USC VICERREITORÍA DE CULTURA — UNIVERSIDADE DE VERÁN 2010

Inteligencia Artificial

Santiago de Compostela
27-30 de Julio de 2010

¿Ciencia, tecnología, ficción o marketing?

Computación Bioinspirada: Cuando la informática imita a los seres vivos

Francisco Herrera

Research Group on Soft Computing and Information Intelligent Systems

(SCI²S) http://sci2s.ugr.es

Dept. of Computer Science and A.I.

University of Granada, Spain

Email: herrera@decsai.ugr.es

http://decsai.ugr.es/~herrera

DECSAI
Universidad de Granada

Introducción

Los sistemas indiscutiblemente inteligentes en este planeta son biológicos.

Los seres vivos y los organismos biológicos, frente a las exigencias de su medio ambiente, utilizan soluciones diferentes a los enfoques tradicionales de la ingeniería humana para resolver problemas. Los sistemas biológicos tienden a ser adaptables, distribuidos, cooperativos, …

¿Pueden estos sistemas biológicos y los procesos que los crearon aportar lecciones que nos ayuden a diseñar sistemas artificiales inteligentes?

La respuesta es "SI".

Introducción

¿Pueden estos sistemas biológicos y los procesos que los crearon aportar lecciones que nos ayuden a diseñar sistemas artificiales inteligentes?

La respuesta es "SI".

Entre los modelos biológicos que nos permiten crear sistemas inteligentes nos encontramos modelos de imitación de los sistemas nerviosos e inmunológicos dando lugar a los modelos computaciones de redes neuronales y sistemas inmunológicos, la evolución natural en la que están basados los algoritmos evolutivos, ...

El diseño y aplicación de métodos de computación que modelan/imitan los principios biológicos de la naturaleza ha dado lugar al área de la inteligencia artificial denominada **Computación Bioinspirada.**

Introducción

¿Pueden estos sistemas biológicos y los procesos que los crearon aportar lecciones que nos ayuden a diseñar sistemas artificiales inteligentes?

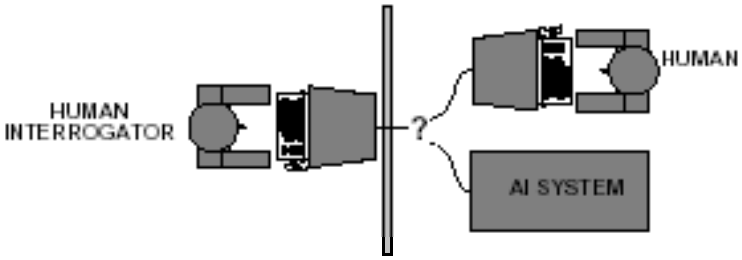
La respuesta es "SI".

En esta conferencia introduciremos brevemente la Computación Bioinspirada, analizaremos sus orígenes, los modelos y mostraremos algunas de sus aplicaciones.

Bioinspired Algorithms and Potted history of AI(1/3)

1943 McCulloch&Pitts: Boolean circuit model of brain

1950 Turing's "Computing Machinery and Intelligence:



1950s Early AI programs, including Samuel's checkers program, Newell & Simon's Logic Theorist, Gelernter's Geometry Engine

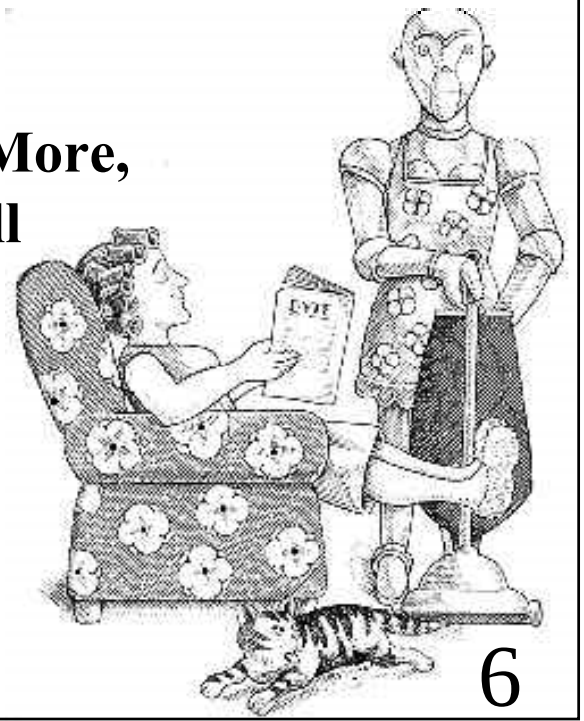
1956 Dartmouth meeting:

