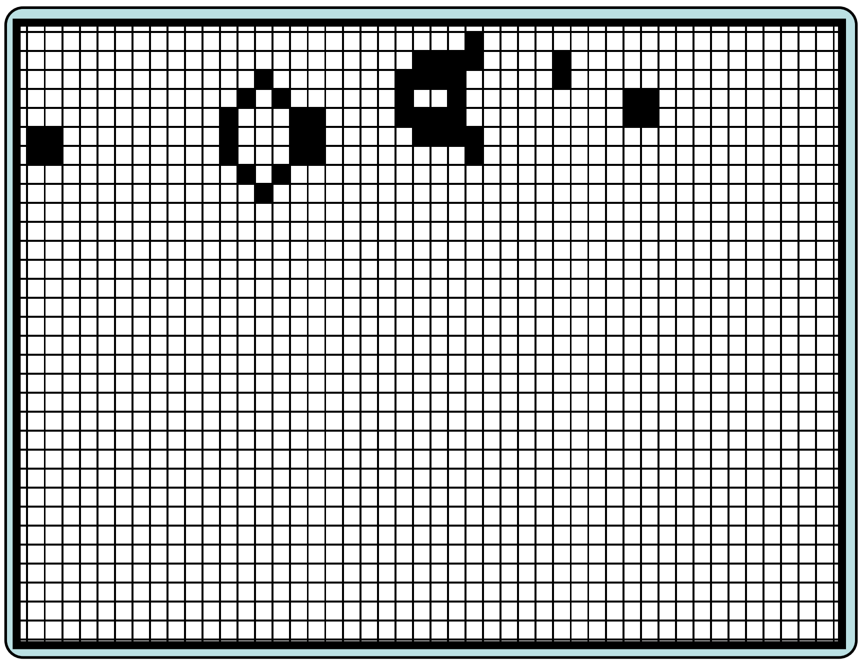


Glider

Delay



Glider gun

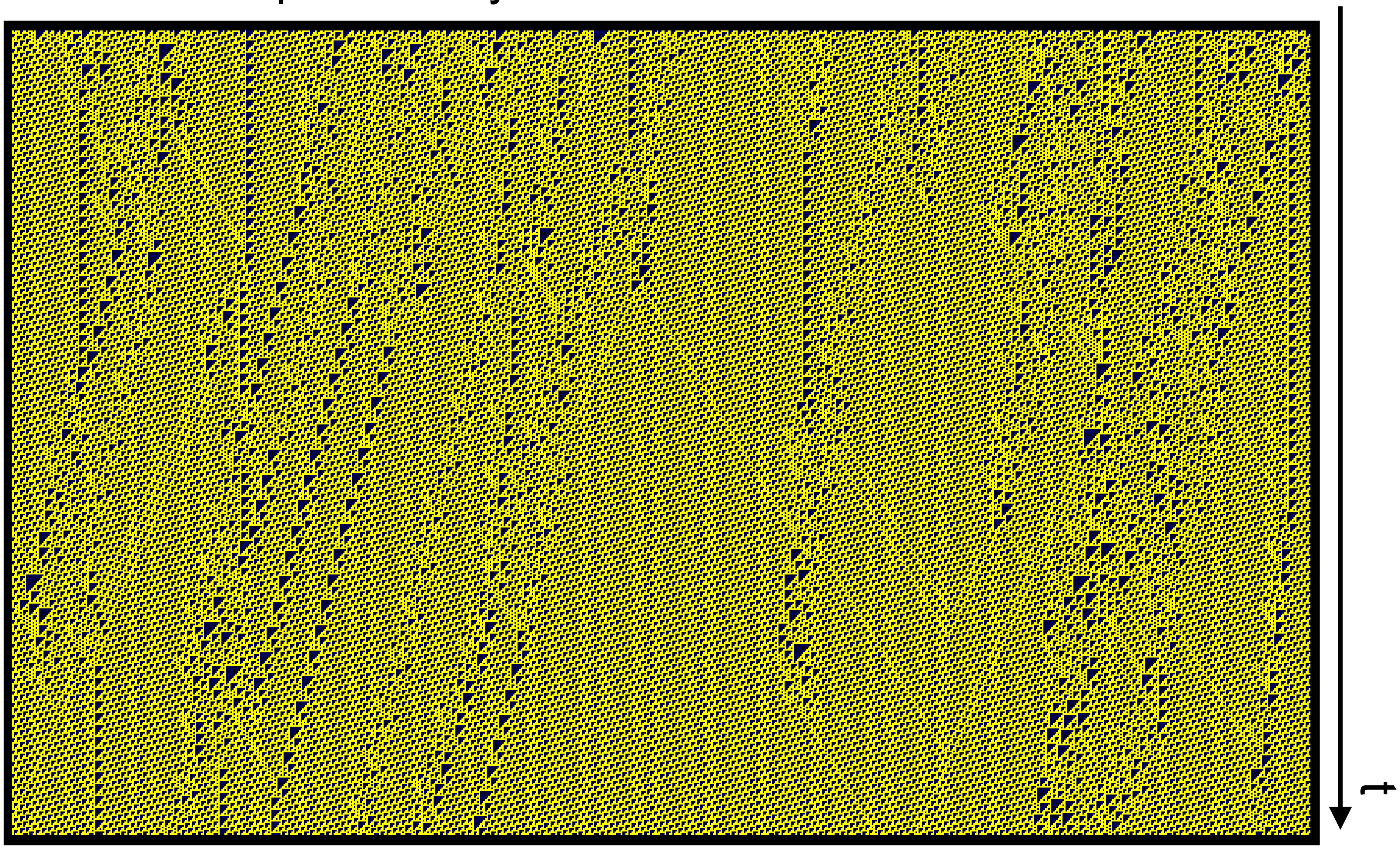


Computational Universality

- In *Life* we can define signals (as streams of gliders interpreted as bits), implement all logic gates (AND, NOT,...), implement delays, memory banks, signal duplicators, and so on.
- Hence, *Life* can emulate any computing machine; we say that it is capable of **universal computation**.
- The theory of computation says that, in general, given an initial state for the automaton, there is no short-cut way to predict the result of *Life*'s evolution. We must run it.
- We say that *Life* is **computationally irreducible**.
- In simple words, this means that a very simple CA such as *Life* (and Rule 110 in 1D) can produce highly nontrivial behaviors, that cannot be predicted simply by observing the transition rule.
- The "universe" constituted by a CA can be an interesting backcloth for the emergence of complex phenomena.

