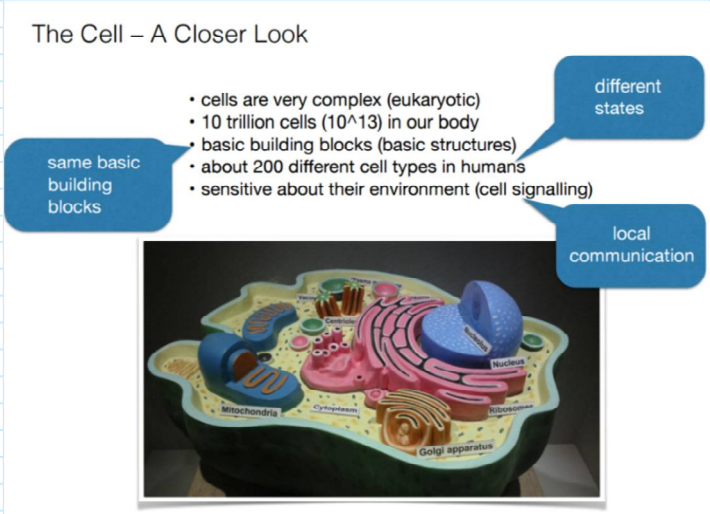
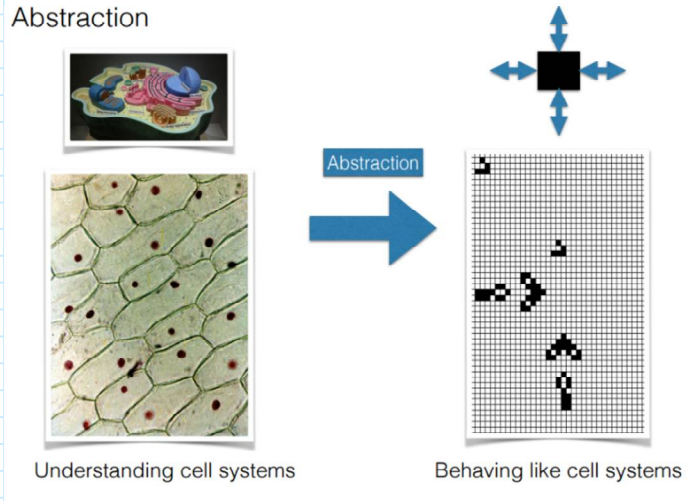
Cellular Systems

22 May 2017 02:14

Key idea: simple local rules between discreet cells can lead to complex global structure and/or behaviour.

Biological Cells:





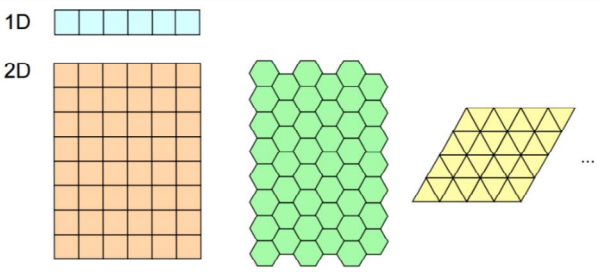
Self-replicating systems:

- J. von Neumann's universal constructor.
- Wolfram's elementary cellular automata (CA)

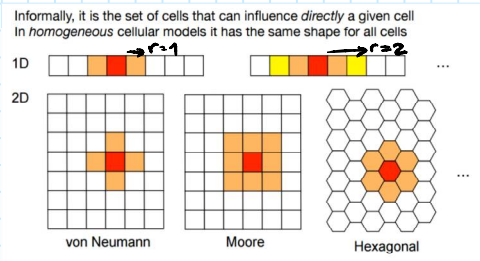
Building blocks of cellular system:

- Virtual space
- Spatial interaction
- Cell state
- Transition rules.

