Self-Organising Systems

Rudolf Mayer

Institute of Information Systems Engineering, Data Science Group

(mayer@ifs.tuwien.ac.at)

SBA Research
(www.sba-research.org)

Agenda

- Introduction & Definitions of Self-organising Systems
- Cellular Automata
- Genetic algorithms
- Ant Colony Optimisations
  - *to be extended in swarm optimisation lectures*
- Agent Based Systems / Multi Agent Systems

Agenda

- Introduction & Definitions of Self-organising Systems
- Cellular Automata
- Genetic algorithms
- Ant Colony Optimisations
  - *to be extended in swarm optimisation lectures*
- Agent Based Systems / Multi Agent Systems

Self-Organising Systems

- Methods considered in this course are applicable to
  - Simulation tasks (e.g. Ants, Swarms, Agents, ..)
  - Optimisation proplems (Swarms, Agents, GA, ...) / problem solving
  - Data analysis (SOM)

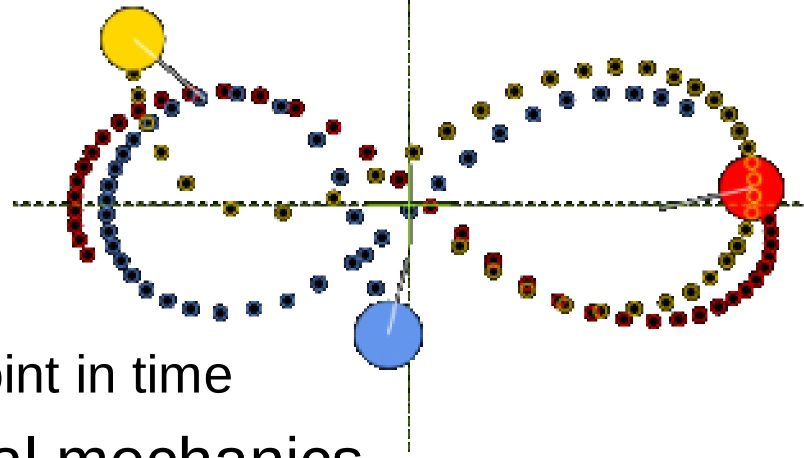
  - Some methods suited for more, some for just one of these tasks
- Especially for simulation: useful when the system to be modelled is ***complex***

Introduction: Complex Systems

- A system that exhibits some of these characteristics
  - Feedback loops
  - Emergent organization (appearance of unplanned organized behavior)
  - Numerosity
  - Hierarchical organization
  - Some degree of spontaneous order (robustness of the order)
  - Difficult to model top-down (e.g. with differential equations)

- Examples
  - Three-Body Problem
  - Weather Forecast
  - Ecosystems

Introduction: Complex Systems



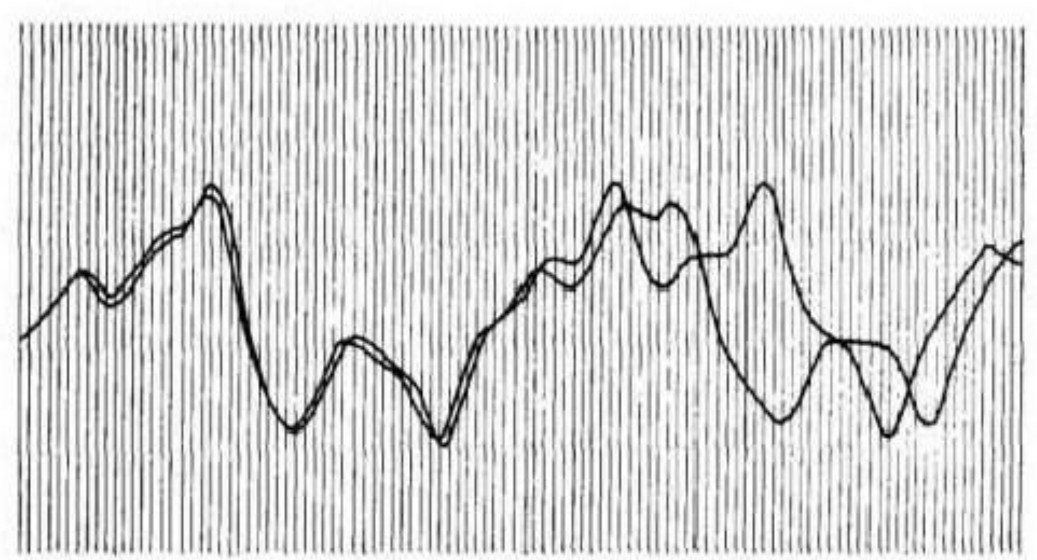
- Three-Body Problem
  - E.g. sun, moon, earth
    - Initial set of positions, masses and velocities of the bodies, for particular point in time
  - In accordance to the laws of classical mechanics
    - Newton's laws of motion and of universal gravitation
  - Determining the motions of the three bodies
  - No general analytical solution given by algebraic expressions and integrals; motion generally non-repeating (Henri Poincaré & Ernst Bruns, 1887)

Introduction: Complex Systems

- Weather forecast
  - Edward Lorenz in the 1950s: skeptical of appropriateness of linear statistical models in meteorology
    - Most atmospheric phenomena are non-linear
  - Foundation of chaos theory: two states differing by imperceptible amounts may evolve into two considerably different states
  - If there is any error in observing the present state: acceptable prediction of a future state may be impossible

Introduction: Complex Systems

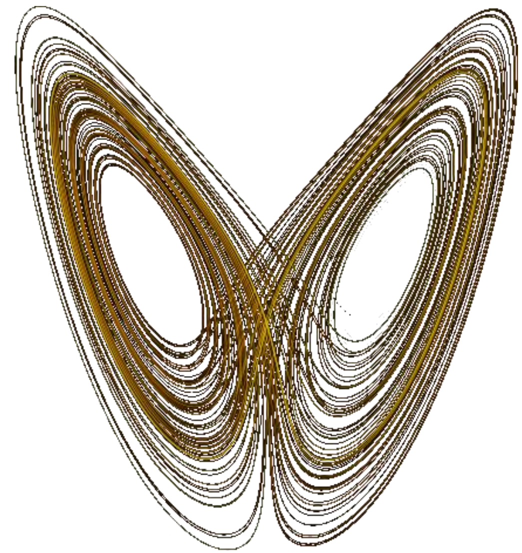
- Weather forecast
  - Built a weather model based on 12 variables
    - Observed that minute changes in the initial state would create two very differing results
    - Initial state changed from 6-digit precision to 3-digit precision
      ➔ final result not comparable anymore