"Artificial Intelligence" adopted

The birth of "Artificial Intelligence"

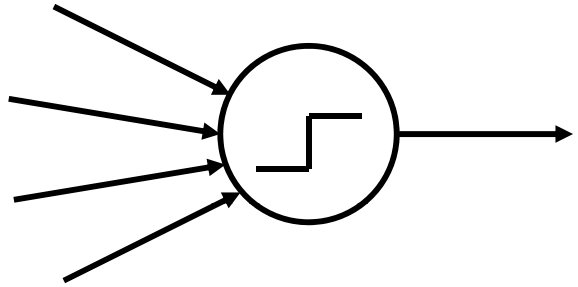
- **John McCarthy used the term "Artificial Intelligence" for the first time as the topic of the Dartmouth conference in 1956.**
  - **Venue:**
    - **Dartmouth College, Hanover, state New Hamphshire, USA**
  - **Organizers:**
    - **John McCarthy, Marvin Minsky, Nathaniel Rochester, and Claude Shannon**
  - **Participants:**
    - **Ray Solomonoff, Oliver Selfridge, Trenchard More, Arthur Samuel, Herbert Simon, and Allen Newell**
  - **Proposal:**

    To prove that every aspect of learning or any other feature of intelligence can be so precisely described that a machine can be made to simulate it.



The First Biologically inspired model: Artificial Neural Networks

- **Warren McCulloch and Walter Pitts**
  - **Model of artificial neuron (1943)**
  - **Neuron represents functions.**



- **Donald Olding Hebb**
  - **Rule for neural network training (1949)**
- **Marvin Minsky and Dean Edmonds have built the first computer with neural network.**
  - **SNARC (1951)**

Bioinspired Algorithms and Potted history of AI(2/3)

1965 Robinson's complete algorithm for logical reasoning

1966-74 AI discovers computational complexity, Neural network research almost disappears

1969-79 Early development of knowledge-based systems

1980-88 Expert systems industry booms

1988-93 Expert systems industry busts: "AI Winter"

1995- Agents

Bioinspired Algorithms and Potted history of AI(3/3)

1985-95 Neural networks return to popularity

1988- Resurgence of probability; general increase in technical depth "Nouvelle AI": ALife, GAs, Soft Computing, Bio-inspired Computation

Contents

Contents

Introduction: Bio-inspired Computation

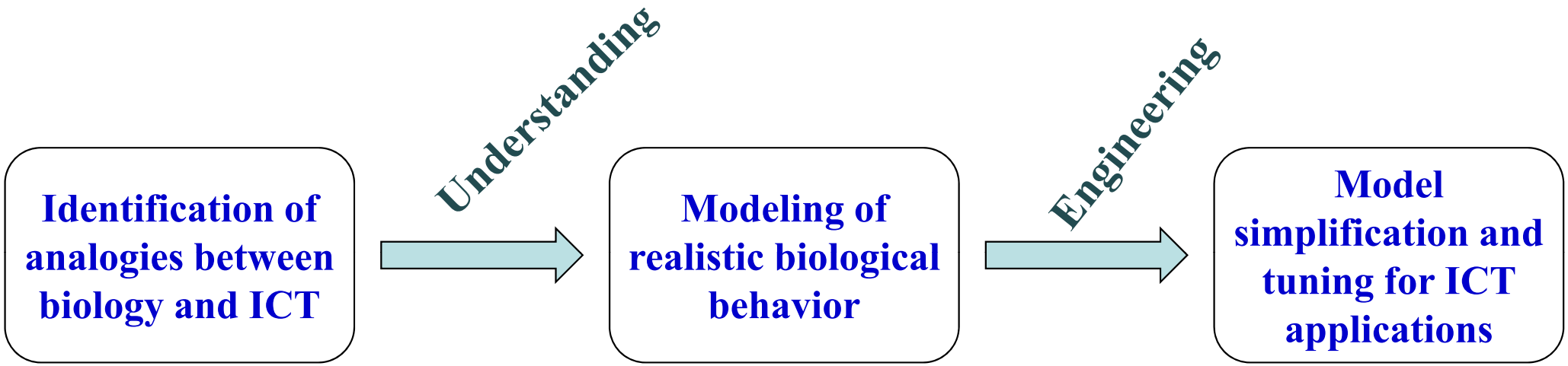
The term “bio-inspired”

