Cellular Automata

Guillaume Hutzler
IBISC (Informatique Biologie Intégrative et Systèmes Complexes)
COSMO team (COmmunications Spécifications MOdèles)
guillaume.hutzler@ibisc.univ-evry.fr
http://www.ibisc.univ-evry.fr/~hutzler/Cours/

Course outline

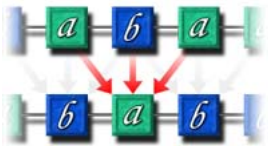
- Introduction
  - Simple Examples
  - Historical perspective
- Design choices
- Theoretical considerations
- Application to the modelling and simulation of complex systems
- Conclusion

1-D automaton

- 1-dimensional model
  - Linear array of cells
- Limited state space
  - Each cell takes its values from the {0, 1} set
- Reduced neighbourhood
  - The cells neighbourhood is limited to the two adjacent cells
- Simple transition rules
  - The automaton progresses through successive generations
  - The state of a cell to the next generation depends on its state and on the state of its neighbors at the current generation
  - All cells change state synchronously
- Studied by S. Wolfram in a systematic way

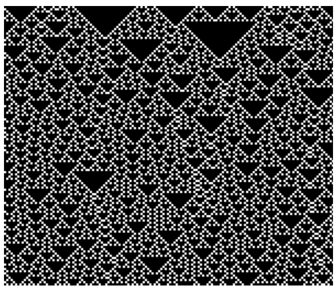
Transition rules

- Base principle



- For a given cell
  - 8 possible configurations (2^3)
  - 2 new possible states for each configuration
  - 256 possible automata (2^8)
  - ex. : rule 18
    - f(111)=0 – f(110)=0
    - f(101)=0 – f(100)=1
    - f(011)=0 – f(010)=0
    - f(001)=1 – f(000)=0

State-space diagram



n° rule = f(111)f(110)f(101)f(100)f(011)f(010)f(001)f(000)

A model of morphogenesis?

- Observation
  - The patterns obtained by some automata resemble that exhibited by some sea-shells
  - May model a simple diffusion model [Meinhardt & Klingler 1987]
    - « Shell patterns are time records of a one-dimensional pattern forming process along the growing edge. Oblique lines result from travelling waves of activation (pigment production). Branches and crossing result from a temporary shift from an oscillatory into a steady mode of pigment production... »

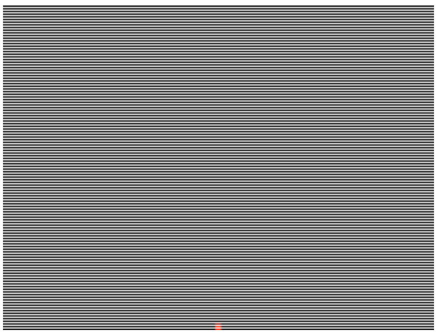
General considerations

- Very experimental domain
  - an automaton is fully described by its specification
  - **But**: impossible to predict a priori the state of an automaton without executing
- But also theoretical results
  - Complexity classes
  - reversibility
  - Eden Gardens and limit-sets
  - Conservation laws
  - universality
- applications
  - simulation of spatio-temporal phenomena (physics, chemistry, biology as well as engineering, traffic, sociology, etc.).
  - image processing and classification

Historical background (1)

- Stanislaw Ulam
  - was interested in the evolution of graphic constructions created from simple rules
  - principle
    - Two-dimensional space divided into "cells" (a kind of graph paper)
    - Each cell can have two states: on or off
    - Starting from a given configuration, the next generation was determined by neighbourhood rules
      - eg. if a given cell is in contact with two on-cells, it goes on, else it goes off
  - results
    - generation of complex and aesthetic figures
    - in some cases, these figures could replicate
  - Questions
    - may these recursive mechanisms explain the complexity of reality?
    - is this complexity only apparent, the fundamental laws themselves being single

Maltese cross (1)

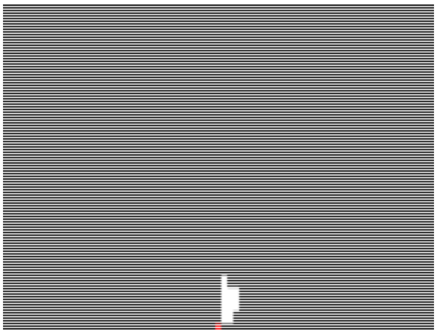


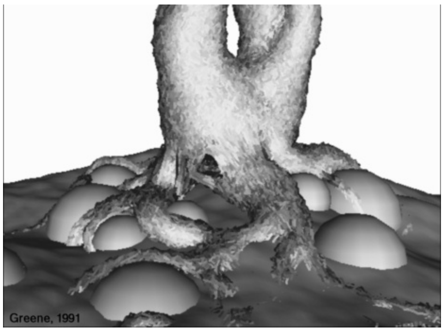
http://algorithmicbotany.org/vmm-deluxe/QT/Maltese/maltese.qt