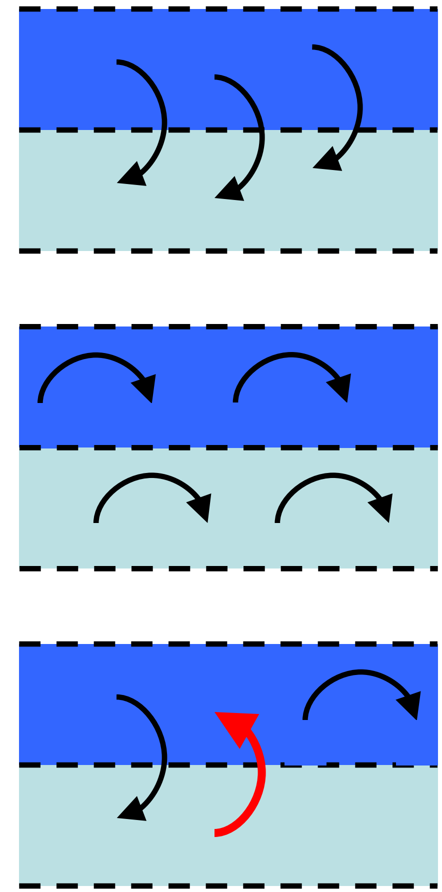
Universality in 1D CA

CA even simpler than *Life* display the same properties.
Rule 110 is computationally universal.



The Growth of Complexity

- Usually a machine produces machines les complex than itself: can we prove formally that there exist machines which can produce more complex machines?
- von Neumann's approach:
  - A machine capable of self-reproduction would produce machines of equal complexity
  - If the self-reproduction process could tolerate some "error" (*robust* self-reproduction) then some of the resulting machines might have greater complexity than the original one

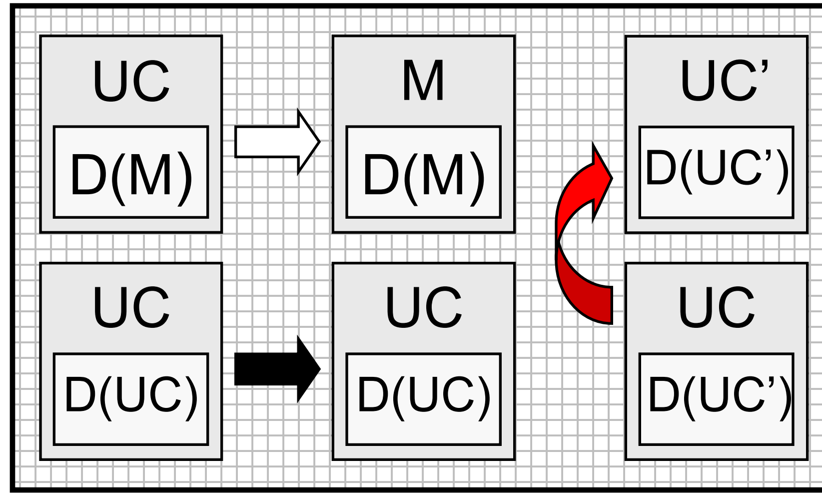


Self-Reproducing Automata

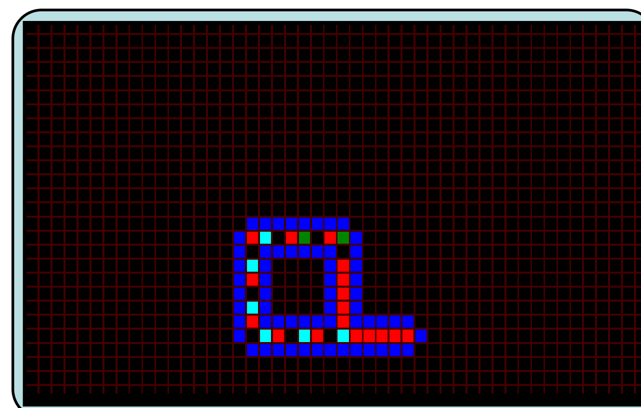
von Neumann solved his problem by defining an automaton composed by a **universal constructor** UC and a description D(M) of the machine to be generated.