- **The term *bio-inspired* has been introduced to demonstrate the strong relation between a particular system or algorithm, which has been proposed to solve a specific problem, and a biological system, which follows a similar procedure or has similar capabilities.**

Introduction: Bio-inspired Computation

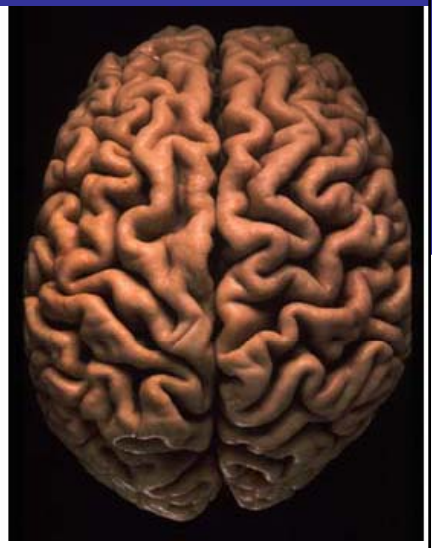
The design of bio-inspired solutions

- **Identification of analogies**
  - In nature or biology and Information Technology based systems
- **Understanding**
  - Computer modeling of realistic biological behavior
- **Engineering**
  - Model simplification and tuning for IT applications



Contents

Neural Netwoks

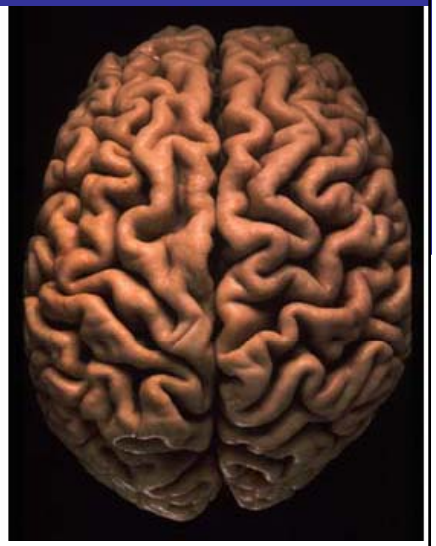


The brain as a computer

Higher level functions in animal behaviour

- Gathering data (sensation)
- Inferring useful structures in data (perception)
- Storing and recalling information (memory)
- Planning and guiding future actions (decision)
- Carrying out the decisions (behaviour)
- Learning consequences of these actions