Introduction - S. Ulam

Maltese cross (2)

http://algorithmicbotany.org/vmm-deluxe/JPEG/Maltese/maltese_obst.jpg





[Greene 1991]

Introduction

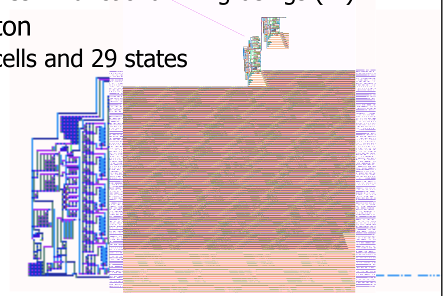
Historical background (2)

- John von Neumann
  - Worked on the design of a self-replicating machine design, the *kinematon*
    - Capable of producing any machine described in its program, including a copy of itself from materials found in the environment
  - difficulties
    - self-reference in the description
      - The machine should have a description of itself, so also a description of the description ...
      - The description is seen as both a program and a component
        - the description is interpreted to build the new machine
        - it is then copied
      - Similar to the operation of the DNA (found out later)
    - Physical conditions of realization of the machine

Introduction

Historical background (3)

- Self-replicating automata
  - Ulam suggested that von Neumann use what he called the "cellular spaces" (cellular spaces) to build his machine
    - "By axiomatizing [self-replicating] automata this way, one (...) has resigned to not explain how these elements are made of real things, particularly how these elements are made up of elementary particles or even molecules (...) we will simply assume that elementary particles with certain properties exist. The question we hope to answer, or at least consider is: what principles are implemented in the organization of these molecules in functional living beings (...) "
  - each cell is a finite state automaton
    - 2-dimensional CA with 200,000 cells and 29 states
      - signal transmission
      - logical operations
    - Logical Architecture
      - a universal constructor
      - a strip of cells



Introduction – historical background

Later developments

- Developments in different directions
  - Complement the work of Von Neumann cellular automata [Burks 70]
  - Extending the work of Von Neumann on self-replicating machine [Codd 68 Langton 84]
  - Games based on cellular automata
    - Game of Life [Conway 70]
    - Brian's brain [Silverman 84]
  - Theoretical study of the properties of cellular automata
  - Extension of the original model (coupled map lattices)
  - Application to the modeling of complex systems (biology, physics, sociology, etc.).

Introduction - automates auto-reproducteurs

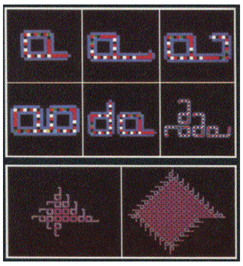
Self-replicating automata

- Edgar Codd (1968)
  - Simplified version of the Von Neumann automaton
    - only 8 states
    - still a universal constructor
- Christopher Langton (1984)
  - abandoned the idea of universal replicator
  - designing a cellular automaton supporting a structure whose components are the information needed for its own replication
  - structure both itself and representation of itself
  - uses 8 states and 29 rules
  - loop consisting in a "membrane" in which circulates the information necessary for replication

Introduction – self-replicating automata

The Langton loop

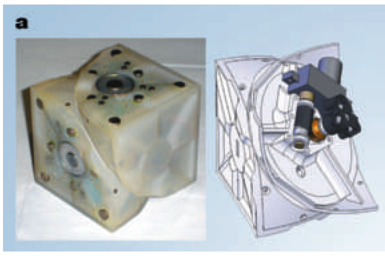
	2	2	2	2	2	2	2	2					
2	1	7	0	1	4	0	1	4	2				
2	0	2	2	2	2	2	2	0	2				
2	7	2					2	1	2				
2	1	2					2	1	2				
2	0	2					2	1	2				
2	7	2					2	1	2				
2	1	2	2	2	2	2	2	1	2	2	2	2	2
2	0	7	1	0	7	1	0	7	1	1	1	1	1
	2	2	2	2	2	2	2	2	2	2	2	2	2



- cells in state 2 form the membrane
- internal cells contain the replication information (kind of DNA)
- 7-0 and 4-0 sequences spread towards the tail
  - 7-0 sequences extend the tail
  - 4-0 sequences construct a right angle to the left
- a rule of "sterilization" blocks changes after a number of generations and allows the crystallization of the oldest loops

Introduction – self-replicating automata

A robotic automaton



http://mae2.wdg.us/ccsl/research/selfrep/

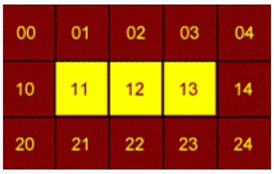
Introduction – Games based on cellular automata