0.506127 → 0.506

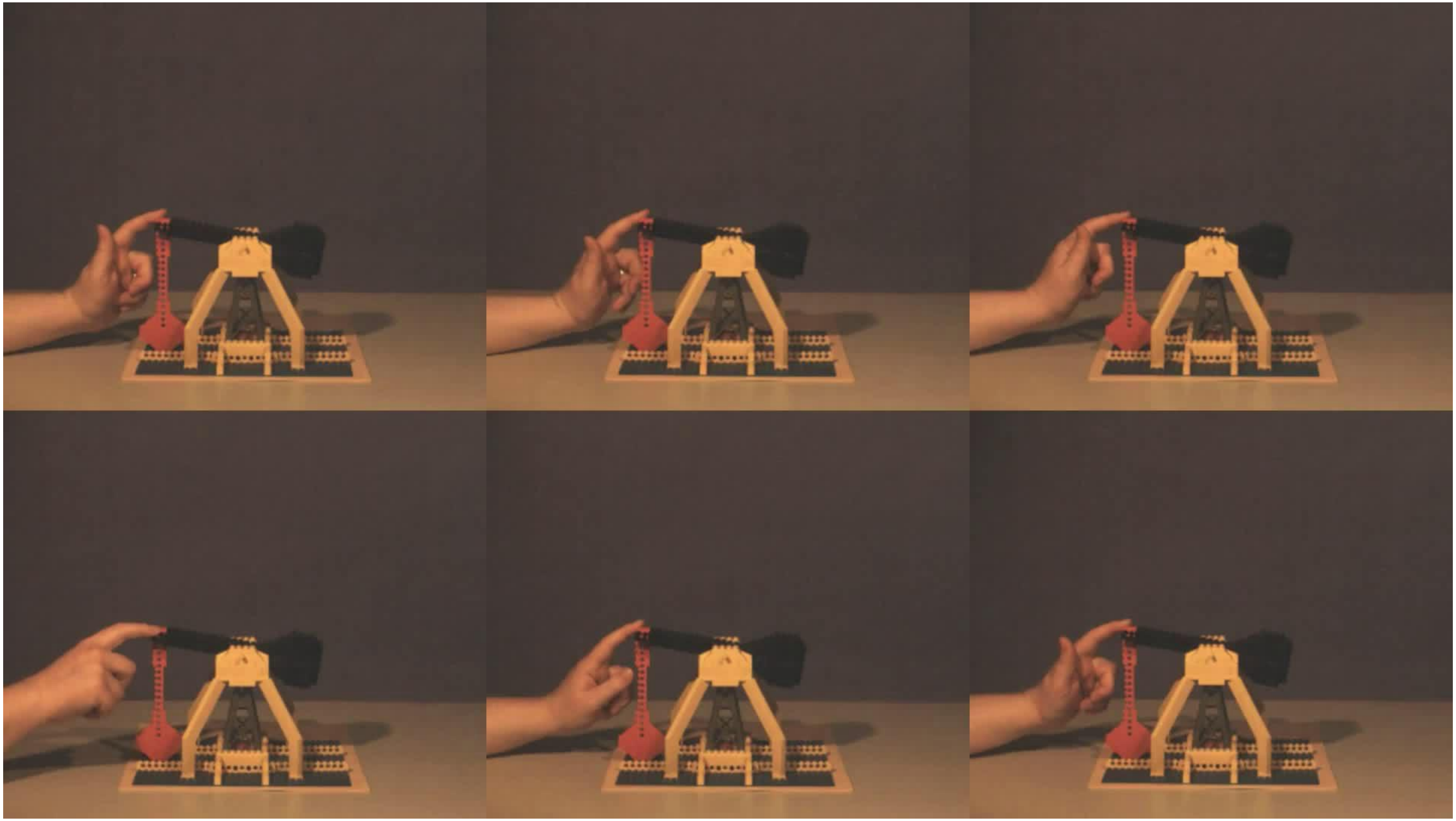
Introduction: Complex Systems

- Weather forecast
  - Butterfly effect
    - Not the 2004 movie ☺
  - Butterfly effect (1969):
    small changes in initial state
    ➔ (potentially) very different final state



  - Inevitable inaccuracy and incompleteness of observations
    ➔ precise long-range forecasting seem nonexistent
  - “Does the flap of a butterfly’s wings in Brazil set off a tornado in Texas?”