von Neumann automata is quite complex (29 states per cell, and about 200.000 active cells)

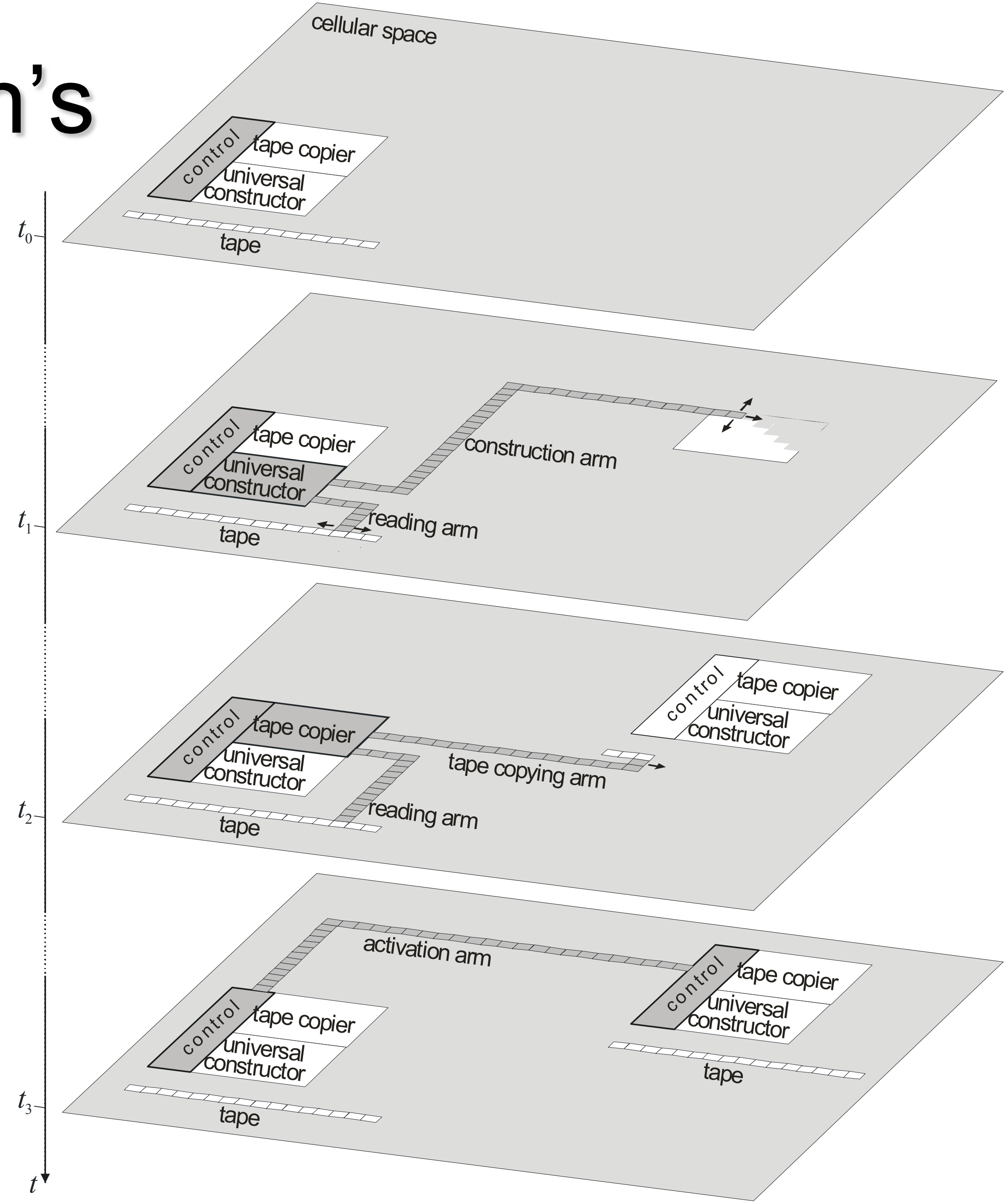
Other scientists focused on the issue of self-reproduction and offered simpler solutions to this sub-problem (trivial self-reproduction)