The Game of Life [Conway 1970]

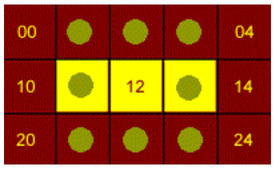
- Originally presented as a mathematical game
  - a rectangular grid of cells
  - each cell can either be « alive » or « dead »
  - the state of the cells is randomly initialized
  - at time *t+1*, the state of each cell depends on its own state and on the state of its 8 neighbours at time *t*
    - a dead cell becomes alive if it has exactly three live neighbours (reproduction)
    - a live cell dies if it has
      - less than 2 live neighbours (isolation)
      - more than 3 live neighbours (overcrowding)
- Result
  - emergence of dynamical structures
  - variant : different rules for the evolution of the cells

Introduction – The game of life

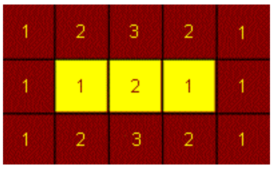
A simple example



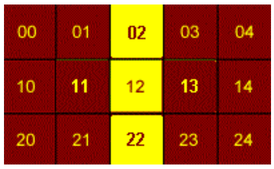
Numbered cells
(alive in yellow, dead in red)



Neighbourhood of cell n°12



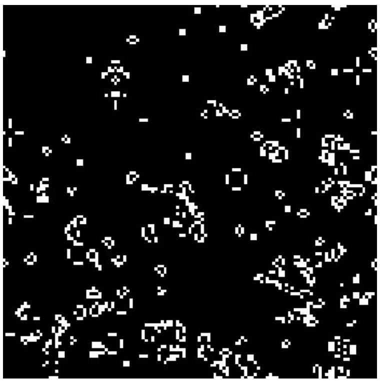
Number of live nieghbours



State of the automaton at the
next generation

Introduction – The game of life

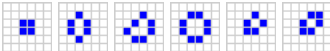
Example of dynamics



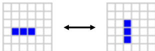
Introduction – The game of life

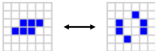
Remarkable structures

- Stable structures (1-periodic)

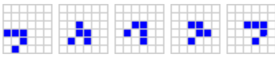


- Oscillating structures (2-periodic)





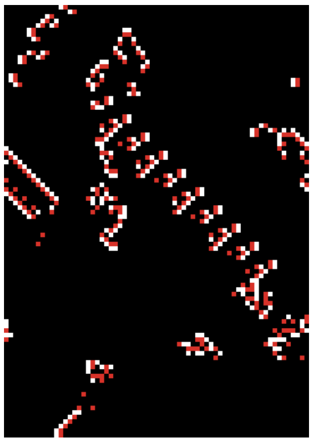
- N-periodic with translation (e.g. glider)



Introduction – Games based on cellular automata

Brian's brain [Silverman 84]

- 3 states instead of 2
  - excited (white)
  - refractory (red)
  - dead (black)
- Transition rules
  - excited cells go refractory at next timestep
  - refractory cells go dead at next timestep
  - a dead cell becomes excited if it has exactly two excited neighbours (among its 8 neighbours)



Cellular automata characterization

Informal description

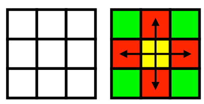
- Framework for a large class of discrete models with homogeneous interactions
- These models have the following properties:
  - Discretization
    - space: decomposed into a grid cell space
    - time: evolution through discrete timesteps
  - Parallelism
    - cells evolve simultaneously and independently
  - Locality
    - each cell evolves according to its own state and that of a finite set of neighbouring cells
  - Homogeneity
    - the topology is regular cells
    - the neighbouring relationship is uniform
    - transition rules are the same for all cells

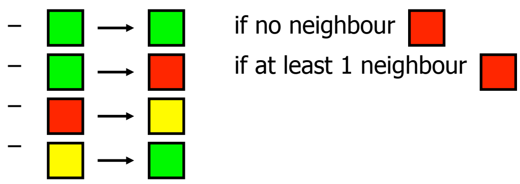
Cellular automata characterization

Simple example : Greenberg-Hastings

- Model of excitable medium
  - resting state (0)
  - excitation state (2)
  - refractory state (ou remission)