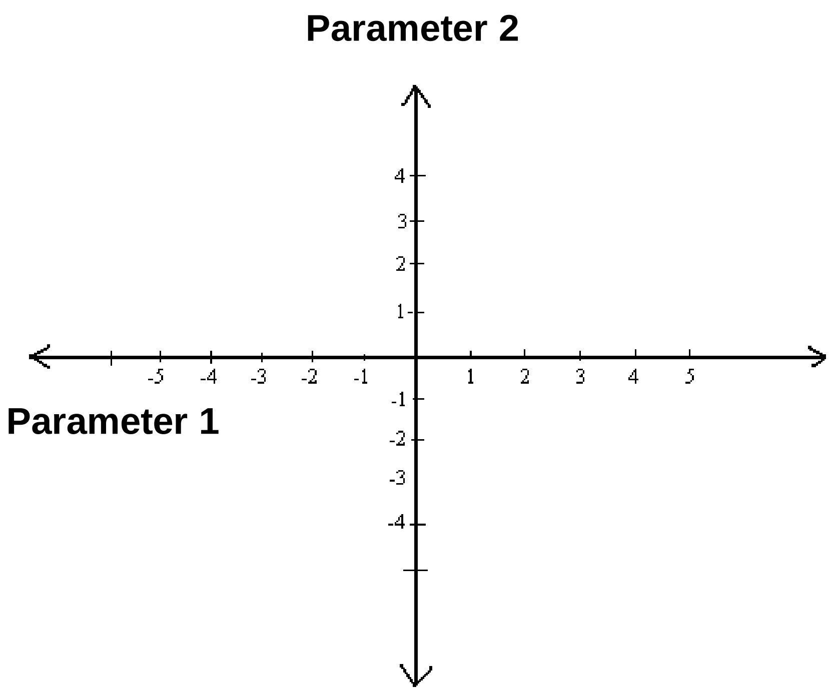
Introduction: Complex Systems



- Multiple double pendulums, differing slightly in intial state

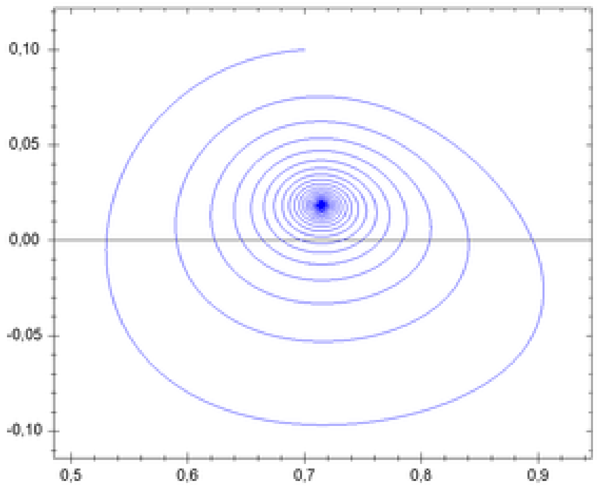
Phase Space

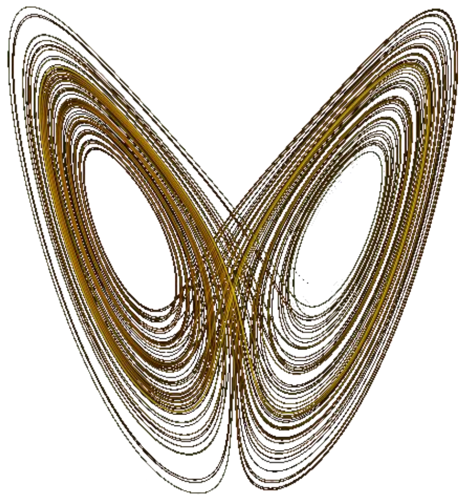
- Phase Space is the n-dimensional space that is spanned by the relevant parameters of a system



Phase Space

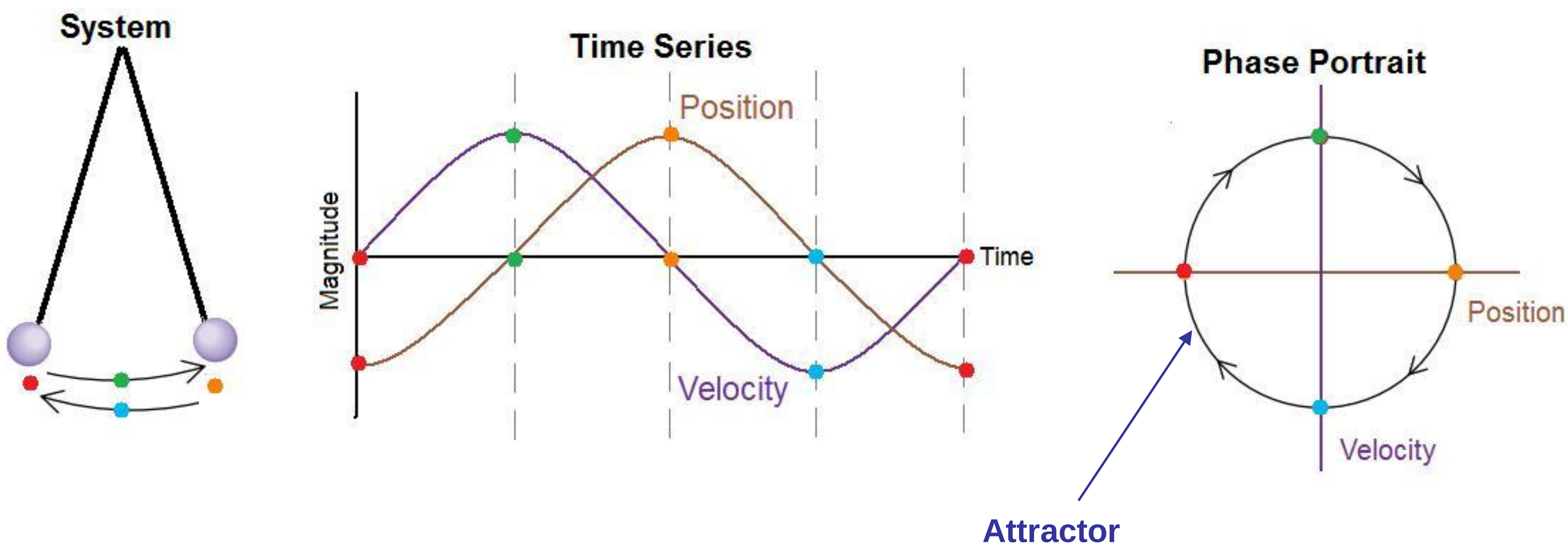
- Phase Space is the n-dimensional space that is spanned by the relevant parameters of a system
  - Space in which all possible states of a system are represented
  - System's evolving state (over time) traces a path





Phase plots: Attractors

- Space in which all possible states of a system are represented
- Example: Pendulum



Tipping Point

- Systems can have different attractors
- Stability within a certain range of parameters (self-stabilisation)
- If range is exceeded, systems can “switch” to other attractor