Virtual Space



Spatial interaction

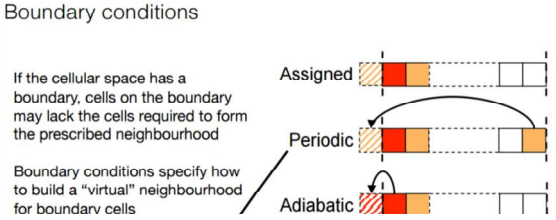


Transition rule

Each cell has a state, often represented a a number. The value of the state is often represented by cell colours. There can be a special **quiescent state** s_0.

$$S = \{s_0, \dots, s_{k-1}\} = \{0, \dots, k-1\} = \{\bullet, \dots, \bullet\}$$

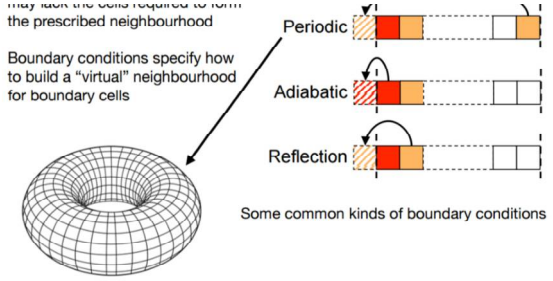
The **transition rule** is the fundamental



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 neighbourhood.

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

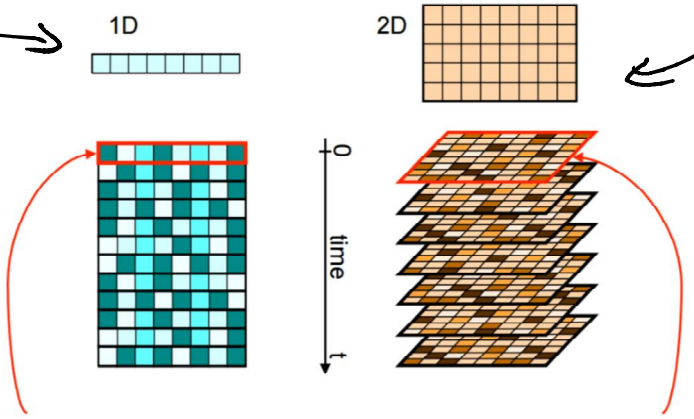
transition func
$\delta : State \times [State] \rightarrow State$
at each time step,
$s_{t+1}(i) = \delta(s_t(i), ns)$
where ns = n neighbors



Initial conditions

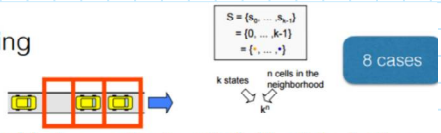
image where y-axis shows development

video of simulation

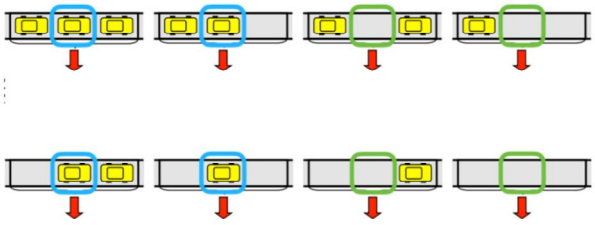


In order to start with the updating of the cells of the CA we must specify the initial state of the cells (initial conditions or seed)

Example: Traffic Modeling



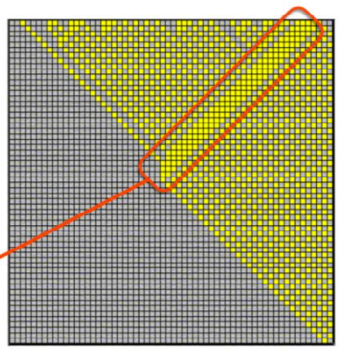
We construct an elementary model of car motion in a single lane, based only on the **local** traffic conditions. The cars advance at **discrete time steps** and at **discrete space intervals**. A car can advance (and must advance) only if the destination interval is free.

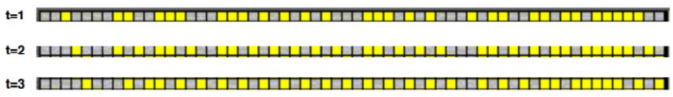


Example: Traffic Jam

Start 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).





Key idea: δ moves cars forward, where they can't move forward if they are blocked

Doesn't account for drivers slowing down before upcoming traffic, and other more complex decisions.