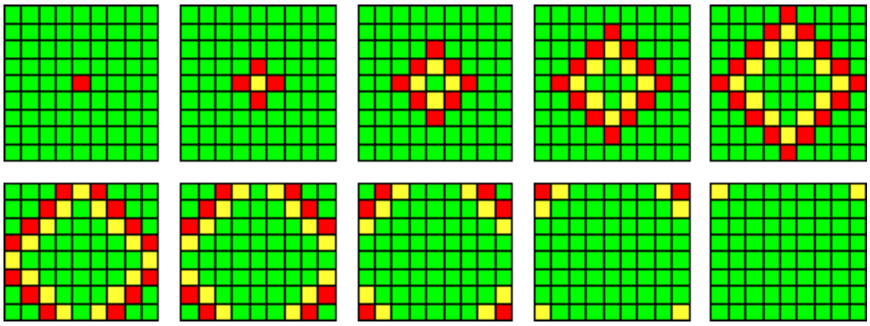
- Parameters of the automaton
  - rectangular grid
  - 4-neighbourhood
  - rules





Cellular automata characterization

Evolution with one excited cell



Cellular automata characterization

Formal definition

- Let
  - *L* a regular grid (its elements are cells)
  - *S* a finite number of states
  - *N* a finite number of neighbouring indexes (of size *n*) so that :

    $\forall c \in \mathcal{N}, \forall r \in \mathcal{L} : r + c \in \mathcal{L}$
  - *f* a transition function : $f : S^n \rightarrow S$
- A cellular automaton is defined by the 4-tuple $(\mathcal{L}, S, \mathcal{N}, f)$
- A configuration $C_t : \mathcal{L} \rightarrow S$ is a function which associates a state to each cell of the grid
- The role of the transition function *f* is to change C_t into C_{t+1} according to : $C_{t+1}(r) = f(\{C_t(i) \mid i \in \mathcal{N}(r)\}),$
  - where $\mathcal{N}(r)$ is the set of the neighbours of cell *r*

    $\mathcal{N}(r) = \{i \in \mathcal{L} \mid r - i \in \mathcal{N}\}.$

Design choices

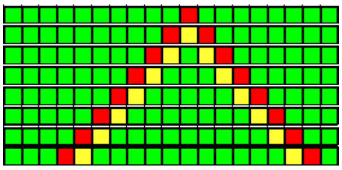
- Space dimension and lattice geometry
- Shape and size of the neighbourhood
- Boundary conditions
- Initial conditions
- State space
- Transition rules

Grid geometry

- Regular grid = periodic tiling of a n-dimensional space
  - cells entirely tile a d-dimensional space
  - the grid reproduces identically by translations in *d* independant directions
- Solutions in 1, 2, 3 dimensions

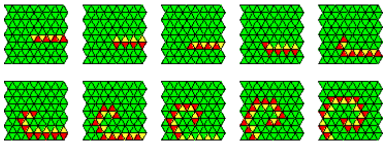
1-dimensional space

- Only one possibility
  - linear array of cells

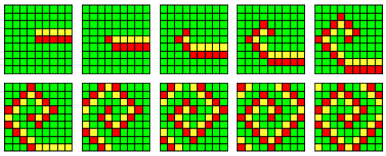


2-dimensional space (1)

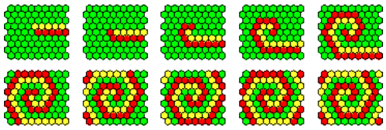
- 3 regular grids
  - triangular



  - square



  - hexagonal



Cellular automata characterization – Design choices

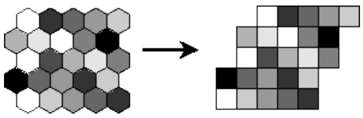
2-dimensional space (2)

- Triangular grid
  - advantage = little number of neighbours (3)
  - disadvantage = difficult to represent and to visualize
- Square grid
  - advantage = simple representation and visualization
  - disadvantage = anisotropy
- Hexagonal grid
  - advantage = lowest anisotropy of the three
  - disadvantage = difficult to represent and to visualize

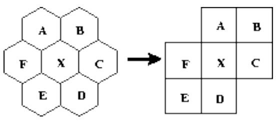
Caractérisation des automates cellulaires - Choix de conception

Mappings hexagonal -> square grids (1)

- Shearing



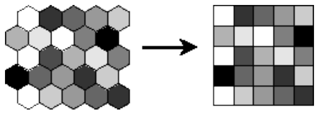
  - cell (i,j) has its center in $\left(i - \frac{j\%2}{2}, \frac{\sqrt{3}}{2}j\right)$ with $i, j \in \mathbb{Z}$,
    - origin in the upper-left corner
    - unity = distance between the cells
  - cell (i,j) is mapped into $\mathrm{shear}(i,j) = \left(i + \left\lfloor \frac{j}{2} \right\rfloor, j\right)$,
  - the neighbourhood relationship becomes :



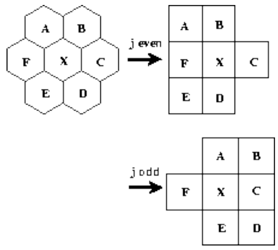
Caractérisation des automates cellulaires - Choix de conception

Mappings hexagonal -> square grids (2)

- Shifting of successive lines in opposite directions



  - cell (i,j) is mapped into
  - the neighbourhood relationship becomes : $\mathrm{shift}(i, j) = (i, j)$.