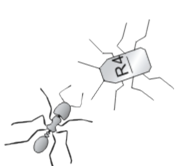


Example: Langton’s Loop



von Neumann's Automaton

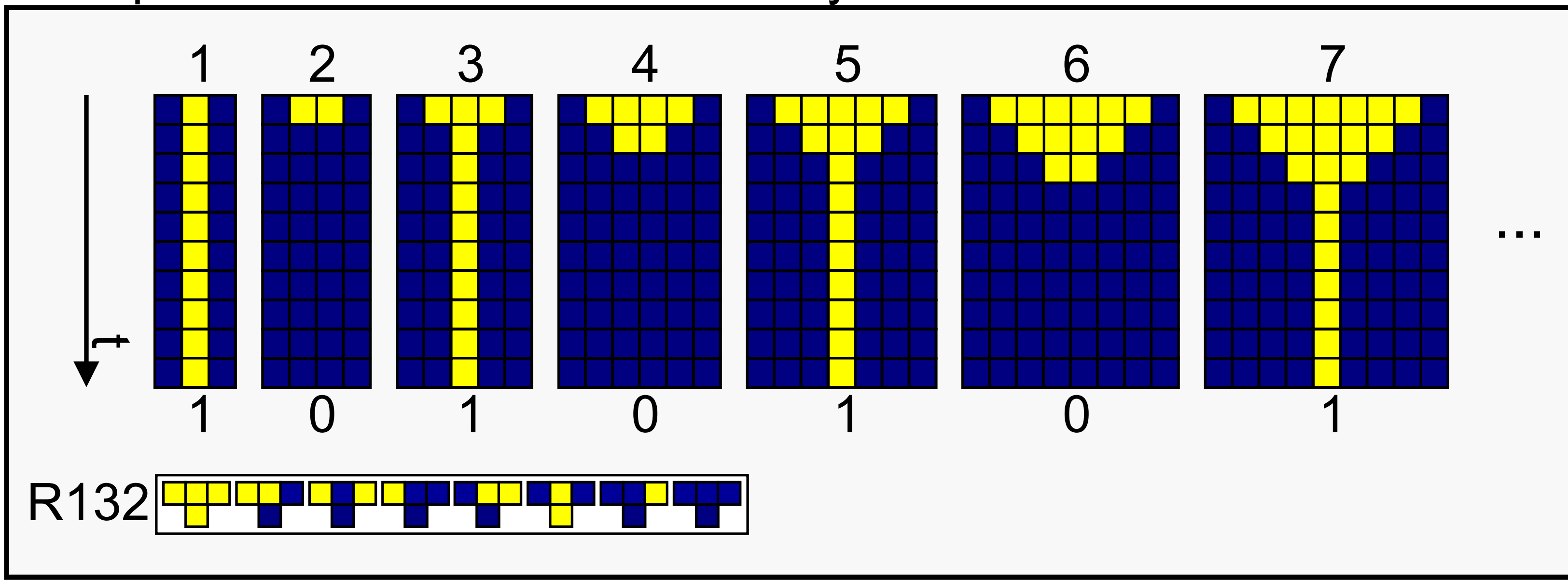




Computation with CA

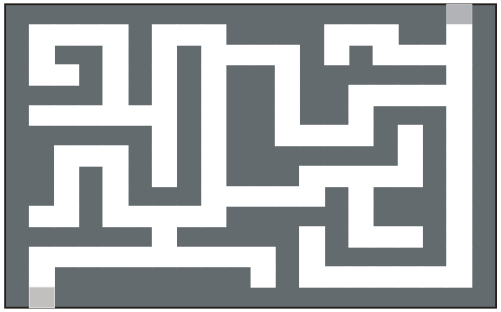
CA used as input-output devices. The initial state is the input. The CA should go to a quiescent state (fixed point), which is the output.