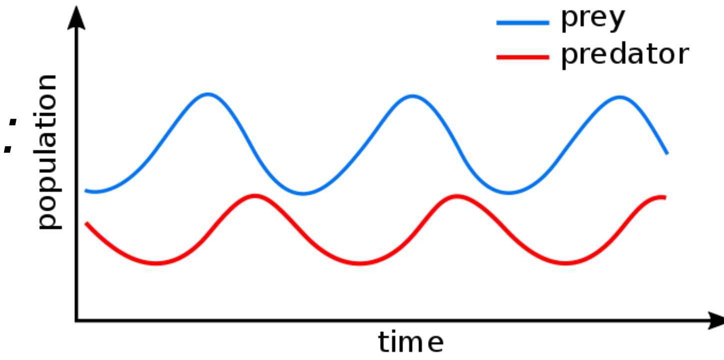
  ➔ Tipping Point

Tipping Point

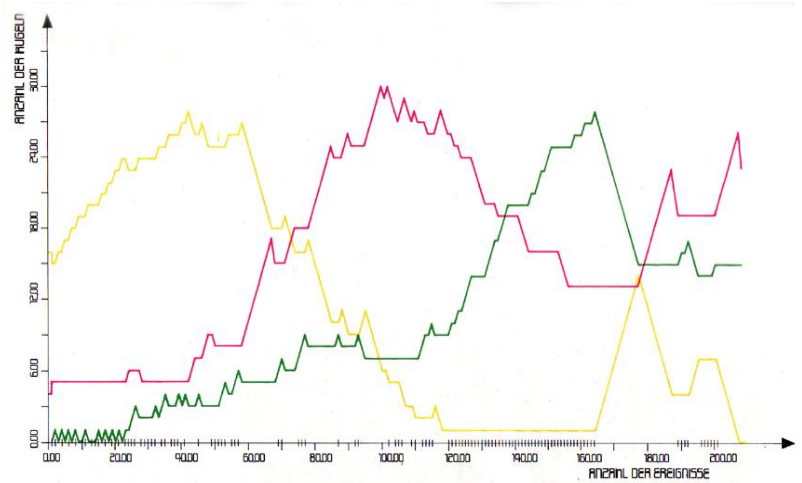
- Systems can have different attractors
- Stability within a certain range of parameters (self-stabilisation)
- If range is exceeded, systems can “switch” to other attractor
  ➔ Tipping Point
- Example: Climate System of the Earth
  – To a certain degree self-stabilising over certain range of parameters
  – If e.g. global temperature rises over tipping point: “runaway greenhouse effect”
  – Positive feedback loops can be induced
    • Melting of Greenland ice
    • Methane release from Siberian unfreezing
  – Eventually stabilisation at different attractor

Introduction: Complex Systems

- Well-known model of biological system: **predator/prey** (*Lotka–Volterra equations*): dynamics of two species



- More complex Ecosystems
  - Community of living organisms (plants, animals & microbes)
  - In conjunction with nonliving components of their environment
    - air, water, mineral soil, …
  - Interacting as a system
  - Becomes difficult to model as analytical model
    - Numerical simulation model



Modelling Approaches

- Top Down
  - “Traditional”
  - Differential Equation

- Bottom Up
  - Simulation using Simple Rules
  - Complexity Emerges by Interaction

Self-Organisation

- What is self-organisation?
  - **Process** where some form of global order (or coordination) arises out of local interactions between components of an initially disordered system
  - Process is spontaneous: not directed or controlled by any agent or subsystem inside or outside of the system
    - Often triggered by **random fluctuations,** amplified by **positive feedback**
    - ***Rules*** followed by process may be chosen or caused by an agent
  - Formative and constraining influences are **decided by the elements** of the system itself.
    System develops towards state of equilibrium (attractor)
    - If close enough to the attractor: remain close even if disturbed
    - Survive / self-repair substantial damage or perturbations

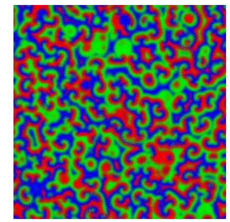
Self-Organisation: Properties

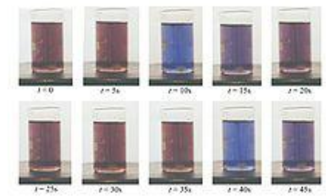
- Strong dynamical non-linearity
  - Often involving (positive and/or negative) feedback
- Multiple interactions (between elements)
- Complex: elements are interlinked by many & constantly changing relations
  - Elements can change as well ➔ hinders prediction of state
- Self-referring: behaviour has impact on the system and future behaviour
- Autonomous: interactions are defined by the system itself

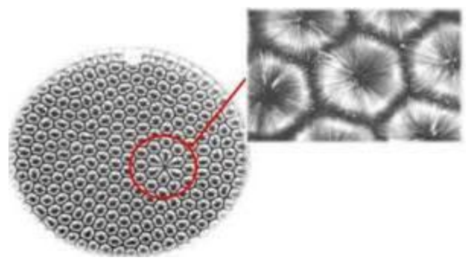
Self-Organisation

- Self-organization occurs in a variety of *physical, chemical, biological, social and cognitive* systems

- Common examples
  – Crystallization, chemical clocks / oscillators
  – Swarming in groups of animals
  – Convection patterns







- Inspired
  – Economics, collective intelligence, …
  – Mathematical and computational simulations & models

SOS in mathematics / computer science

- Cellular automata
- Random graphs
- Evolutionary computation (genetic algorithms, genetic programming) and artificial life

- Swarm intelligence: emergent behavior
  - Emergence: the way complex systems and patterns arise out of a multiplicity of relatively simple interactions.

- Agents / Multi-agent systems

Self-organising systems

- A bit of unsupervised learning (no labels)

- A bit of reinforcement learning
  (reward / penalise for action)

- Often applied to
  - Simulation tasks (e.g. Ants, Swarms, Agents, ..)
  - Optimisation proplems (Swarms, Agents, GA, ...) / problem solving
  - Data analysis

Complex vs. complicated system

- ?

Complex vs. complicated system

- Complicated systems / problems
  - Scale of the problem
  - Increased requirements around coordination or specialised expertise
  - Relatively high degree of certainty of outcome repetition
    - E.g.: sending a rocket to space

- Complex systems
  - Based on relationships, self-organisation and evolution
  - Formula have limited application
  - Experience & expertise contribute but are neither necessary nor sufficient to assure success

Agenda

- Introduction & Definitions of Self-organising Systems
- Cellular Automata
- Genetic algorithms
- Ant Colony Optimisations
  - *to be extended in swarm optimisation lectures*
- Agent Based Systems / Multi Agent Systems

Cellular Automata

- *A CA is an array of identically programmed automata, or cells, which interact with one another in a neighbourhood and have definite state*
- ***Microscopic*** modelling approach

- Topics
  - Roots of the Cellular Automata Idea
  - Building Cellular Automata
  - Examples of well-known CAs
  - Behaviour of CAs
  - Applications

Cellular Automata

- Origins in the 1940s
  - John von Neumann, Stanislaw Ulam, Norbert Wiener
  - Theory of Computation, Automata Theory
  - Simulation of Complex Systems by Interaction of using “Simple” Rules

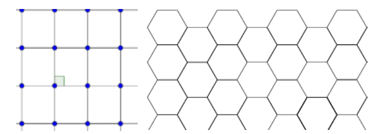
- 1970s: Game of Life
  - John Horton Conway

- 1980s – present: Stephen Wolfram & others
  - creator of Mathematica & Wolfram Alpha (http://www.wolframalpha.com/)

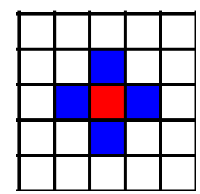
Building Cellular Automata

- Basic building blocks
  - The Cell (Node)