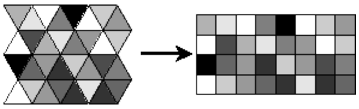
Caractérisation des automates cellulaires - Choix de conception

Mappings hexagonal -> square grids (3)

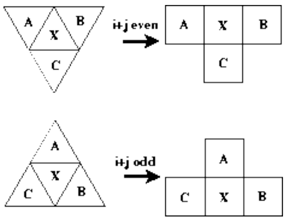
- Shearing
  - advantage = the local neighbourhood relationship remains uniform
  - disadvantages =
    - border conditions more difficult to implement
    - necessary to to transform the representation again for visualization
- Shifting
  - advantages =
    - border conditions as simple to implement as with the square representation
    - simple visualization
  - disadvantage = the neighbourhood depends on the parity of j
    - neighbourhood and rules not homogeneous

Mappings triangular -> square grids (1)

- Similar to the shifting mapping

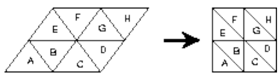


  - cell (i,j) is mapped into $\mathrm{shift}(i,j) = (i,j)$.
  - the neighbourhood relationship becomes:



Mappings triangular -> square grids (2)

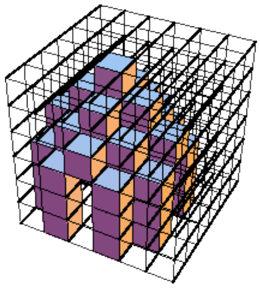
- Other solution :



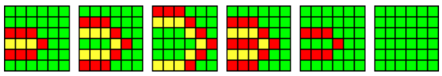
  - state space extended for each square cell $S' = S \times S$
  - neighbourhood of two triangular cells = neighbourhood of a square cell
  - advantage = uniform neighbourhood and transition rules
  - disadvantage =
    - bigger state space
    - bigger transition table for the cells

3-dimensional space

- Lots of possible grids
  - the simplest = cubic grid
  - no grids symmetric-enough for hydrodynamics problems
- Specific visualization problems
  - 3D representation



  - 2D slices representation



Size and shape of the neighbourhood (1)

- Neighbourhood = set of cells with which a given cell will be able to interact
- Description of the neighbourhood = set of cells that belong to the neighbouring of cell (*i,j*)

Cellular automata characterization – Design choices

Size and shape of the neighbourhood (2)

- Von neumann neighbourhood

$$N_{i,j} = \left\{(k,l) \in \mathcal{L} \,\middle|\, |k-i| + |l-j| \leq 1\right\}.$$

- Moore neighbourhood

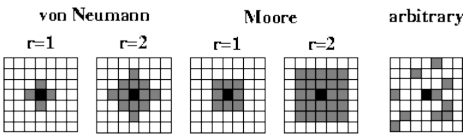
$$N_{i,j} = \left\{(k,l) \in \mathcal{L} \,\middle|\, |k-i| \leq 1 \text{ and } |l-j| \leq 1\right\}$$

- Generalized von Neumann neighbourhood

$$N_{i,j} = \left\{(k,l) \in \mathcal{L} \,\middle|\, |k-i| + |l-j| \leq r\right\}$$

- Generalized Moore neighbourhood

$$N_{i,j} = \left\{(k,l) \in \mathcal{L} \,\middle|\, |k-i| \leq r \text{ and } |l-j| \leq r\right\}.$$



Cellular automata characterization – Design choices

Limit conditions

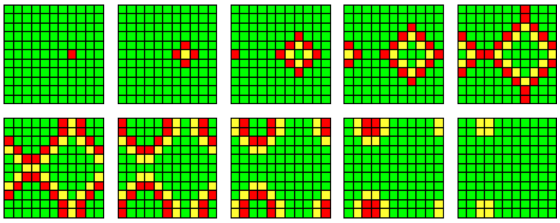
- Formal definition given for infinite grids
  - reasonable and necessary from a theoretical point of view but unworkable
  - some problems have natural boundaries
- 3 types of border
  - periodic
  - reflective
  - with fixed value

Periodic border

- Periodic extension of the grid
- In 1 dimension : ring

J	A	B	C	D	E	F	G	H	I	J	A

  - often used since the closest to a grid of infinite size
- In 2 dimensions : torus
  - not possible to obtain a sphere



