Genetic Algorithms and Genetic Programming

Michael Herrmann

Lecture **1**: Introduction (25/9/09)

School of Informatics

michael.herrmann@ed.ac.uk, phone: 0131 6 517177, Informatics Forum 1.42

Problem Solving at Decreasing Domain Knowledge

- Direct calculation, straight-forward recipe
- Solution by analogy, generalization
- Cartesian *method*, divide and conquer
- Iterative solution, continuous improvement
- Genetic algorithms, “suggestive” trial and error
- Random guessing

Paralipomena

- Theory of natural evolution
- Genetics, genomics, bioinformatics
- *The Philosophy of Chance* (Stanislaw Lem, *1968*)
- Memetics (R. Dawkins: *The Selfish Gene*, 1976)
- Neural Darwinism -- *The Theory of Neuronal Group Selection* (Gerald Edelman, 1975, 1989)
- (artificial) Immune systems
- Individual learning
- Computational finance, markets, agents

Prehistory of GA

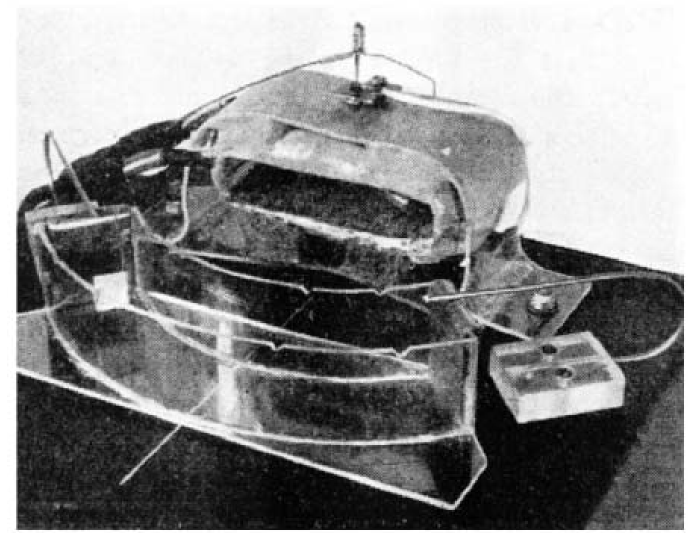
W. Ross Ashby (1903-1972)

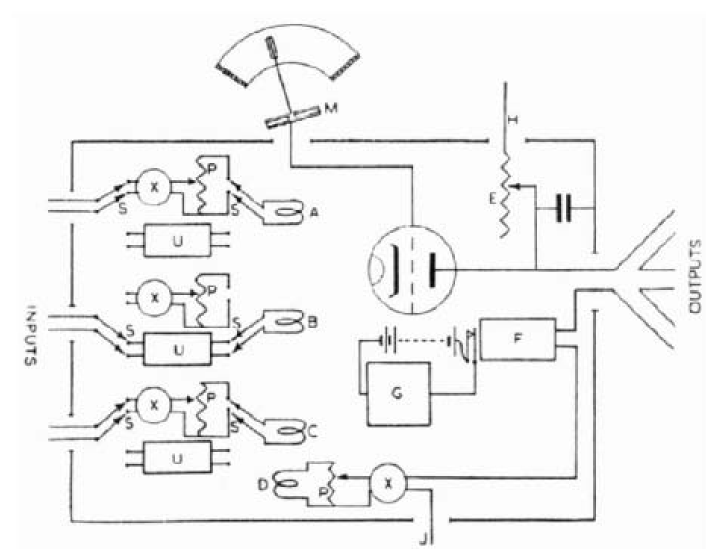
- Design for a brain (1952, 2nd edition 1960)
- An introduction to cybernetics (1956)

“who when asked what he wished done with his voluminous unpublished research notes responded characteristically with: ’Destroy it all’ (to give the next generation a chance for rediscovery)”