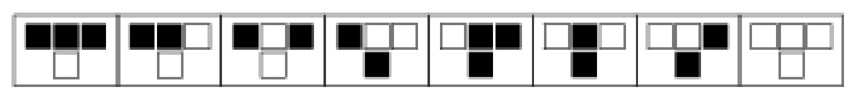
    - States
  - The Lattice



    - Cell Arrangement
  - Neighbourhoods



  - Transition Rules



Building Cellular Automata

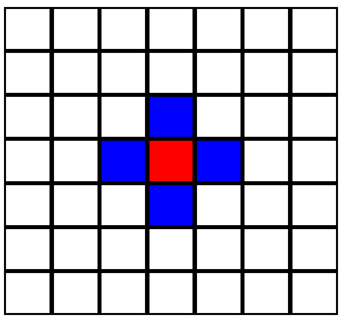
- CA consists of a regular grid of cells (lattice)
  - Each cell is in one of a finite number of states (simplest case: on/off)
- Grid can be in any finite number of dimensions
  - often 1D / 2D (especially if spatial aspect is relevant)
- Neighbourhood of a cell: set relative to a specified cell
- *Generation* creation
  - Initial state t=0 with cell assignment
  - Subsequent states (t=1,…n) created with **rules** that define the state of each cell

➔ Discrete modelling

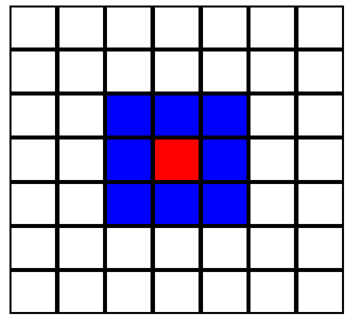
Building Cellular Automata

- Neighbourhoods

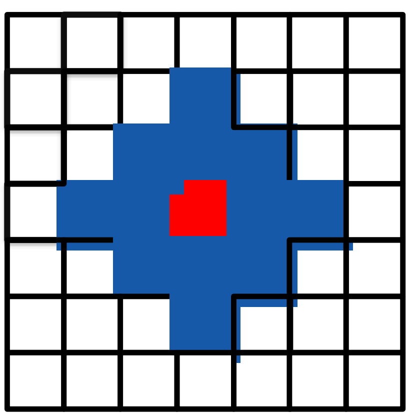
2-D

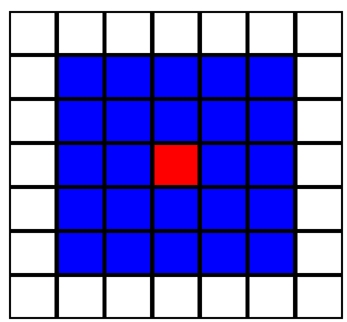


von Neumann Neighbourhood



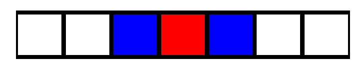
Moore Neighbourhood





Extended Neighbourhood
(2nd order)

1-D



Building Cellular Automata

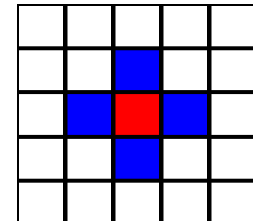
- Rules: mathematical functions; generally:
  - One rule applicable to **all** cells
  - Applied to all cells **at the same time**
    - Order is not important
  - Does not change over time
    - Exceptions, e.g. asynchronous cellular automaton

- Update rule
  - $S_{t+1} = f(s, s_1, s_2, \ldots s_n)$

new state of cell **state of cell** **state of the neighbors**

Update rule: simple example

Update rule: $f(s, s_1, s_2, s_3, s_4) = \sum s \bmod 2$



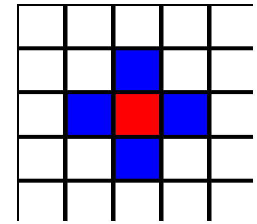
Old state

1	0	1	0	0	1	1
0	1	1	0	1	1	1
0	0	1	0	0	0	1
1	0	1	1	1	1	1
0	1	0	0	0	0	0
0	0	1	1	1	1	1
0	0	0	0	1	1	0

New state

Update rule: simple example

Update rule: $f(s, s_1, s_2, s_3, s_4) = \sum s \bmod 2$



Old state

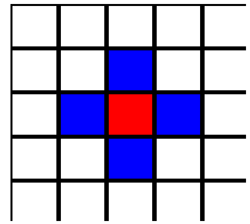
1	0	1	0	0	1	1
0	1	1	0	1	1	1
0	0	1	0	0	0	1
1	0	1	1	1	1	1
0	1	0	0	0	0	0
0	0	1	1	1	1	1
0	0	0	0	1	1	0

New state

			?			

Update rule: simple example

Update rule: $f(s, s_1, s_2, s_3, s_4) = \sum s \bmod 2$



Old state

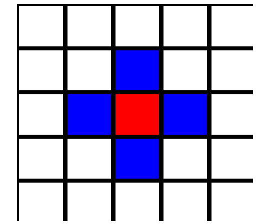
1	0	1	0	0	1	1
0	1	1	0	1	1	1
0	0	1	0	0	0	1
1	0	1	1	1	1	1
0	1	0	0	0	0	0
0	0	1	1	1	1	1
0	0	0	0	1	1	0

New state

			1			

Update rule: simple example

Update rule: $f(s, s_1, s_2, s_3, s_4) = \sum s \bmod 2$



Old state

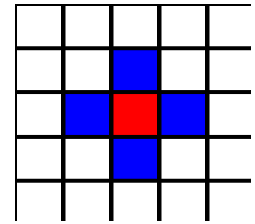
1	0	1	0	0	1	1
0	1	1	0	1	1	1
0	0	1	0	0	0	1
1	0	1	1	1	1	1
0	1	0	0	0	0	0
0	0	1	1	1	1	1
0	0	0	0	1	1	0

New state

			1			
				?		