Neural Netwoks



The brain as a computer

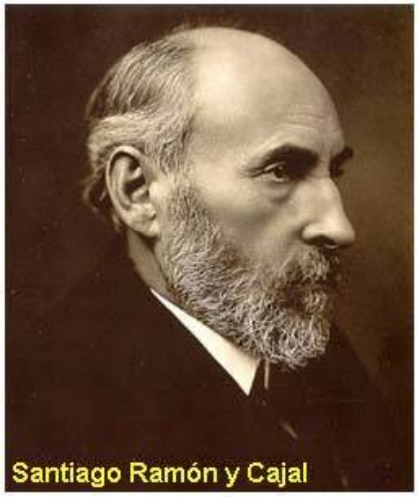
Hardware functions and architectures

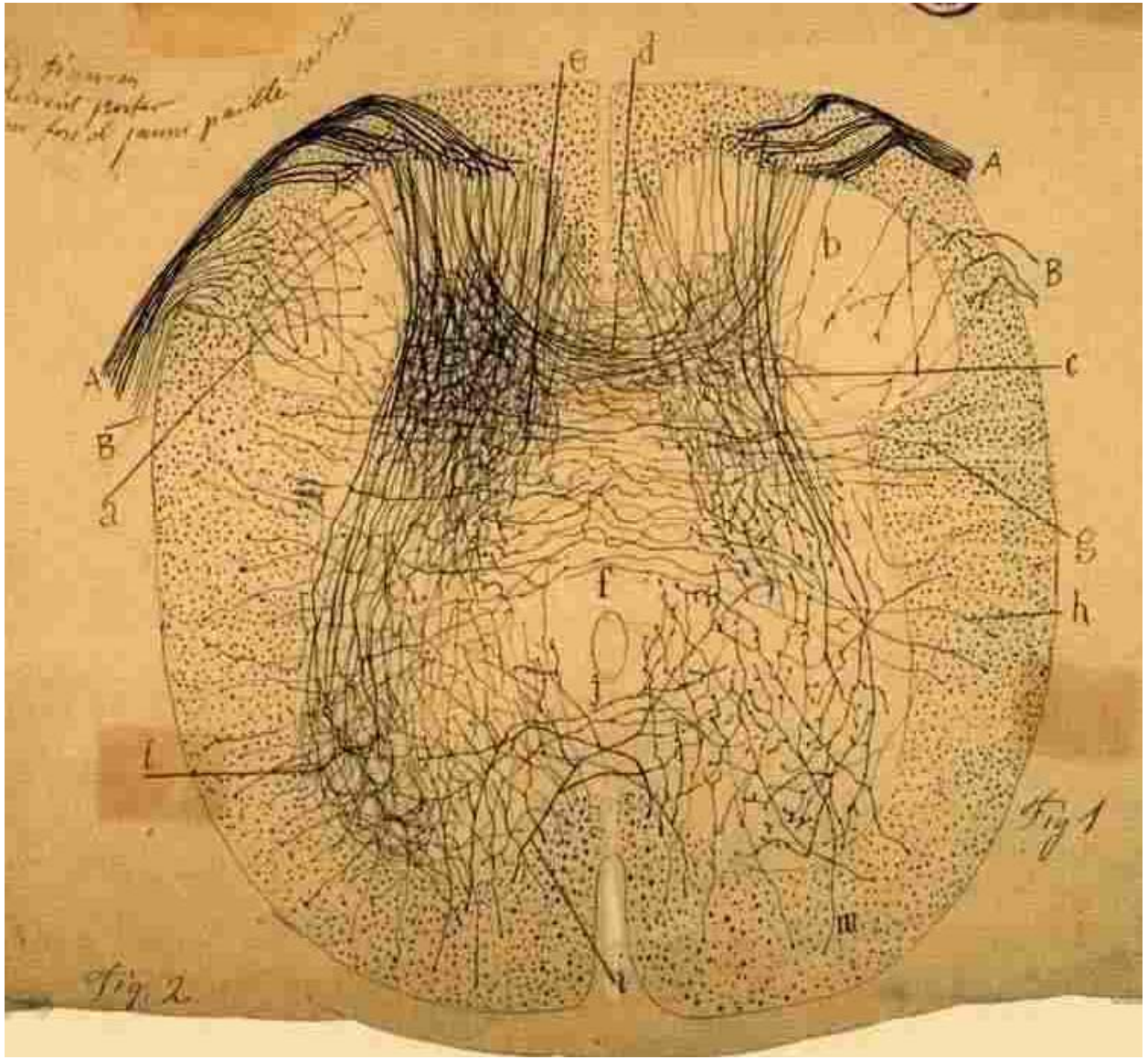
- 10 billion neurons in human cortex
- 10,000 synapses (connections) per neuron
- Machine language: 100mV, 1-2msec spikes (action potential)
- Specialised regions & pathways (visual, auditory, language…)

Neural Netwoks

The neuron doctrine

Ramon y Cajal (1899)