Example: Remainder after division by 2

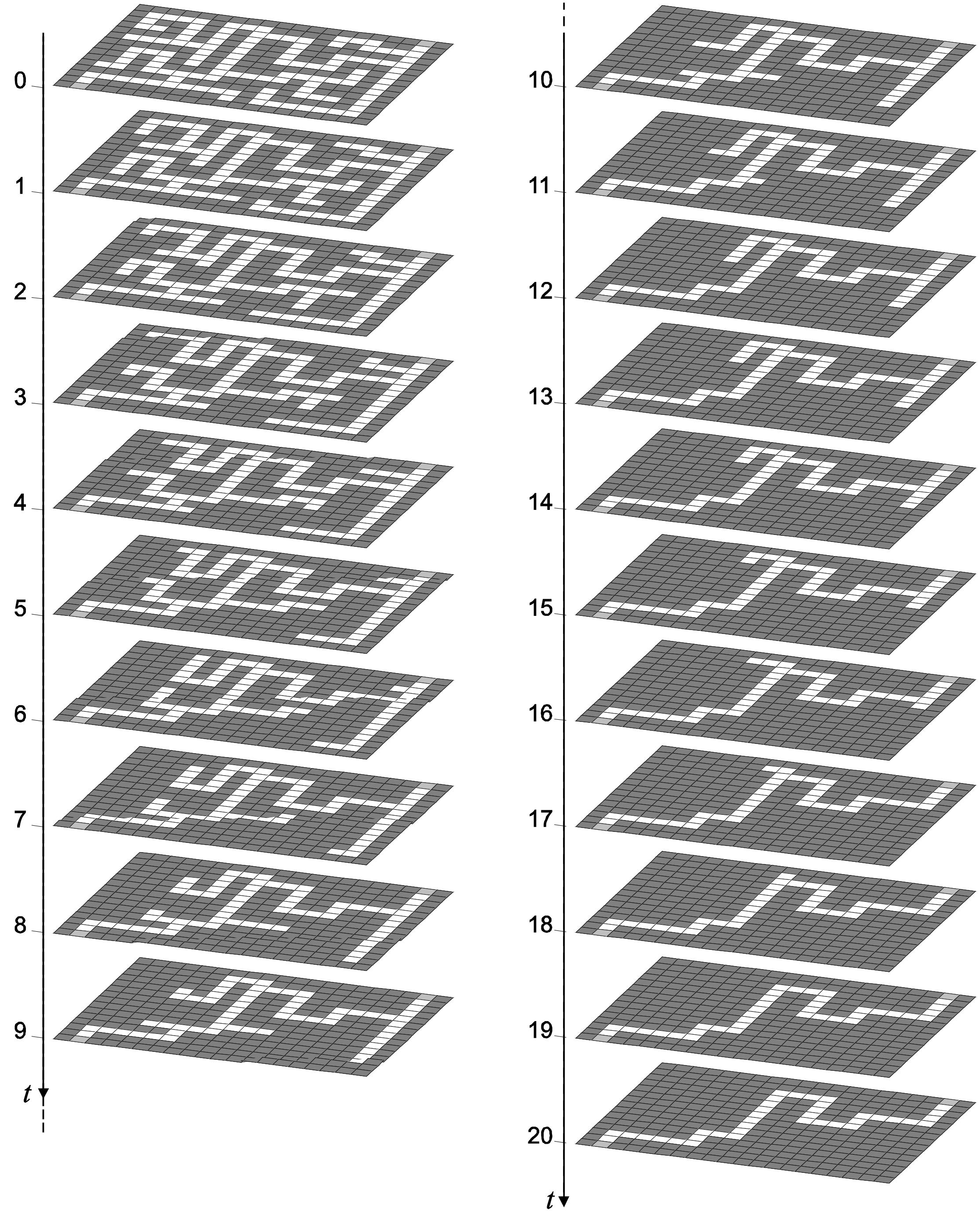


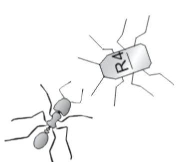
The difficulty stems from the fact that we use a local rule to evaluate a property that depends on information distributed globally.

Example: CA maze solver



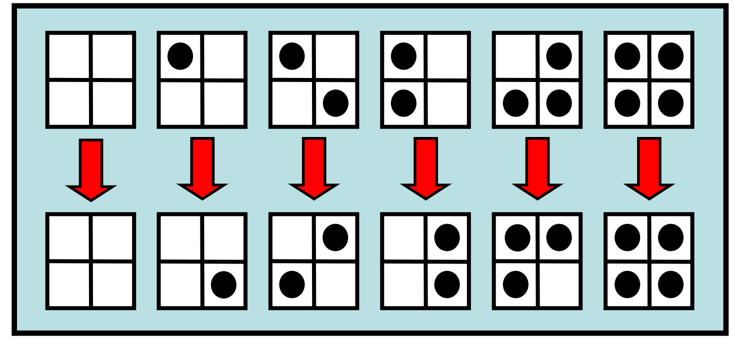
- Given a maze the problem consists in finding a path from the entrance to the exit.
- The conventional approach marks blind alleys sequentially
- The CA solver removes blind alleys in parallel



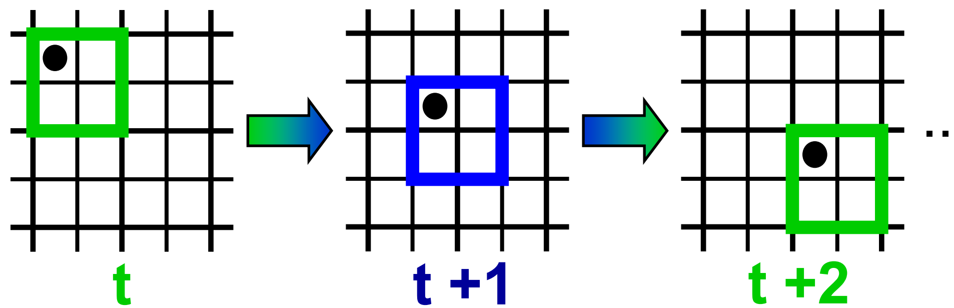


Particle CA

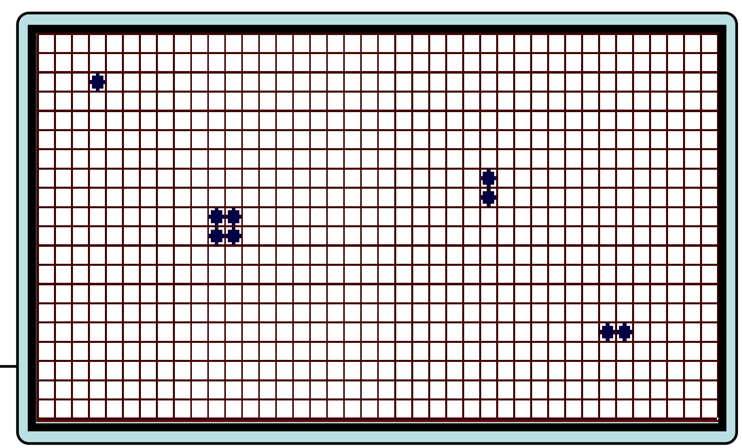
CA can be used to model phenomena that involve particles. The transition rule can be specified in terms of the motion of particles within **blocks** of two by two cells (block rules).



The automaton space is **partitioned** in non-overlapping blocks



To allow the propagation of information the position of the blocks alternates between an odd and an even partition of the space (Margolus neighborhood).