Ashby's Homeostat

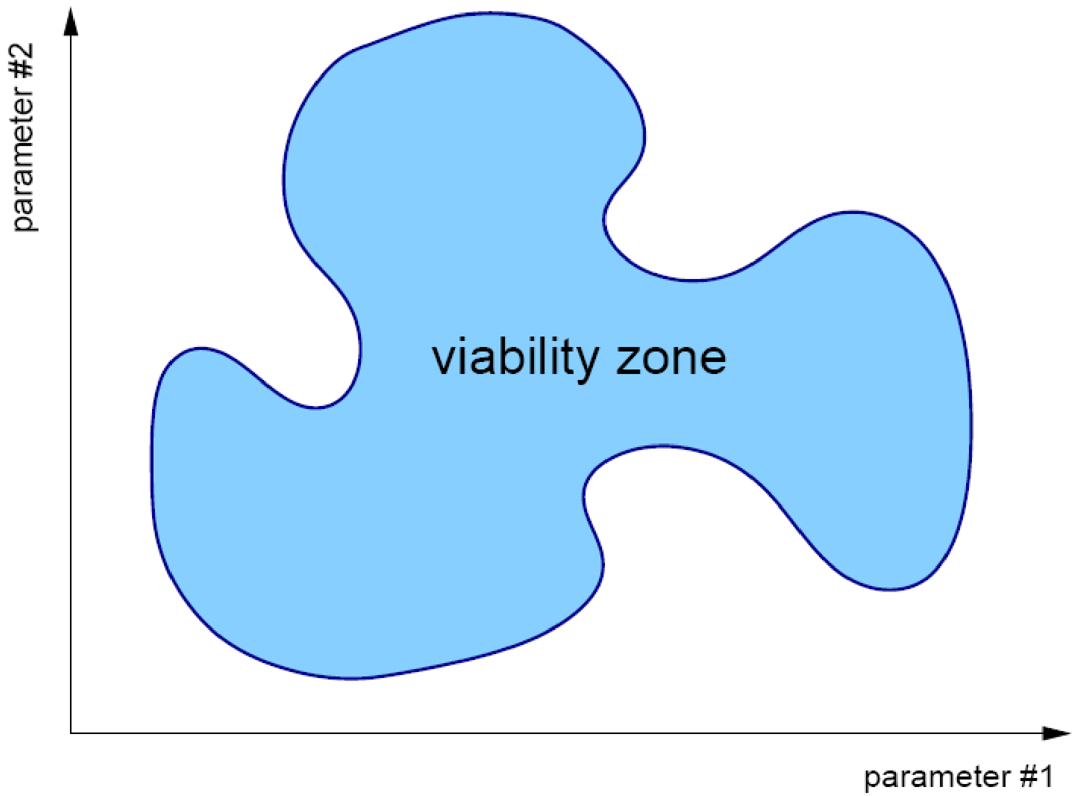
Was conceived as an implementation of regulatory mechanisms in living beings (body temperature, blood pressure, …)





Essential variables

- strongly linked to survival



Ashby's Homeostat

according to Zemanek & Hauenschild

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$$

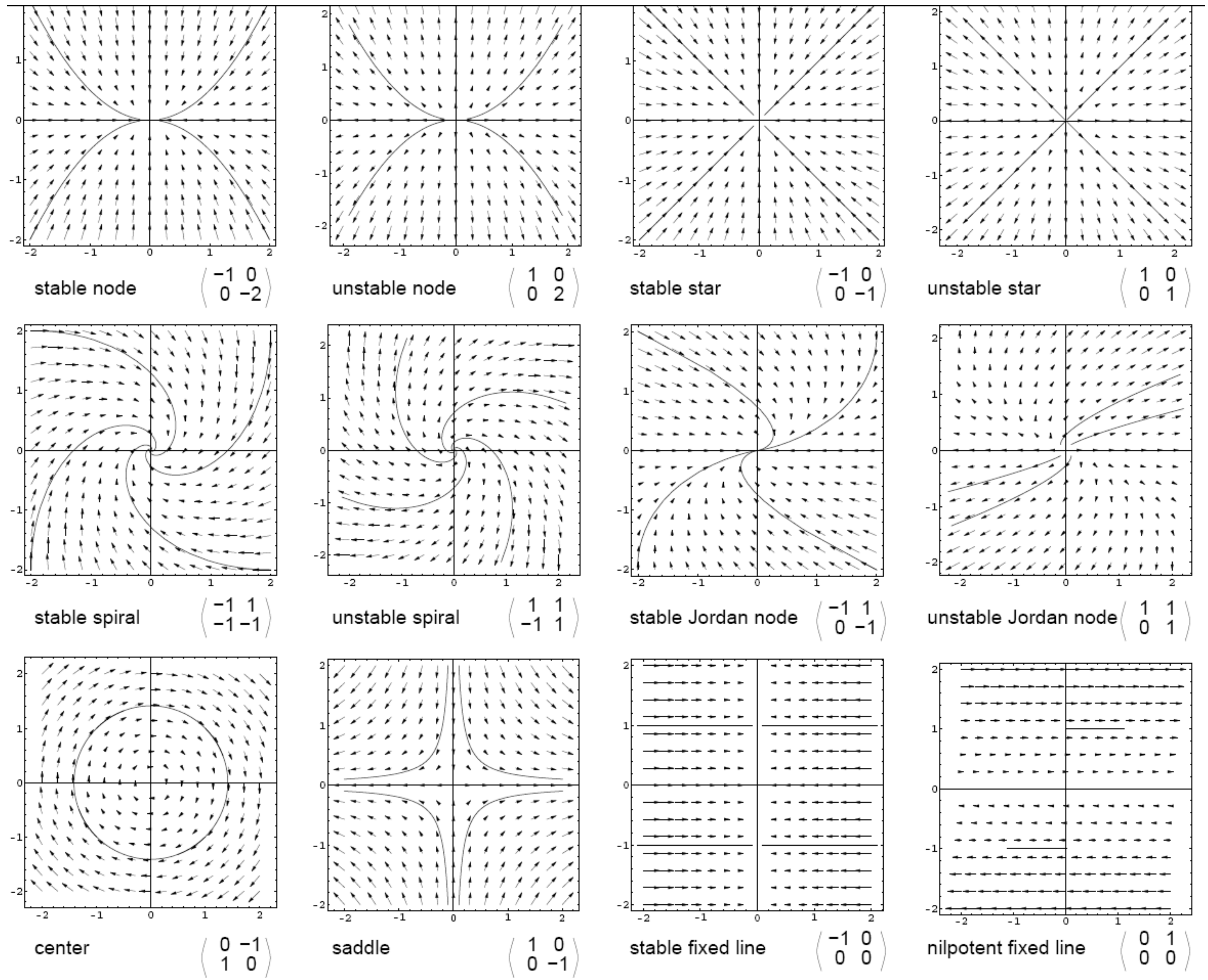
- stability if all eigenvalues of $\{a_{ij}\}$ have negative real parts
- if not: x_i reaches the critical surfaces $|x_i| = \theta = \frac{\pi}{4}$
- switching of a_{ij} for $j \neq i$ (3 entries)
- $a_{ij} \in \{0, \pm 0.48c, \pm 0.73c, \pm 0.89c, \pm c\}$, $i \neq j$, ($a_{ii} < 0$)
- 9^3 combinations per variable (only 25 used)
- total: $25^4 = 390625$ different dynamical behaviors