The general rules for designing CA:

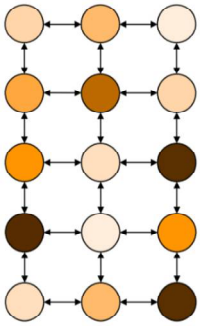
1. Assign CA space geometry
2. Neighborhood geometry
3. Define cell states

2. Neighborhood geometry
3. Define cell states
4. Define δ
5. Assign initial conditions
6. Define boundary conditions
7. Repeatedly update global cells by applying δ.

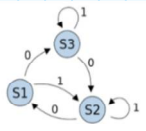
Informal definition of CA

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")



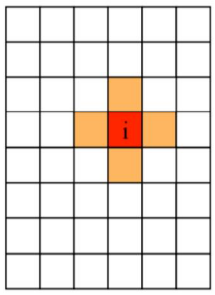
Formal definition of CA



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 **neighbourhood** *N* of the machine, and whose geometry is the same for all machines

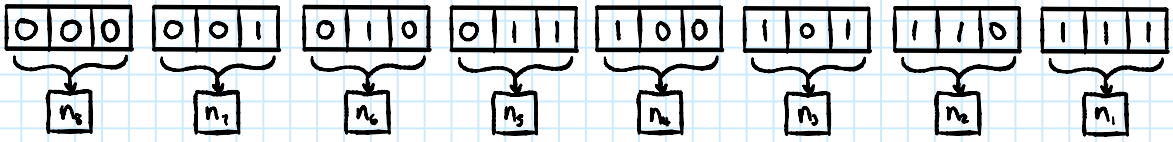
$$s_i(t+1) = \phi(s_j(t) ; j \in N_i)$$



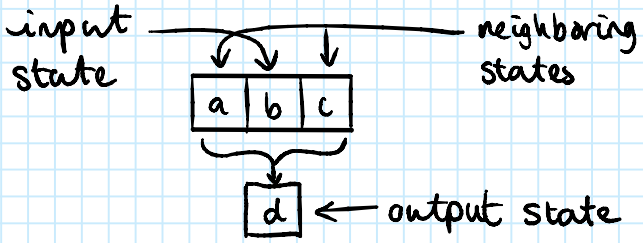
Wolfram's Rule Code for Elementary CA

Elementary CAs are 1D binary CAs $k=2$ with minimal range, $r=1$

CAs are index with Wolfram's code, Rule n is constructed by taking n's base-2 representation, $n = \langle n_1, n_2 \dots, n_8 \rangle$, $n_i \in \{0,1\}$, assuming $0 < n < 256$, the transition rule is defined



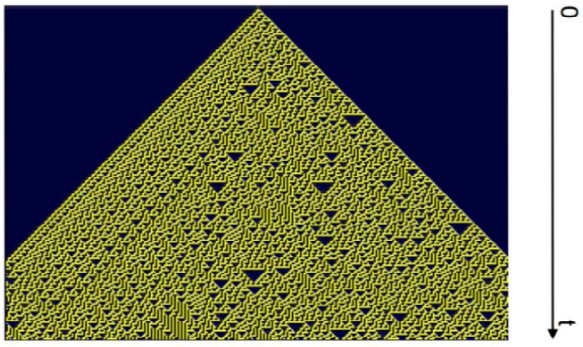
where,



Example of elementary CA

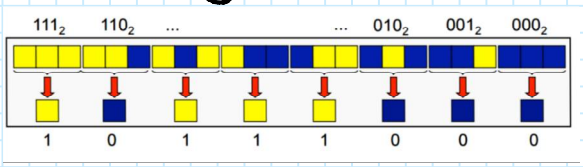
Rule 30 is used by Mathematica as its Random Number Generator (RNG are ubiquitous in bio-inspired experiments).

1, 3, 3, 6, 4, 9, 5, 12, 7, 12, 11, 14, 12, 19, 13, 22, 15, 19, ...