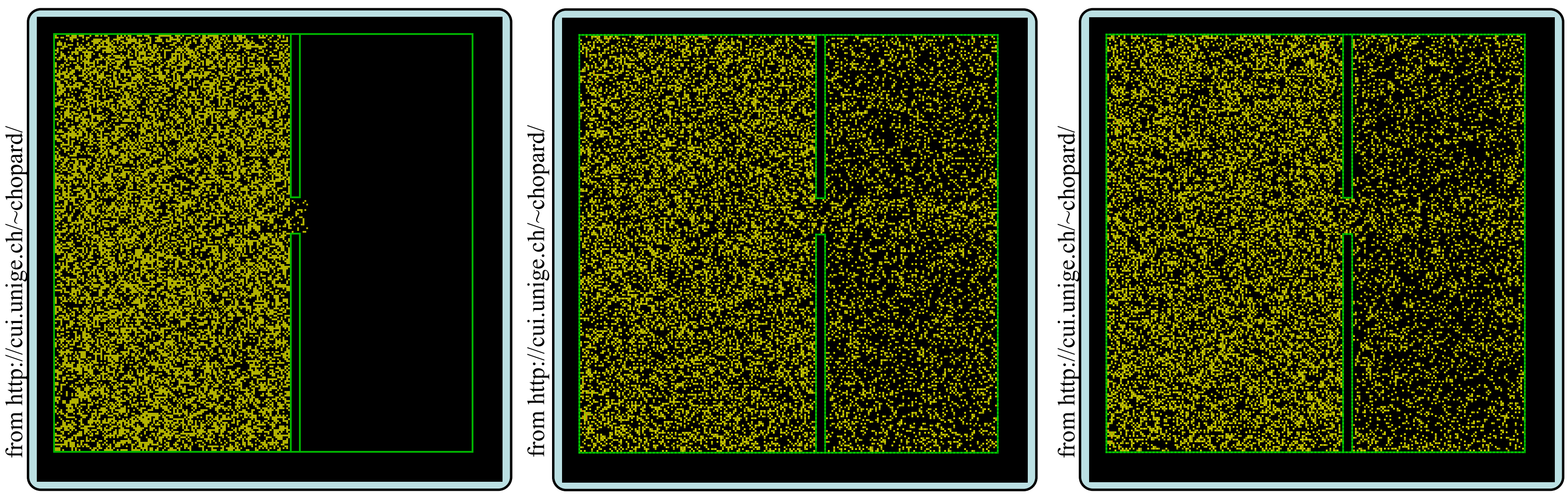


Reversibility

One of the interesting properties of CA is the possibility to display **exact reversibility**. Contrary to conventional numerical simulations, CA are not plagued by approximation errors.

At the microscopic level the laws of physics are assumed as being reversible. A particle CA can display invariance under time reversal. This means that no information is lost during the evolution of the CA. We can therefore observe very subtle effects.

Forward Backward Backward with "error"



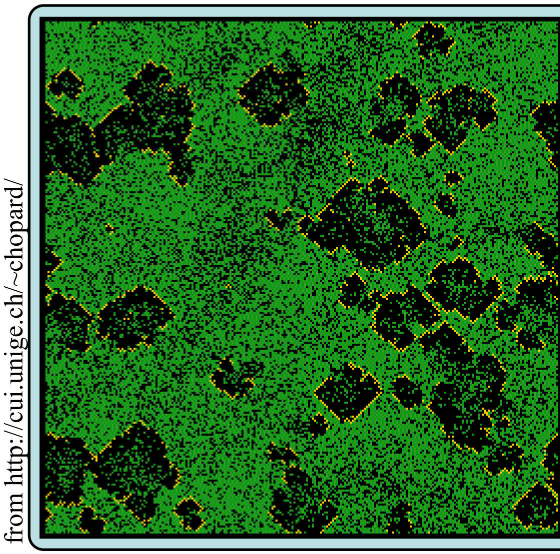
Probabilistic CA

So far we have considered only **deterministic** CA.

To model many phenomena it is useful to transition rules that depending on some externally assigned probability

Example: The **forest fire model**

- Each cell contains a green tree , a burning tree , or is empty
  - A burning tree becomes an empty cell
  - A green tree with at least a burning neighbor becomes a burning tree
  - A green tree without burning neighbors becomes a burning tree with probability *f* (probability of lightning)
  - An empty cell grows a green tree with probability *g* (probability of growth)