Update rule: simple example

Update rule: $f(s, s_1, s_2, s_3, s_4) = \sum s \bmod 2$



Old state

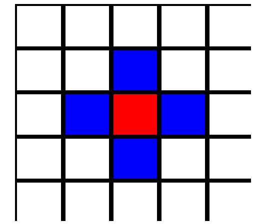
1	0	1	0	0	1	1
0	1	1	0	1	1	1
0	0	1	0	0	0	1
1	0	1	1	1	1	1
0	1	0	0	0	0	0
0	0	1	1	1	1	1
0	0	0	0	1	1	0

New state

			1			
				0		

Update rule: simple example

Update rule: $f(s, s_1, s_2, s_3, s_4) = \sum s \bmod 2$



Old state

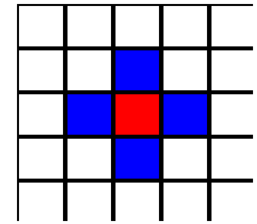
1	0	1	0	0	1	1
0	1	1	0	1	1	1
0	0	1	0	0	0	1
1	0	1	1	1	1	1
0	1	0	0	0	0	0
0	0	1	1	1	1	1
0	0	0	0	1	1	0

New state

						?
			1			
				0		

Update rule: simple example

Update rule: $f(s, s_1, s_2, s_3, s_4) = \sum s \bmod 2$



Old state

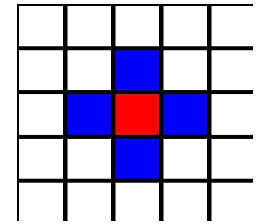
1	0	1	0	0	1	1
0	1	1	0	1	1	1
0	0	1	0	0	0	1
1	0	1	1	1	1	1
0	1	0	0	0	0	0
0	0	1	1	1	1	1
0	0	0	0	1	1	0

New state

						0
			1			
				0		

Update rule: simple example

Update rule: $f(s, s_1, s_2, s_3, s_4) = \sum s \bmod 4$



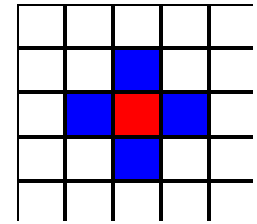
Old state

1	2	1	0	2	1	1
2	3	1	2	3	1	1
0	2	1	2	0	2	1
3	2	1	1	1	5	1
0	1	2	0	2	2	0
2	0	1	1	1	3	1
2	2	0	2	3	1	2

New state

Update rule: simple example

Update rule: $f(s, s_1, s_2, s_3, s_4) = \sum s \bmod 4$



Old state

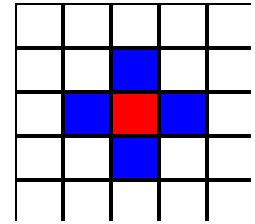
1	2	1	0	2	1	1
2	3	1	2	3	1	1
0	2	1	2	0	2	1
3	2	1	1	1	5	1
0	1	2	0	2	2	0
2	0	1	1	1	3	1
2	2	0	2	3	1	2

New state

			1			

Update rule: simple example

Update rule: $f(s, s_1, s_2, s_3, s_4) = \sum s \bmod 4$



Old state

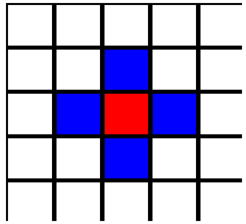
1	2	1	0	2	1	1
2	3	1	2	3	1	1
0	2	1	2	0	2	1
3	2	1	1	1	5	1
0	1	2	0	2	2	0
2	0	1	1	1	3	1
2	2	0	2	3	1	2

New state

			1			
				2		

Update rule: simple example

Update rule: $f(s, s_1, s_2, s_3, s_4) = \sum s \bmod 4$



Old state

1	2	1	0	2	1	1
2	3	1	2	3	1	1
0	2	1	2	0	2	1
3	2	1	1	1	5	1
0	1	2	0	2	2	0
2	0	1	1	1	3	1
2	2	0	2	3	1	2

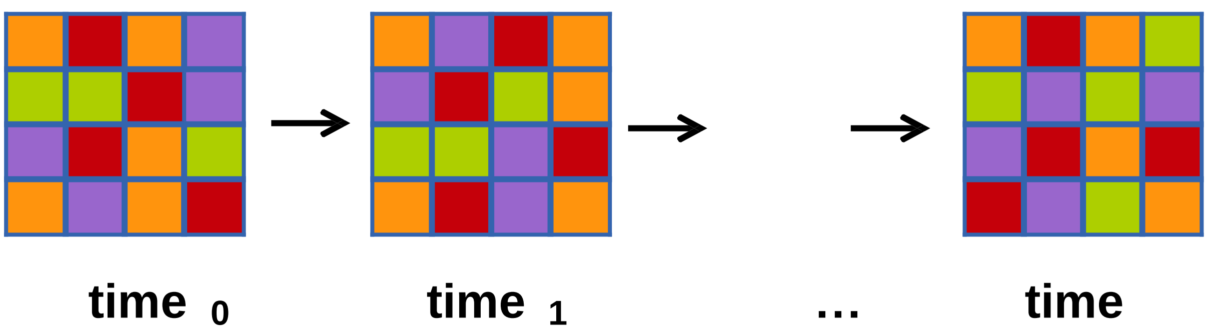
New state

						0
			1			
				2		

Updates

- Updates happen to all cells simultaneously
  - Neighborhoods are all computed from the same system state
  - Update order of cells is irrelevant

- Steps in one iteration
  - Determine neighbors of all cells
  - Compute state updates for all cells (and store them)
  - Apply the updated states



Rule Types

- **Explicit** Rules: every group of states of the neighbourhood cells is related to a state of the core cell
  - E.g. 1-D CA: a rule could be "011 -> x0x"
  - Core cell becomes 0 in the next time step **if left cell** is 0, **right cell** is 1 **and core cell** is 1. *Every possible state has to be described.*

- ***Totalistic*** Rules: state of the core cell only dependent upon a *function of the states* (often: sum) of the neighbourhood cells
  - E.g. If sum of adjacent cells is 4 ➔ state of the core cell is 1; in all other cases the state of the core cell is 0.