Different choices of the interaction matrix

$A=\{a_{ij}\}$

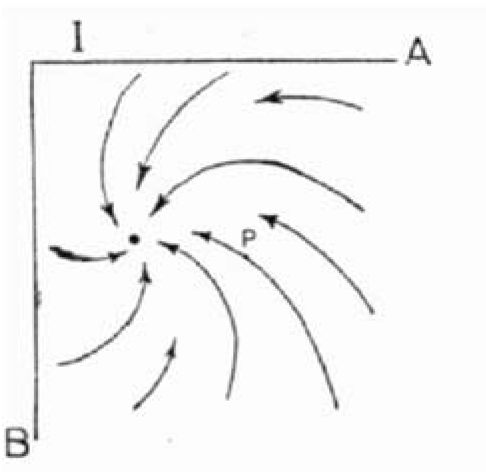
produce a lot of different effects

as implied by the 2D examples:

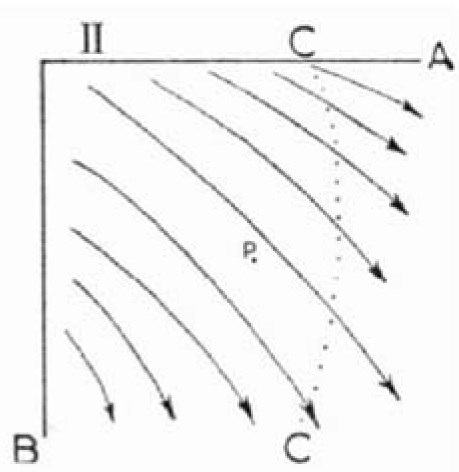


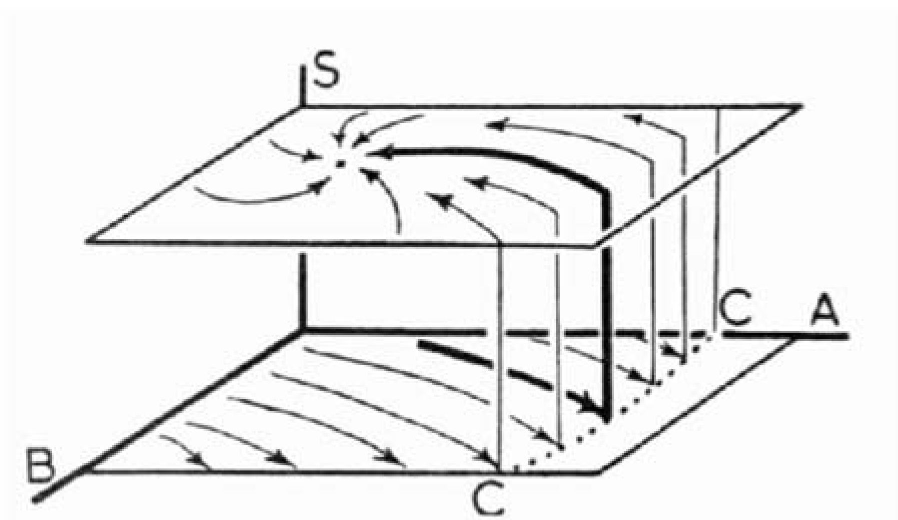
Switching dynamics

“good”