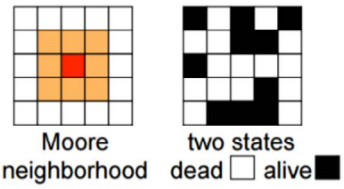
R30 

Traffic rule (R187)



Game of Life (GoL)

Classical 2D CA: Life



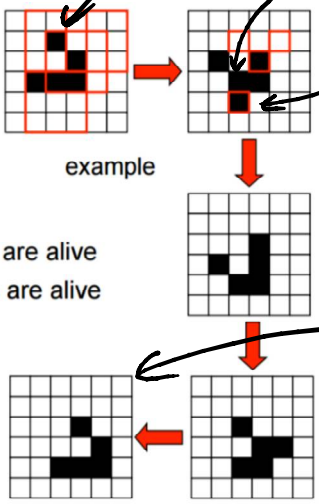
Outer totalistic rule (John Conway)

- **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")



cell dies of isolation

dies of overcrowding

birth

back to original

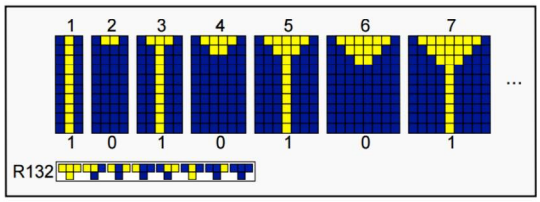
- GoL is Turing complete
- it is computationally irreducible and universal

More examples:

Computation with CA

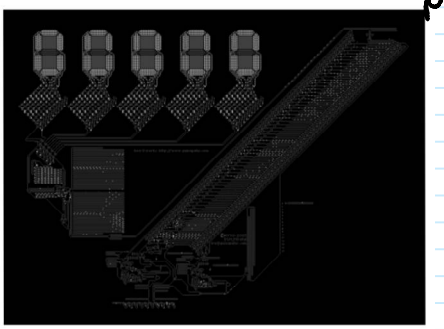
CA used as input-output devices. The initial state is the input. The CA should go to a quiescent state (fixed point), which is the output.

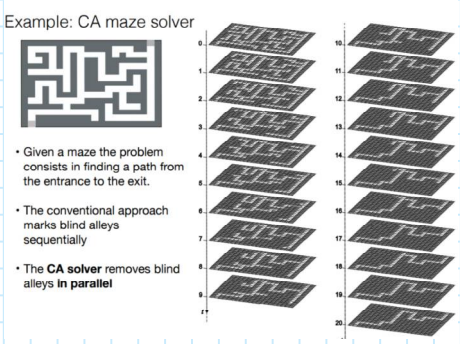
Example: Remainder after division by 2

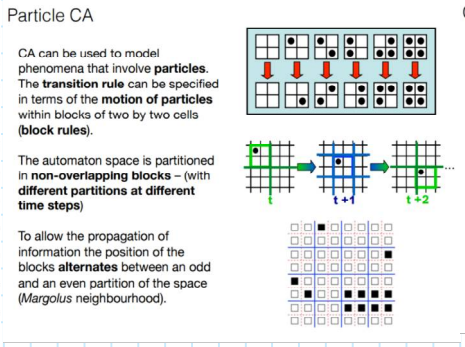


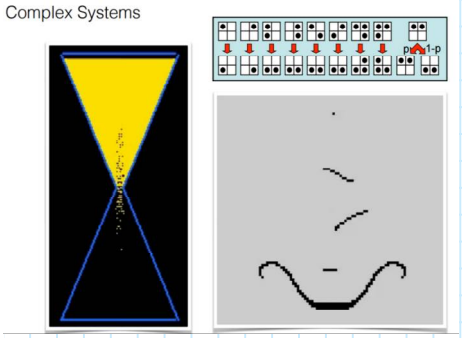
The Wireworld computer

computing primes









Reversibility:

Some CAs are reversible - that is given some global state A it is possible to find some other configuration B such that $A = \delta(B)$ (applying the transition rule globally).
Beyond the trivial cases (R0, R8, R64, R128), reversibility is achieved if the CA conserves state information. For example the traffic rule (R184) is reversible because car "mass" is conserved.

Variants, Extensions