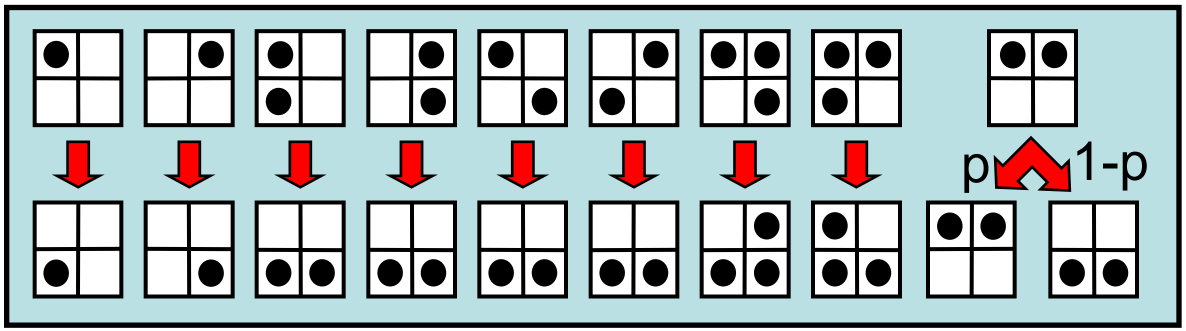
from http://cui.unige.ch/~chopard/

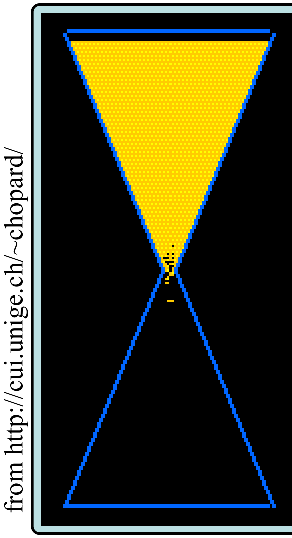
The parameters can be varied in a continuous range and introduce some “continuity” in the discrete world of CA models

Complex Systems

Cellular systems allow the modeling and simulation of phenomena that are difficult to describe with conventional mathematical techniques

Example: The sand rule with friction





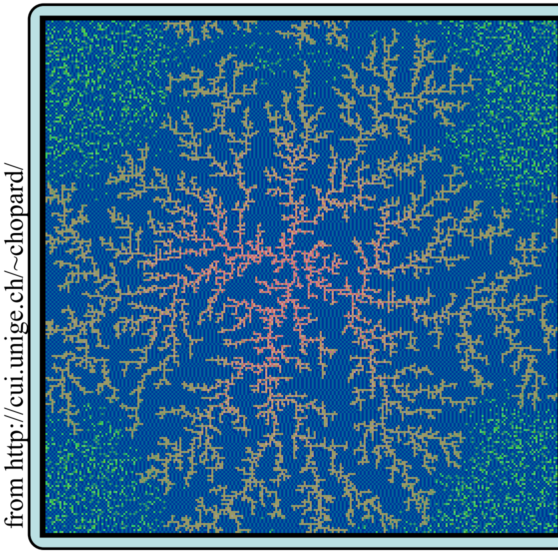
from http://cui.unige.ch/~chopard/

This kind of model permits the exploration of the behavior of *granular media*, which is difficult with conventional tools (e.g., PDEs)

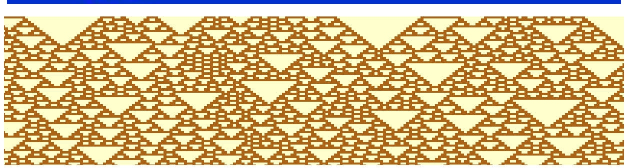
Structures and Patterns

One of the most fascinating aspects of biological and natural systems is the emergence of complex spatial and temporal structures and patterns from simple physical laws and interactions.

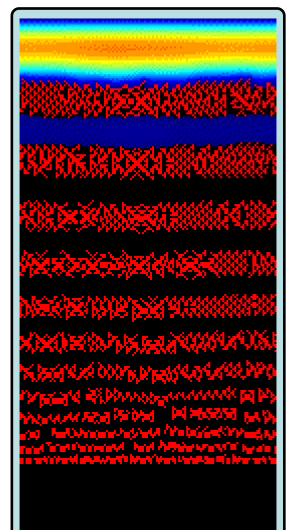
Cellular systems are an ideal tool for the analysis of the hypotheses about the local mechanisms of structure and pattern formation.

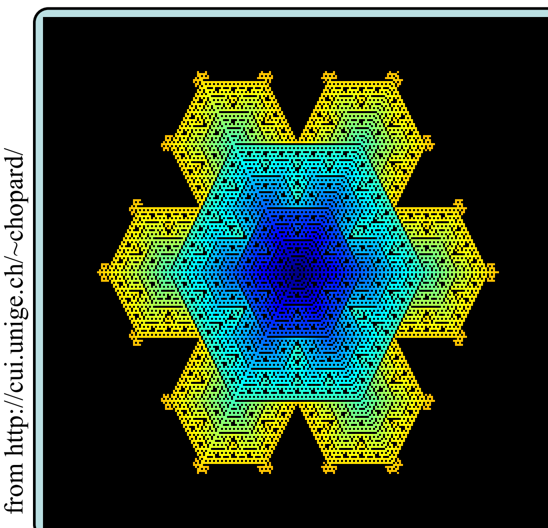


from http://cui.unige.ch/~chopard/



from http://www.btinternet.com





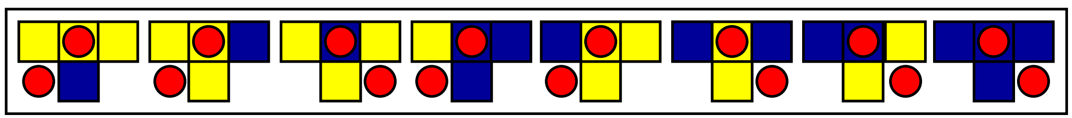
from http://cui.unige.ch/~chopard/

Variants and Extensions

The basic CA is discrete in space, time and state; updates all its cells synchronously; uses the same neighborhood geometry and transition rule for all cells.