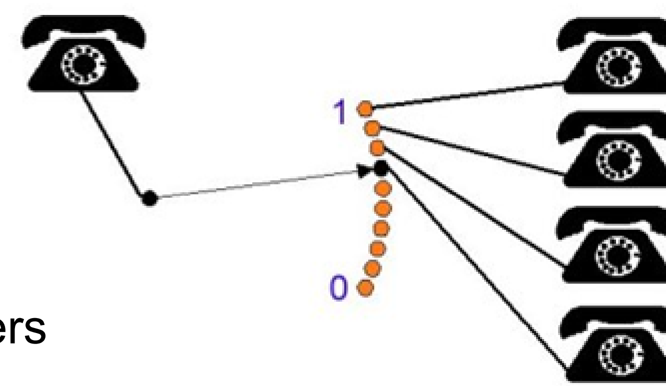
“bad”



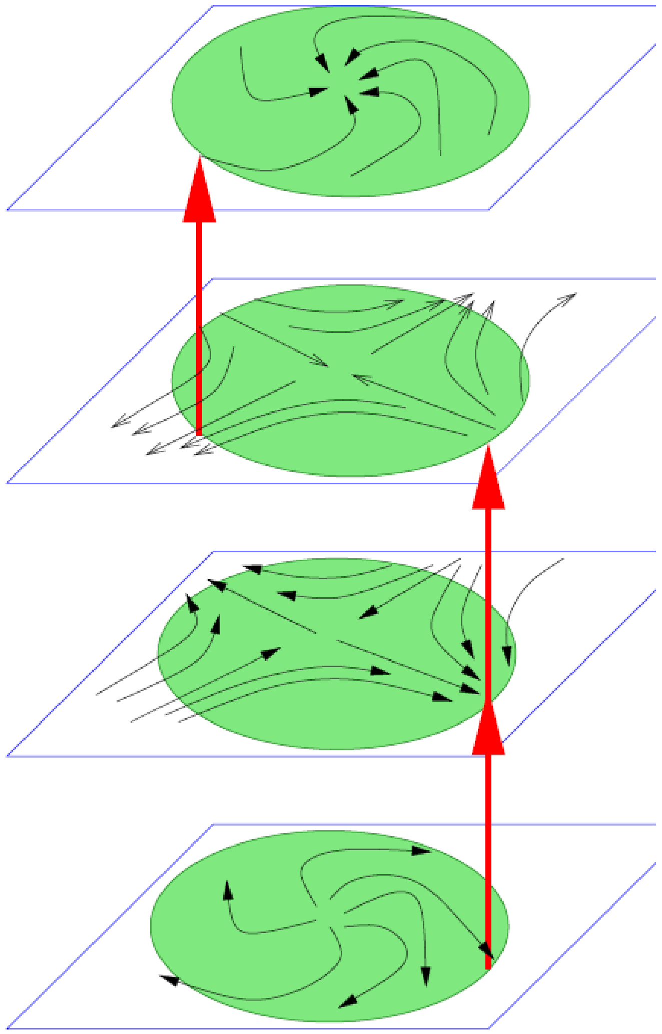


Choose different dynamics by selecting different parameters

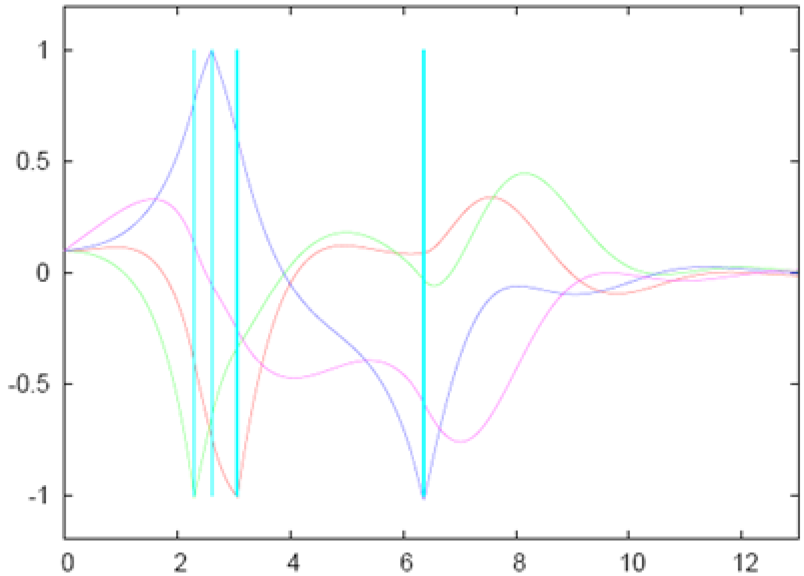
Implementation of switching in the homeostat