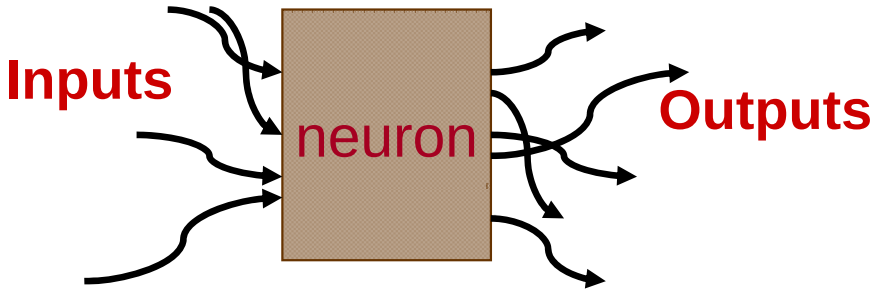


Neural Netwoks

1) Neurons are cells: distinct entities (or agents).

2) Inputs & outputs are received at junctions called synapses.

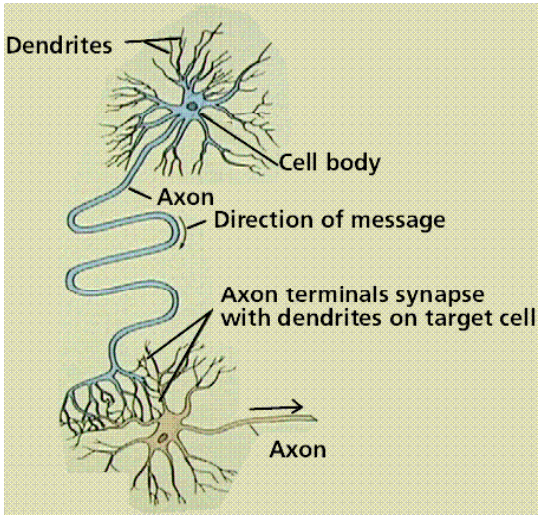
3) Input & output ports are distinct. Signals are uni-directional from input to output.

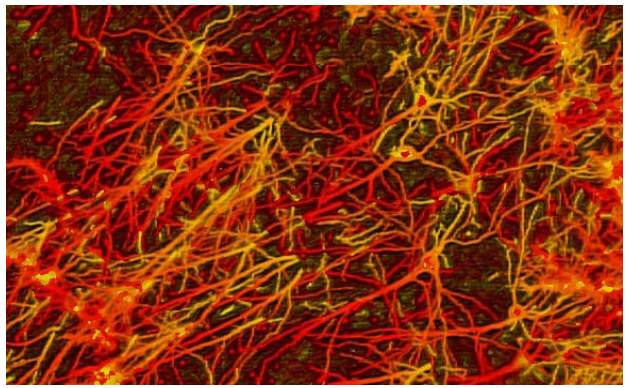


Today, neurons (or nerve cells) are regarded as the basic information processing unit of the nervous system.