- ***Legal*** Rules: Subset of all *possible* rules
  - Those that produce no 1s from 0-state lattices, and provide symmetry (e.g. 110 → 1 and 011 → 1)

Algorithm properties

- CAs develop in space and time
- Cells arranged to n-dimensional lattices
- Finite and discrete cell states
- Cells have identical properties and transition rules
- Future state of cell only depending on
  - Neighbourhood of cell and
  - Defined transition rules

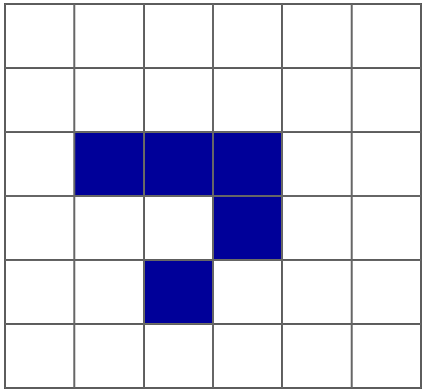
- Discrete Simulation Method

Conway's Game of Life

- John Horton Conway, 1970
- Eight neighbours (Moore neighbourhood)

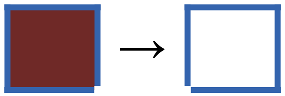
- Rules
  - A cell that is dead becomes alive at time t+1 if exactly three cells are alive (reproduction)
  - A cell that is alive at time t dies at time t+1 if at time t
    - less than two (under-population) or
    - more than three cells are alive (over-crowding)

  - All instances of Game of Life follow same rule
    - Difference is in initial state

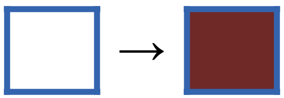


Conway's Game of Life

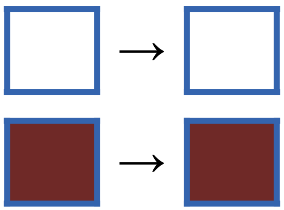
- Rules:



An alive cell with fewer than two or more than three alive neighbors dies ("under-population" or "overcrowding")



A dead cell with exactly three alive neighbors becomes alive ("reproduction")



Cells keep their state in any other case

Conway's Game of Life

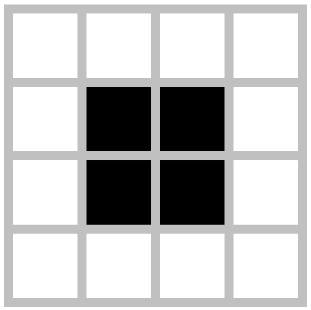
- A very simple example
  - Well suited to show the **concepts** of CAs

- Well studied
  - Pattern analysis of the Game of Life became its own science

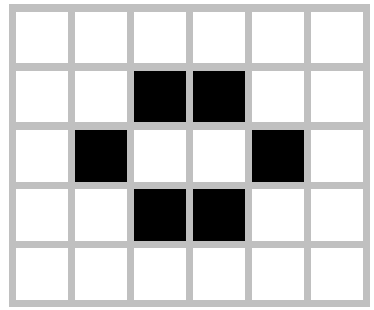
Conway's Game of Life: Examples

- Still / static

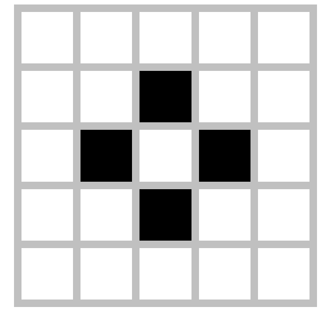
• dead ➔ alive: exactly three cells are alive
• alive ➔ dead: less than two or more than three cells are alive



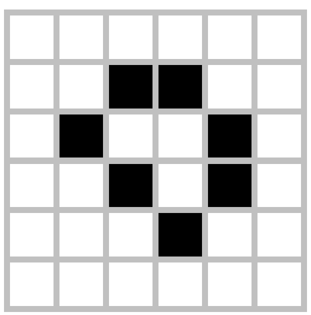
Block



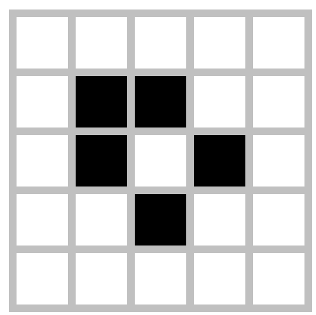
Beehive



Tub



Loaf



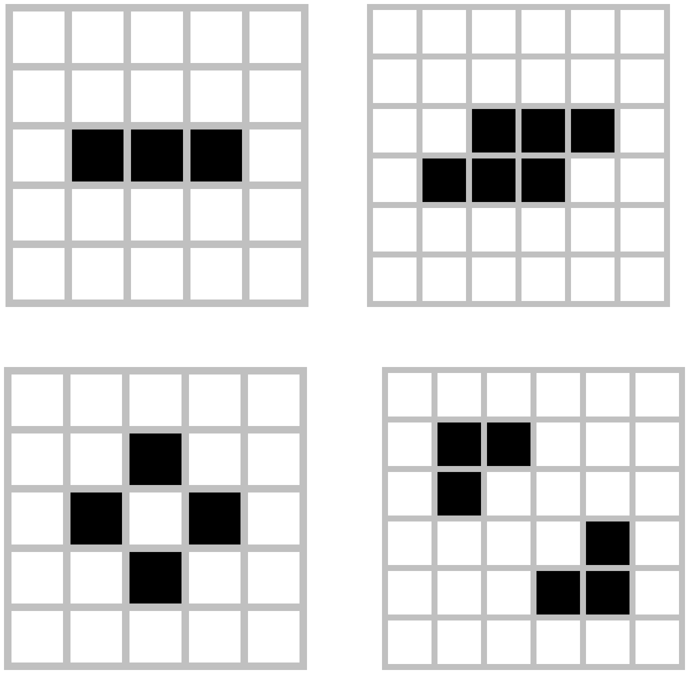
Boat

http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life

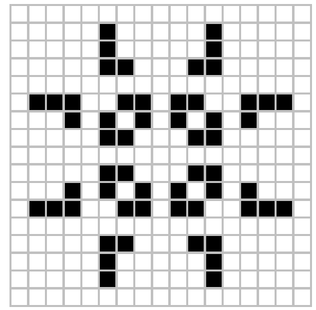
Conway's Game of Life: Examples

- Oscilating

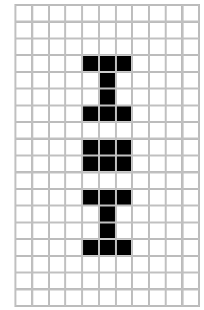
- **dead ➔ alive: exactly three cells are alive**
- **alive ➔ dead: less than two or more than three cells are alive**