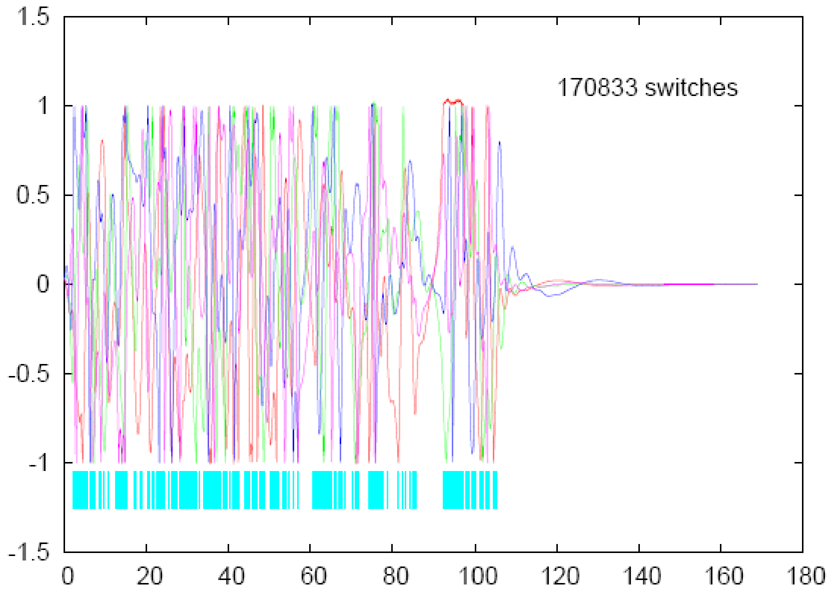


- The switching process stops if the coefficients cause all eigenvalues have negative real parts
- As the reverse of the sign of an a_{ij}, the system returns via a large unpredictable deviation in state space
- For random couplings, negativity of all eigenvalues is realized only with probability 2^{-n}



Discrete/Continous Dynamics of the Homeostat





Switching events (cyan) until stabilization for a homeostat with 4 elements (left) and 10 elements (right)

Translation: Homeostat → GA

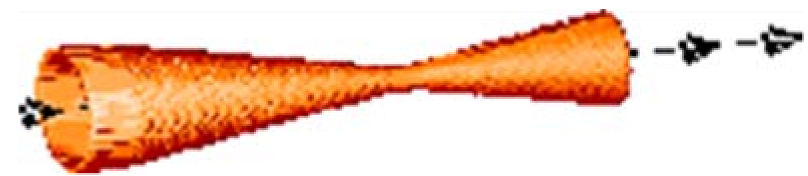
Homeostat	**Genetic algorithm**
Parameters $\{a_{ij}\}$	Genetic code
Viability	Fitness
Dynamics	Determination of fitness
Partial re-selection of new parameters	Mutation
--	**Recombination**

Experimental contour optimization of a supersonic flashing flow nozzle

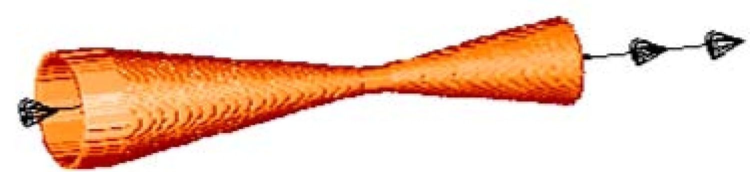
(1967-1969)

Hans-Paul Schwefel

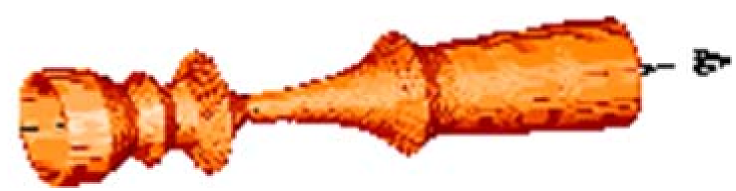
Start



Evolution



Result

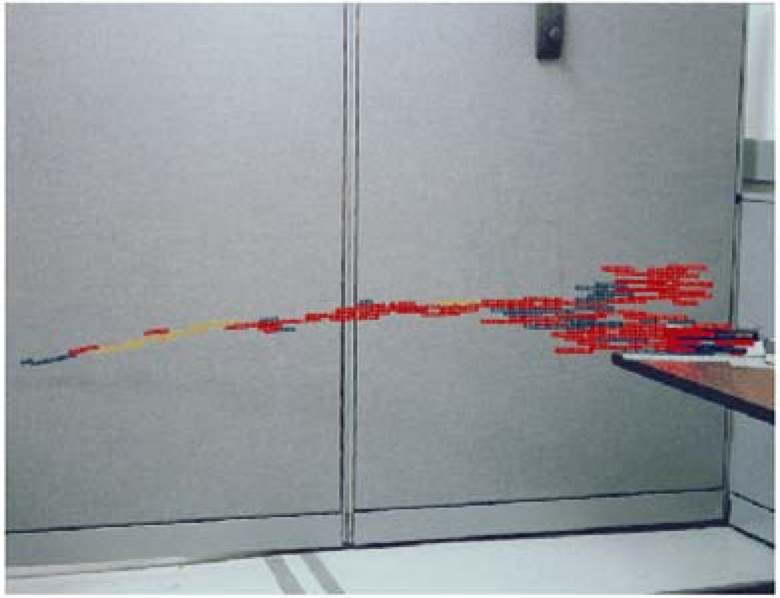


Genetic Algorithms

- global search heuristics
- technique used in computing
- find exact or approximate solutions to optimization problems