Neural Netwoks

How brains seem to do pattern recognition



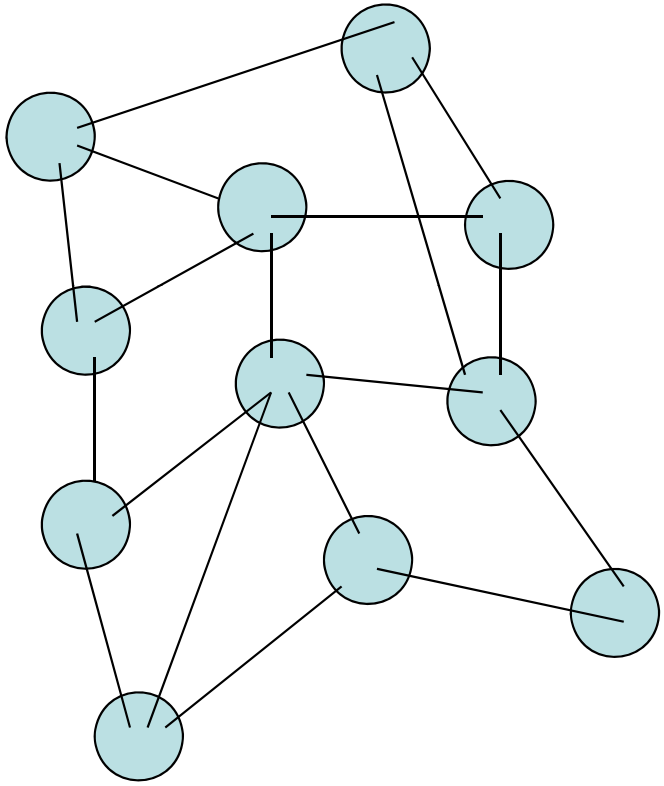


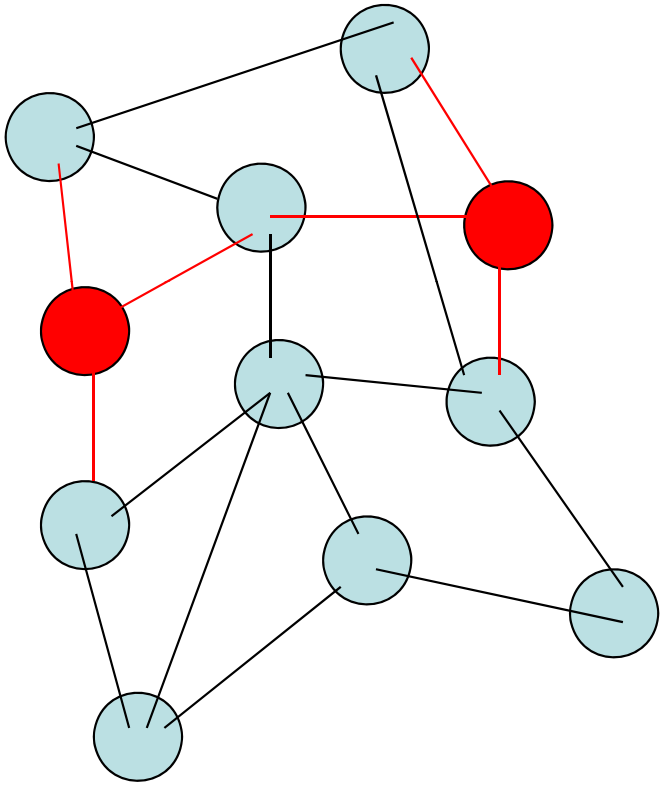
The business end of **this** is made of lots of **these** joined in networks like **this**