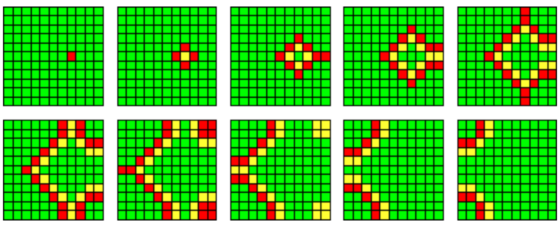
Cellular automata characterization – Design choices

Reflective border

- Repetition of the grid at the border
- In 1 dimension

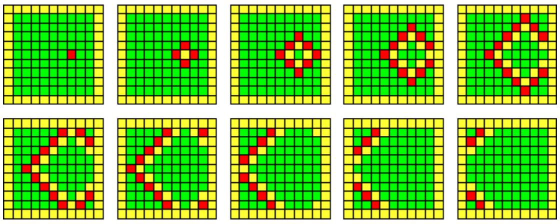
A	A	B	C	D	E	F	G	H	I	J	J

  - adapted when the system to simulate has borders
- In 2 dimensions



Fixed conditions

- Fixed value given to the cells at the border



Mixed conditions

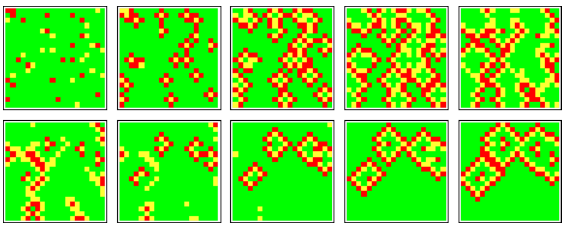
- The 3 types of conditions may be combined
  - different boundaries have different conditions
  - if a boundary has a periodic border, the opposite side will also have the same conditions
  - a long tube can be simulated by imposing periodic conditions in one dimension and reflecting conditions in the other dimension
- Alternatives
  - expand the grid when the structures in the CA touch the edges
    - pb = some structures develop very quickly
  - adopt special transition rules for borders
    - pb = potentially many special cases

Initial conditions

- The initial conditions often condition the future evolution of the automaton
  - construction
  - random generation
- Important considerations
  - lots of automata preserve some quantities
    - peculiar to the model: nb of particules, energy, etc.
    - peculiar to the grid: nb of particules in a line or a column
  - choice so that
    - the first ones are verified
    - the seconds ones are not harmful

Example : Greenberg-Hastings

- If only red cells
  - wave that propagates and vanishes
- If red bar over a yellow bar
  - endless spiral at each end of the bar
  - particular case = red cell next to a cell yellow -> center of emission of periodic waves



  - the number of spirals is preserved!

State space

- CA = finite state automaton
  - Finite number of states
  - Generally small
    - so that we can systematically explore all the CAs of the same family
      - for *s* states and *n* neighbours, the number of Acs is s^{s^n}

s	n	s^n	s^{s^n}
2	3	8	256
2	5	32	4294967296
5	3	125	$2.3 \cdot 10^{87}$

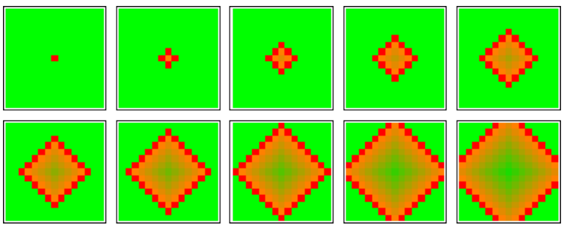
    - to be able to specify the transition rules explicitely and store them in a table (of size s^n)
  - Big number of states interesting to gain in precision
    - but one may prefer a partial differential equations model if a great precision is needed
    - CAs are adapted for systems with fluctuations, where local interactions are important

Generalization of Greenberg-Hastings

- *n* different possible states
- Transition rules:

$$0 \rightarrow \begin{cases} 0 & \text{if no neighbor} = n; \\ n & \text{if at least one neighbor} = n; \end{cases}$$
$$i \rightarrow i-1; \quad \text{for } 1 \leq i \leq n.$$

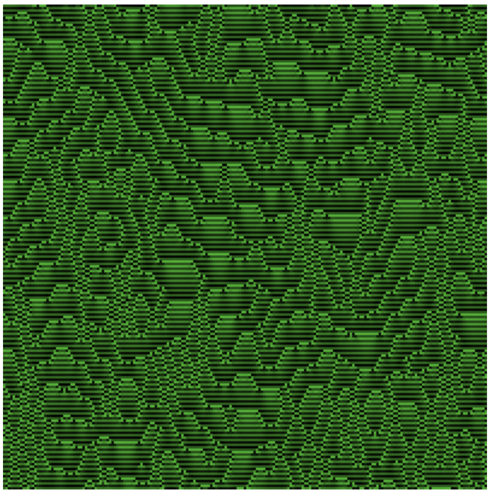
- Longer refractory period



Cellular automata characterization – Design choices

Extension = Coupled-map lattices

- Continuous instead of discrete state-space



Cellular automata characterization – Design choices

State with multiple variables

- The state-space S is the cross-product of the spaces of each of the variables
- Excitable medium characterized by
  - excitation variable *u*
    - *u*=1 -> excited
    - *u*=0 -> resting
  - phase variable *v*
    - *u*=1 -> *v* is the time during which the cell is excited
    - *u*=0 -> *v* is the time before the cell is excitable again