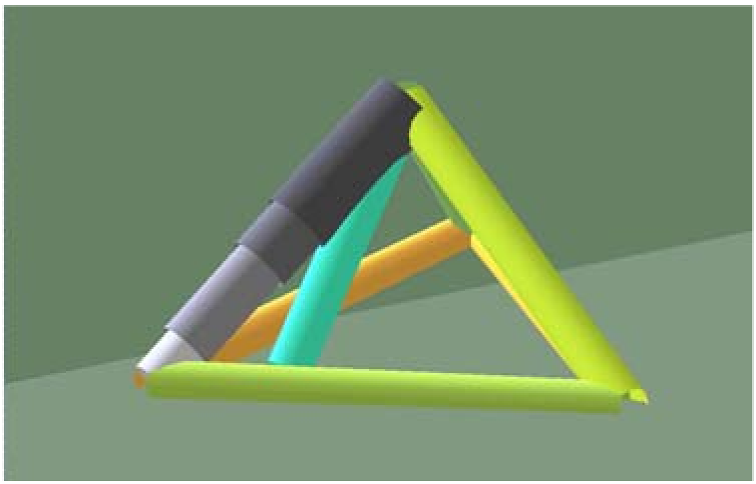
Applications in

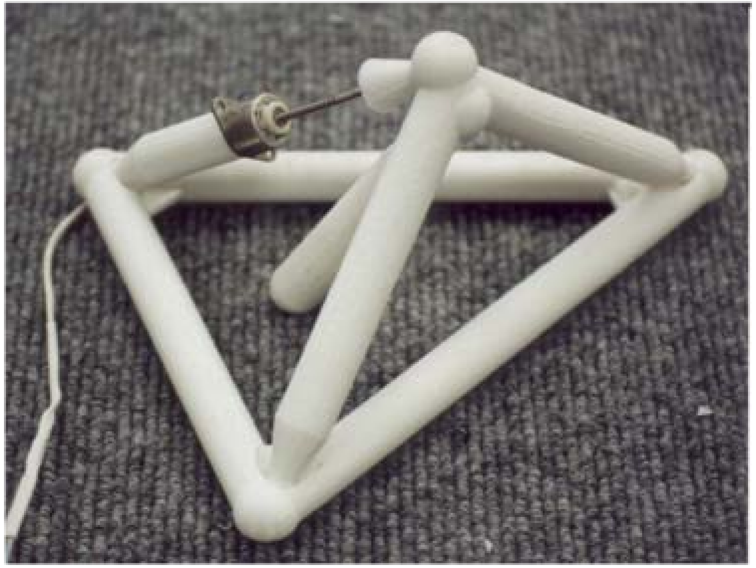
- Bioinformatics
- Phylogenetics
- Computational science
- Engineering
- Robotics
- Economics
- Chemistry
- Manufacturing
- Mathematics
- Physics