Much of our own “computations” are performed in/by this network

Neural Netwoks

The key idea in brain-inspired computing



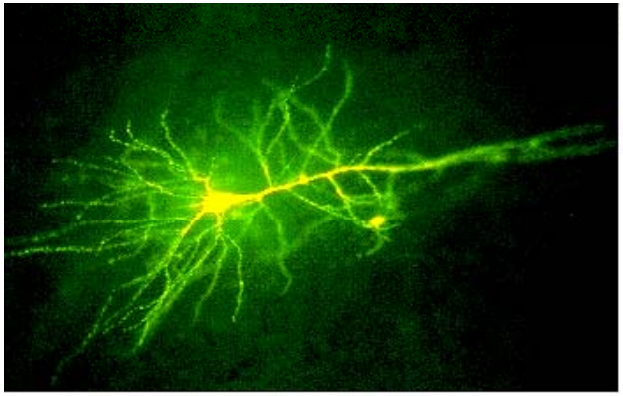


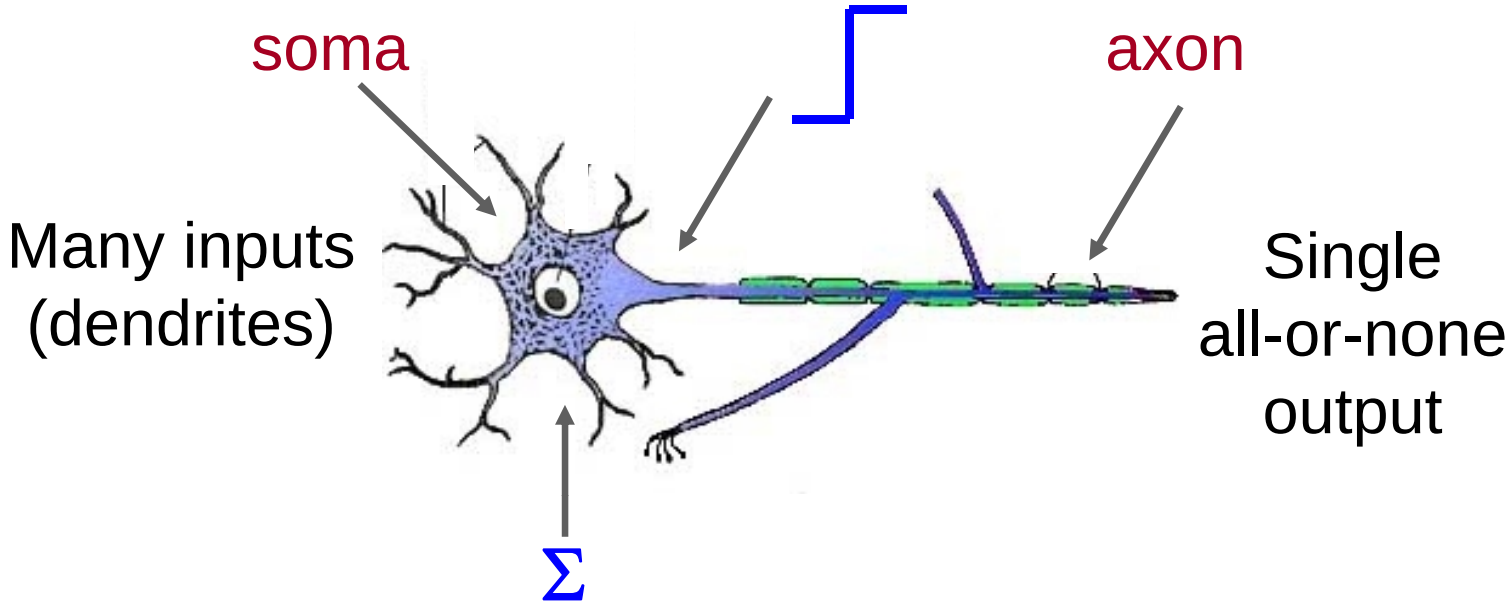
The brain is a complex tangle of neurons, connected by synapses

When neurons are active, they send signals to others.