2 cycles



3 cycles



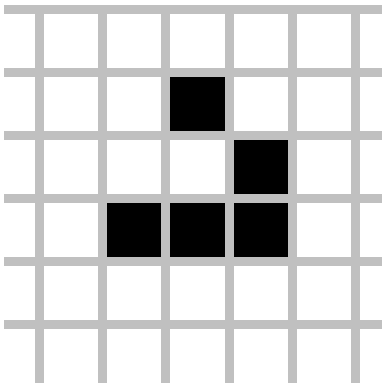
15 cycles

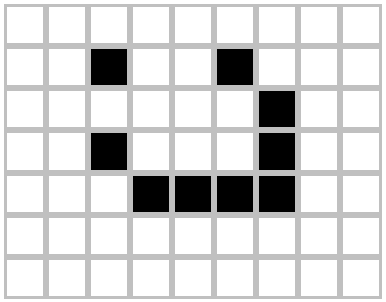
http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life

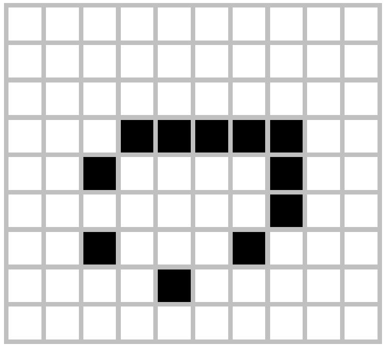
Conway's Game of Life: Examples

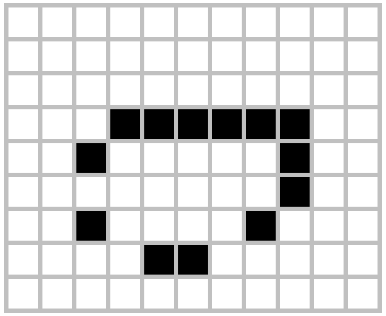
- dead ➔ alive: exactly three cells are alive
- alive ➔ dead: less than two or more than three cells are alive

- Spaceships







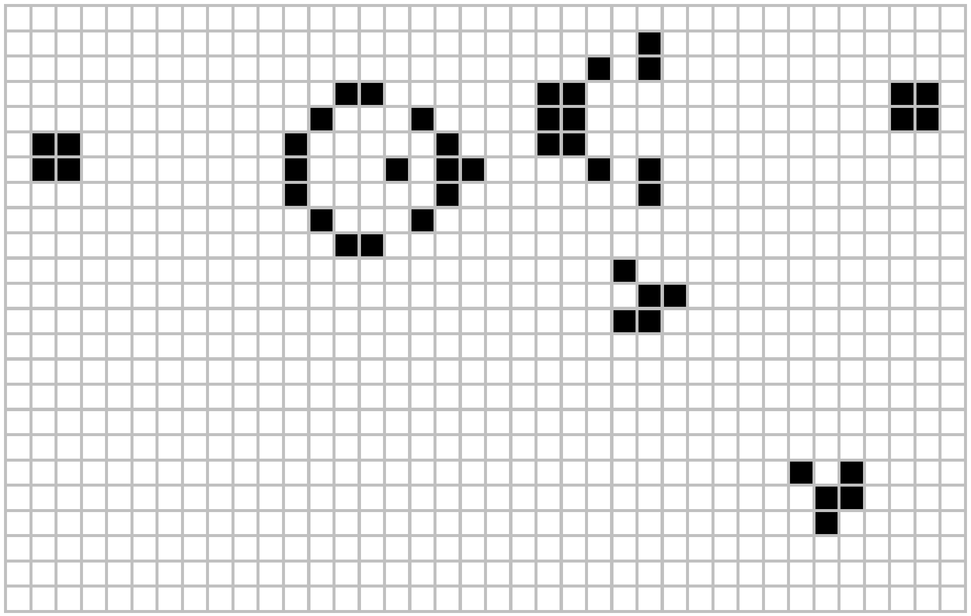


http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life

Conway's Game of Life: gun

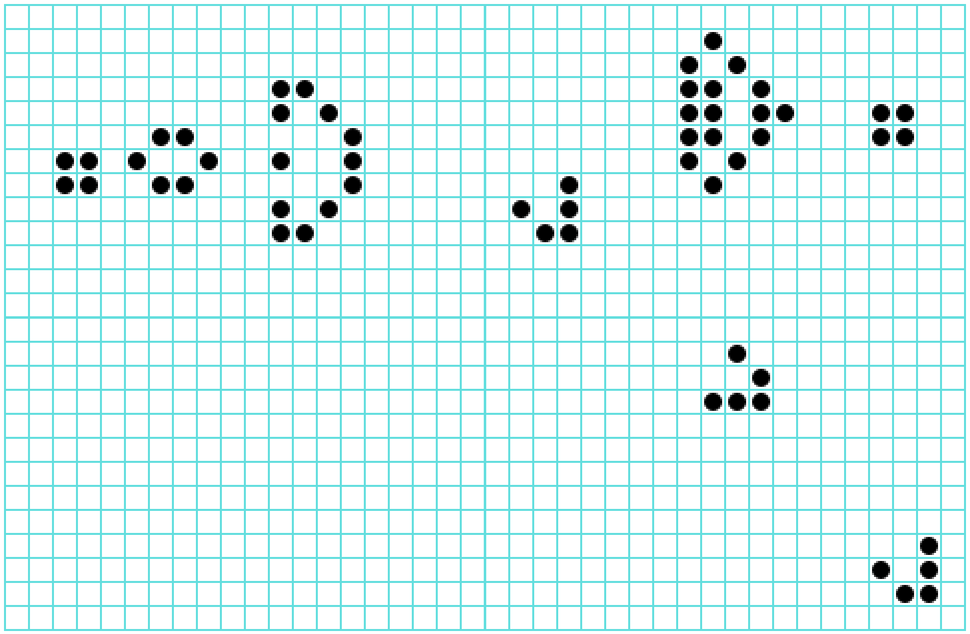
- Infinite growth: pattern that exhibit infinite growth, population is unbounded
  - Conway initially considered that impossible
- First example: Gosper glider gun, found by Bill Gosper (1970)
- Gun: main part that oscillates and periodically emits “spaceships”

Conway's Game of Life: gun



- Source: http://www.conwaylife.com

Conway's Game of Life: gun



- Source: http://www.numericana.com

Conway's Game of Life: “Breeder”

http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life

Conway's Game of Life: “Puffer train”

http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life

Conway's Game of Life: “Rake”

http://en.wikipedia.org/wiki/Conway%27s_Game_of_Life

Conway's Game of Life

- A very simple example
  - Well suited to show the **concepts** of CAs

- Well studied
  - Pattern analysis of the Game of Life became its own science

- But - doesn’t demonstrate the **full power** of CAs