$$
\begin{aligned}
(0, v\ ) &\rightarrow \big(1, v\big) \text{ if} \geq 1 \text{ neighbor } u = 1 \text{ and } v \leq v_{th}. \\
(0, v\ ) &\rightarrow \big(0, \max\{0, v - v_-\}\big) \\
(1, v_m) &\rightarrow \big(0, v_m\big) \\
(1, v\ ) &\rightarrow \big(1, \min\{v_m, v + v_+\}\big),
\end{aligned}
$$

Transition rules

- The most important aspect of CAs
- Conditionned by
  - the geometry
  - the neighbourhood
  - the state space
- Even if the transition rule directly determines the evolution of the CA, it is often impossible to predict its evolution except by simulating it

Direct vs. indirect specification

- direct specification = transition rules for all possible configurations
  - for the 1D automaton with 3 states

$$
\begin{array}{lll}
(0,0,0) \to 0 & (1,0,0) \to 0 & (2,0,0) \to 2 \\
(0,0,1) \to 0 & (1,0,1) \to 0 & (2,0,1) \to 2 \\
(0,0,2) \to 2 & (1,0,2) \to 2 & (2,0,2) \to 2 \\
(0,1,0) \to 0 & (1,1,0) \to 0 & (2,1,0) \to 0 \\
(0,1,1) \to 0 & (1,1,1) \to 0 & (2,1,1) \to 0 \\
(0,1,2) \to 0 & (1,1,2) \to 0 & (2,1,2) \to 0 \\
(0,2,0) \to 1 & (1,2,0) \to 1 & (2,2,0) \to 1 \\
(0,2,1) \to 1 & (1,2,1) \to 1 & (2,2,1) \to 1 \\
(0,2,2) \to 1 & (1,2,2) \to 1 & (2,2,2) \to 1
\end{array}
$$

  - long and tedious
  - possible to introduce « wildcards »
- implicit specification = rules given as formulas

Cellular automata characterization – Design choices

Direct specification + wildcards

- The same rule using wildcards

$$
\begin{array}{ll}
(0,0,0) \to 0 & (0,0,1) \to 0 \\
(\bullet,0,2) \to 2 & (1,0,0) \to 0 \\
(2,0,\bullet) \to 2 & (1,0,1) \to 0 \\
(\bullet,2,\bullet) \to 1 & \\
(\bullet,1,\bullet) \to 0 &
\end{array}
$$

  - being careful that the specification enable the construction of the full table in a consistant way
  - still more complicated than the specification given in the introduction

- Extension by ordering the application of the rules

$$
\begin{array}{l}
(\bullet,0,2) \to 2 \\
(2,0,\bullet) \to 2 \\
(\bullet,0,\bullet) \to 0 \\
(\bullet,2,\bullet) \to 1 \\
(\bullet,1,\bullet) \to 0
\end{array}
$$

Cellular automata characterization – Design choices

Direct specification + wildcards + symmetry

- Simplification by grouping the states depending on the symmetry of the grid

$$
\begin{array}{l}
(2,0,\bullet) \to 2 \\
(\bullet,0,\bullet) \to 0 \\
(\bullet,2,\bullet) \to 1 \\
(\bullet,1,\bullet) \to 0
\end{array}
$$

  - same nb of cases as in the specification in the introduction

- Event more important in 2 dimensions

$$
\begin{array}{ccc}
 & 2 & \\
\bullet & 0 & \bullet \\
 & \bullet &
\end{array} \to 2
\qquad
\begin{array}{ccc}
 & \bullet & \\
\bullet & 0 & \bullet \\
 & \bullet &
\end{array} \to 0
$$

$$
\begin{array}{ccc}
 & \bullet & \\
\bullet & 2 & \bullet \\
 & \bullet &
\end{array} \to 1
\qquad
\begin{array}{ccc}
 & \bullet & \\
\bullet & 1 & \bullet \\
 & \bullet &
\end{array} \to 0
$$

  - the first rule is valide for all the configurations obtained by rotation of the grid
  - the complete table has a size of $3^5 = 243$!!!