We can relinquish some of these prescriptions and obtain:

- **Asynchronous** CA (for example, *mobile automata*, where only one cell is active at each time step, and the transition rule specifies the fate of the activation)



- **Non-homogenous** (or non-uniform) CA
- Continuous-state CA (**Coupled Map Lattices**)
- Continuous-state and time CA (**Cellular Neural Networks**)
- ...

Analysis and Synthesis

Both analysis and synthesis of cellular systems are usually difficult problems. The problem is once again the fact that the link between the local rules and the global behavior is not obvious. A number of different techniques are used.

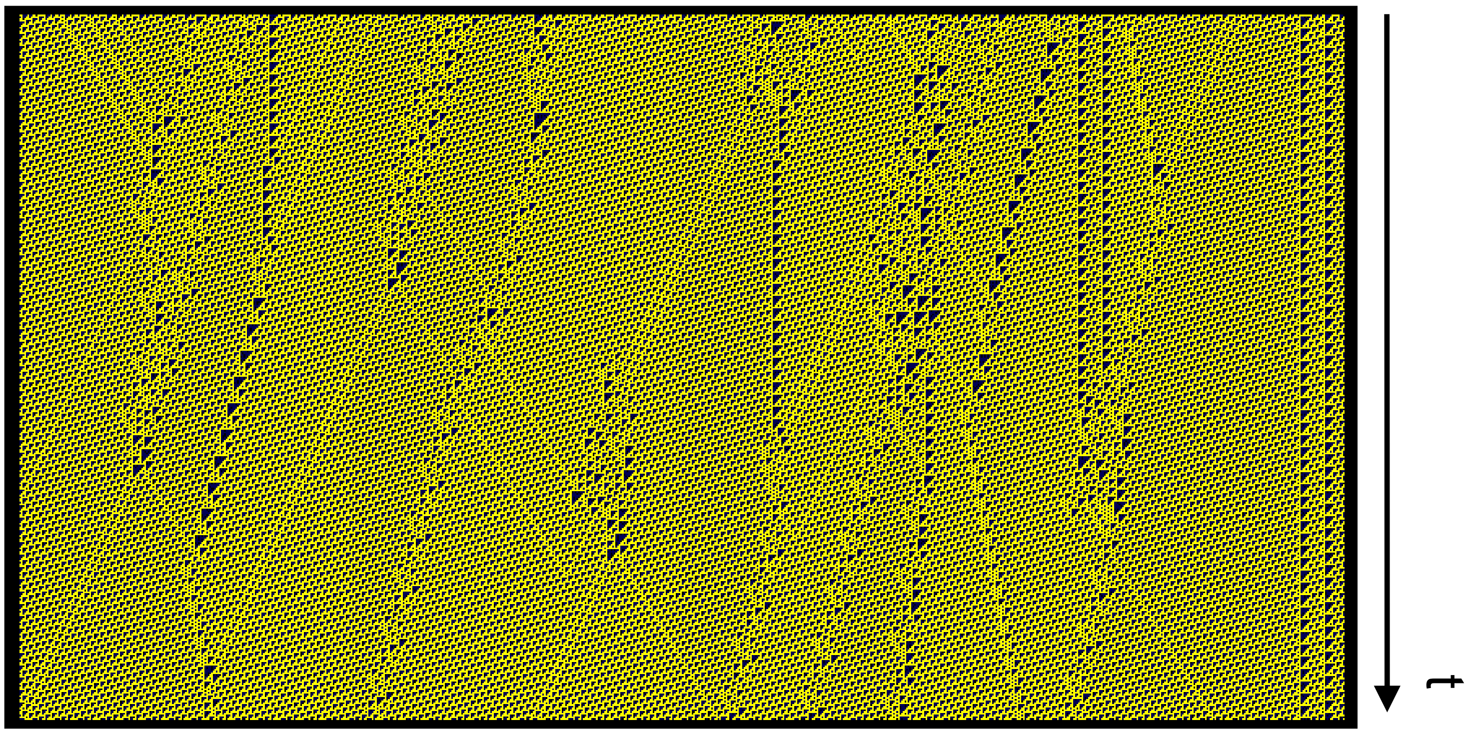
Analysis: Phenomenological approach; Dynamical System Theory (attractors, cycles...); Analytic approach (global mapping, algebraic properties,...). Statistical Mechanics concepts; Probabilistic approach...

Synthesis: By hand (e.g., *Life*'s zoo); Based on some idea about a possible underlying "microscopic" process...

Due to the absence of general principles to rules producing a desired global behavior, the synthesis of cellular systems is a field particularly suited to the application of **Evolutionary Techniques**.

(Wild) speculations about CA

The universe as a CA? (for example, R110 is computationally universal; moving structures of R110 could be interpreted as "particles" within a 1D "universe"; the underlying simple rule is difficult to derive from observation)



There are many difficulties in developing a convincing cellular model of fundamental physical laws (synchrony, anisotropy, space-time invariance ...)

Cellular Systems Summary

We have only scratched the surface of the cellular systems world. However, we have seen that cellular systems can used be at least as:

- Synthetic universes creators in Evolutionary and Artificial Life experiments
- Models and simulators of simple and complex, biological, natural, and physical systems and phenomena
- Computation engines
- Testers of hypotheses about emergent physical and computational global properties and the nature of the underlying local mechanisms