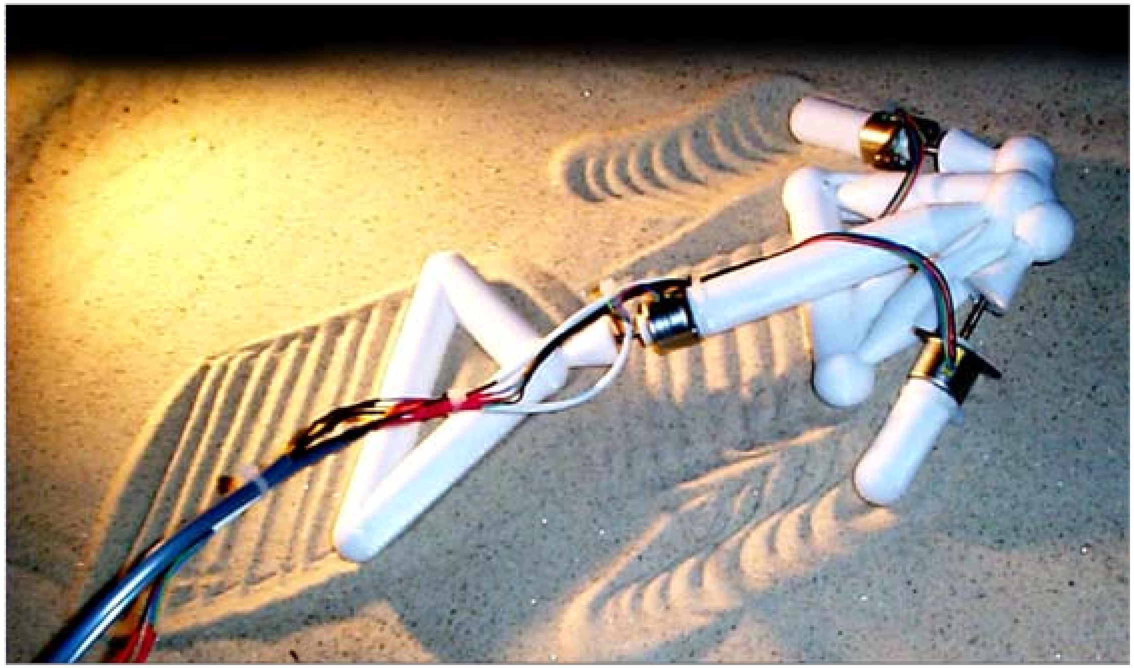


The Golem Project

Hod Lipson &
Jordan B. Pollack (2000)

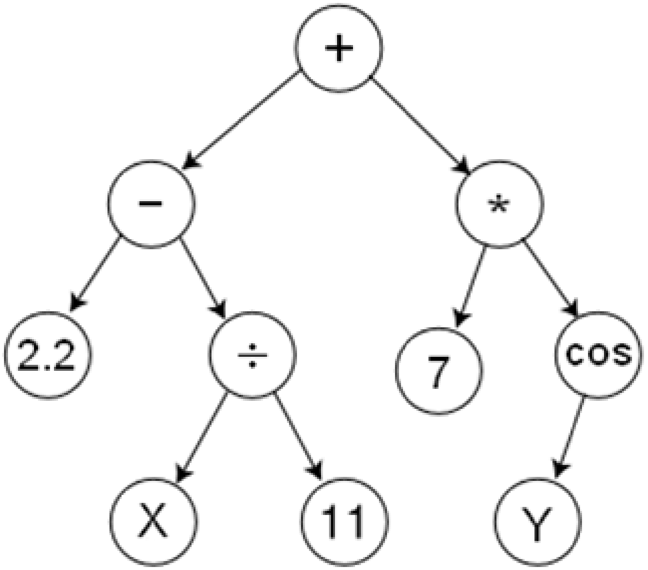






Genetic Programming (GP)

- Evolutionary algorithm-based methodology inspired by biological evolution
- Finds computer programs that perform a user-defined task
- Similar to genetic algorithms (GA) where each individual is a computer program
- Optimize a population of computer programs according to a fitness landscape determined by a program's ability to perform a given computational task.

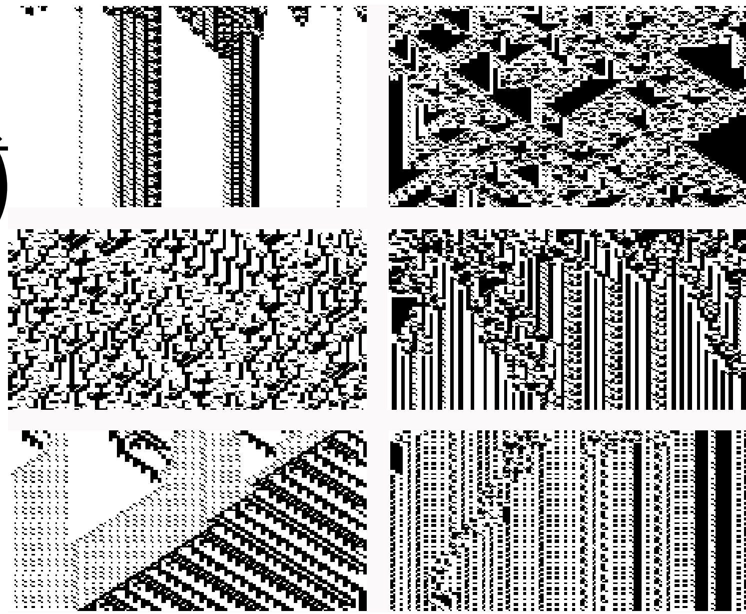


$$\left(2.2 - \left(\frac{X}{11}\right)\right) + \left(7 * \cos(Y)\right)$$

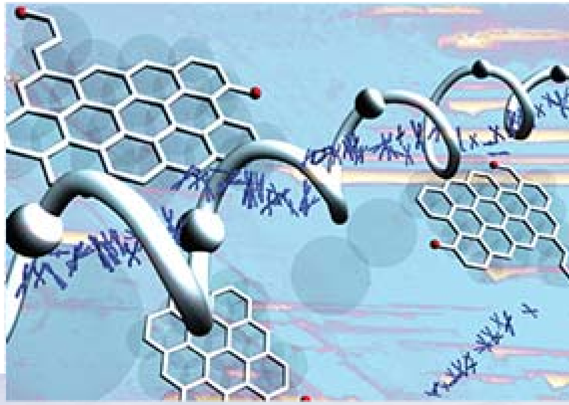
Evolutionary Computation (EC)