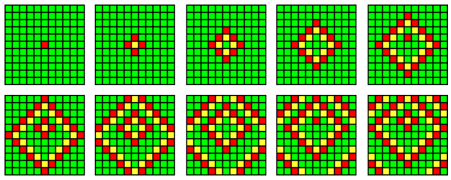
Cellular automata characterization – Design choices

Little changes in the table

- Even a small change in the rules table can lead to very significant changes in the resulting dynamics
  - ex1:

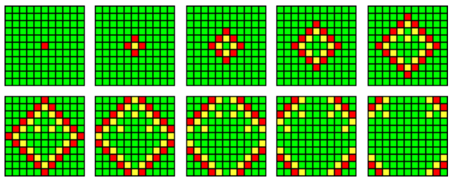
$$\begin{array}{ccc} & 1 & \\ 1 & 0 & 1 \\ & 0 & \end{array} \rightarrow 0$$

$$\begin{array}{ccc} & 1 & \\ 1 & 0 & 1 \\ & 0 & \end{array} \rightarrow 2.$$



  - ex2:

$$\begin{array}{ccc} & 1 & \\ 1 & 0 & 1 \\ & 0 & \end{array} \rightarrow 1,$$



Cellular automata characterization – Design choices

Totalistic rules (1)

- Lots of Cas don’t care about the exact disposal of the cells but only about the number of neighbours in a given state
  - allows to simplify the transition table
- Totalistic CA
  - only consider the sum of the states of the neighbouring cells

$$x^{t+1}(r) = f\left(\sum_{r' \in \mathcal{N}(r)} x^t(r')\right),$$

- « Outer totalistic » CA
  - also depends on the current state of the cell concerned

$$x^{t+1}(r) = f\left(x^t(r), \sum_{r' \in \mathcal{N}(r)} x^t(r')\right).$$

Cellular automata characterization – Design choices

Totalistic rules (2)

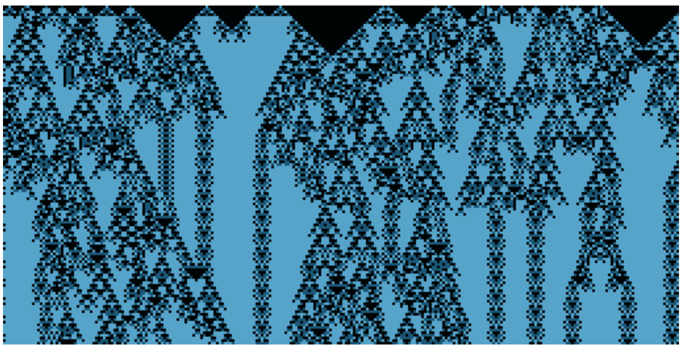
- Most of the time, we sum only a subset of the states
  - ex: greenberg-hastings = sum of the cells in state 2

$$x^{t+1}(r) = f\left(x^t(r), \sum_{r' \in \mathcal{N}(r)} g\left(x^t(r')\right)\right),$$

  - in the example:
    - g(0) = 0
    - g(1) = 0
    - g(2) = 1
    - f(0,0) = 0
    - f(0, x>0) = 2
    - f(1,x) = 0
    - f(2,x) = 1

Cellular automata characterization – Design choices

A 1D totalistic automaton with 3 states



Cellular automata characterization – Design choices

Probabilistic rules (1)

- The new state of the cell depends on:
  - the configuration of the neighbourhood
  - probabilities associated to the different possible states (the sum of the probabilities has to be equal to 1)
  - The probabilistic choices of the cells are independent from one another
  - The transition function becomes: $G : S^n \times S \to [0,1]$,
    - G verifying $\forall \{s_i\}_{i=1}^n \ : \ \sum_{s'} G(\{s_i\}, s') = 1$
    - in general, G is non nul for only some values
- important to simulate a number of systems in which the functionning is noisy

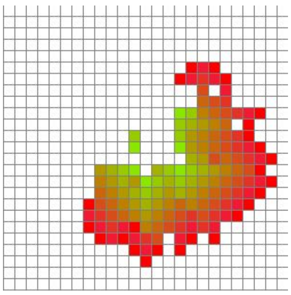
Cellular automata characterization – Design choices

Probabilistic rules (2)

- Example

$$
\begin{aligned}
0 &\to \begin{cases} n & \text{if } \ 1 \text{ neighbor} = n \text{ with } p = 0.65; \\ n & \text{if} > 1 \text{ neighbor} = n \text{ with } p = 0.9; \\ 0 & \text{otherwise} \end{cases} \\
i &\to i-1; \quad \text{for } 1 \leq i \leq n.
\end{aligned}
$$

  - take into account the fact that propagation does not always noccur
  - irregular front but more circular



Cellular automata characterization – Design choices

A probabilistic 1D automaton