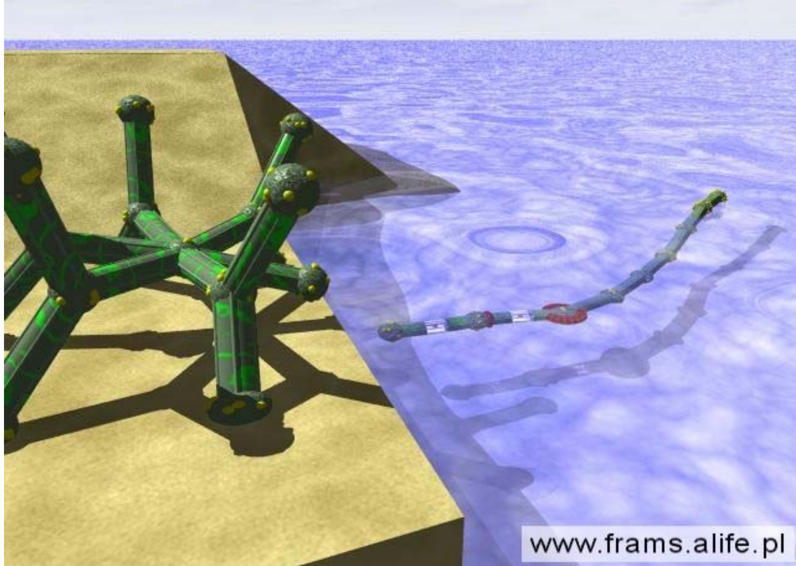
- Genetic algorithms: Solution of a problem in the form of strings of numbers using recombination and mutation
- Genetic programming: Evolution of computer programs
- Evolutionary programming: Like GP, but only the parameters evolve
- Evolution strategies: Vectors of real numbers as representations of solutions

Natural Computation (NC)



- Evolutionary Computation
- Artificial immune systems
- Neural computation
- Amorphous computing
- Ant colony optimization
- Swarm intelligence
- Harmony search
- Cellular automata
- Artificial life
- Membrane computing
- Molecular computing
- Quantum computing





Particular emphasis on:

- Optimization, optimization, optimization
- Evolutionary robotics
- Relation between artificial and natural evolution
- Using background knowledge: Encoding and construction of fitness functions
- Natural computing

Problem Solving as Optimization

Choosing the best option from some set of available alternatives

- Minimize energy, time, cost, risk, …
- Maximize gains, acceptance, turnover, …
- Discrete cost:
  - admissible goal state: maximal gain
  - anything else: no gain
- Secondary costs for:
  - acquisition of domain knowledge
  - testing alternatives
  - doing nothing
  - determining costs

Syllabus

- Part 1: Introduction
  - Introduction to Genetic Algorithms: an example
  - Genetic Algorithms: biological inspiration

- Part 2: Genetic Algorithms (GAs)
  - The canonical genetic algorithm
  - The schema theorem and building block hypothesis
  - Formal analysis of genetic algorithms
  - Methodology for genetic algorithms
  - Designing real genetic algorithms

Syllabus (continued)

- Part 3: Optimisation Problems
  - Solving optimisation problems
  - Swarm intelligence: ant colony optimisation (ACO)
  - Adding local search: hybrid GAs and hybrid ACO
  - Other methods: simulated annealing, tabu search
- Part 4: Evolving Programs and Intelligent Agents
  - Evolving programs: genetic programming
  - Evolving controllers: neural networks and robots
  - Evolving intelligence: agents that play games
  - Evolving intelligence: programs that can plan

- Tuesday & Friday 15:00 – 15:50 at AT LT2
- Reading: From supplied course notes and set book (An Introduction to Genetic Algorithms by Melanie Mitchell, MIT Press 1998, available on amazon.com, also available on MIT CogNet) – see Informatics library website See http://www.lib.ed.ac.uk/resbysub/info/ebooks.shtml
- Two assignments: the first one unmarked the second one marked and worth 25% of the course mark, to be handed in at the end of Week 5 and the end of Week 10.
- Exam: worth 75% of the course mark, taken at the end of Semester 2 (for visiting students: end of S1)
- michael.herrmann@ed.ac.uk
phone: 0131 6 517177, Informatics Forum 1.42

Tutorials

- Mondays
  - group 1: 16:10-17:00 (AT 5:03)
- Tuesdays
  - group 2: 13:05-13:55 (AT 5.07)
- Wednesdays
  - group 3: 12:10-13:00 (AT 5.03)
  - group 4: 12:10-13:00 (AT 5:07)
  - group 5: 13:05-13:55 (AT 5:03)
  - group 6: 13:05-13:55 (AT 5:07)