Neural Netwoks

Computation of a pyramidal neuron



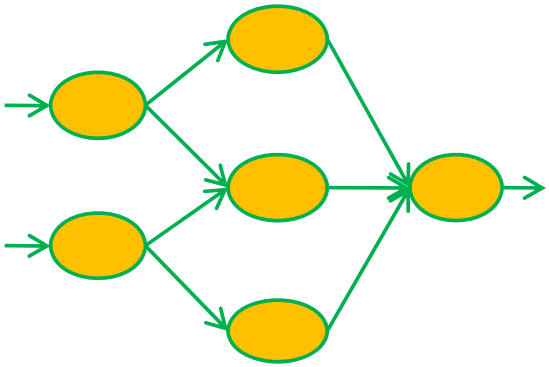


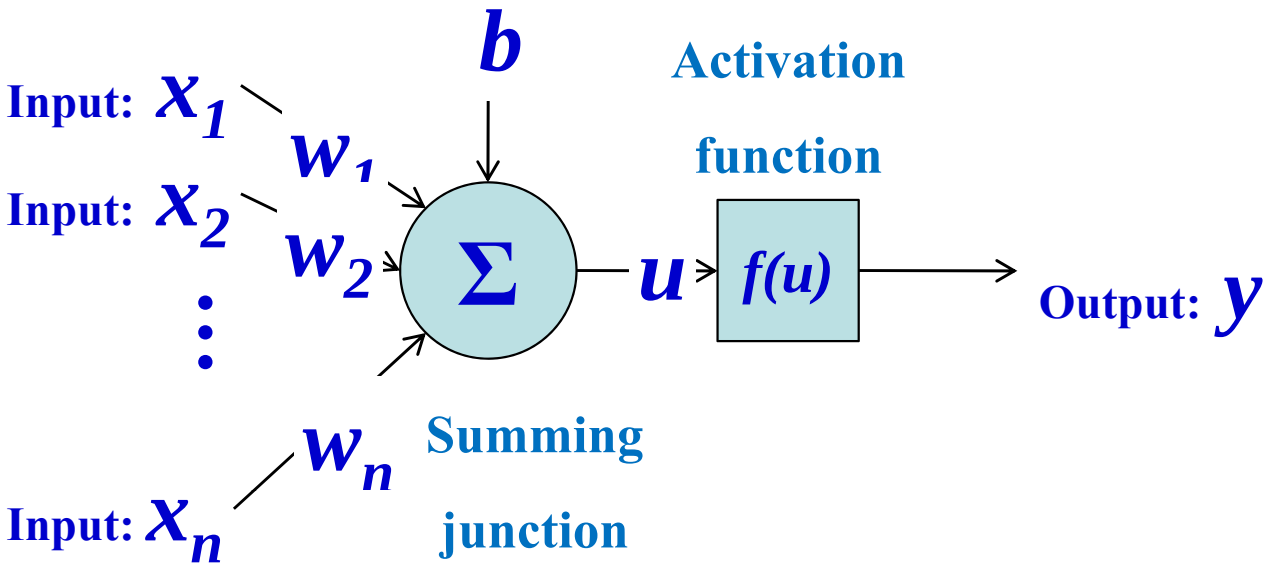
Neural Netwoks

Bio-inspired research – ANNs

- **Artificial neural networks (ANNs)**
  - **Primary objective of an ANN is to acquire knowledge from the environment**

    → self-learning property





Neural Netwoks

Early history (1943)

McCulloch & Pitts (1943). "A logical calculus of the ideas immanent in nervous activity", *Bulletin of Mathematical Biophysics*, 5, 115-137.

In this seminal paper, Warren McCulloch and Walter Pitts invented the first artificial (MP) neuron, based on the insight that a nerve cell will fire an impulse only if its threshold value is exceeded. MP neurons are hard-wired devices, reading pre-defined input-output associations to determine their final output. Despite their simplicity, M&P proved that a single MP neuron can perform universal logic operations.

A network of such neurons can therefore do anything a Turing machine can do, but with a much more flexible (and potentially very parallel) architecture.

Neural Netwoks

Neural Computing

Pattern recognition using *neural networks* is the most widely used form of BIC in industry and science. We will learn about the most common and successful types of neural network.

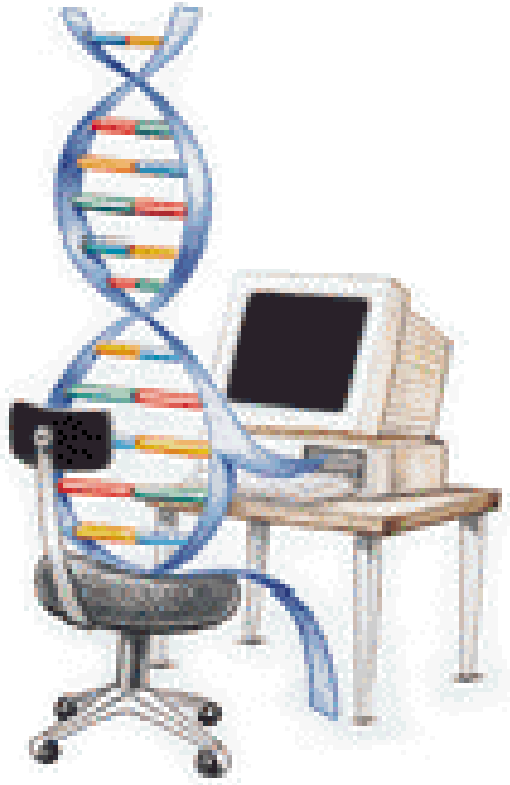
This is Stanley, winner of the DARPA grand Challenge – a great example of bio-inspired computing winning over all other entries, which were largely `classical'

Contents

Evolutionary Computation: A New Way to Search for Solutions

Genetic algorithms

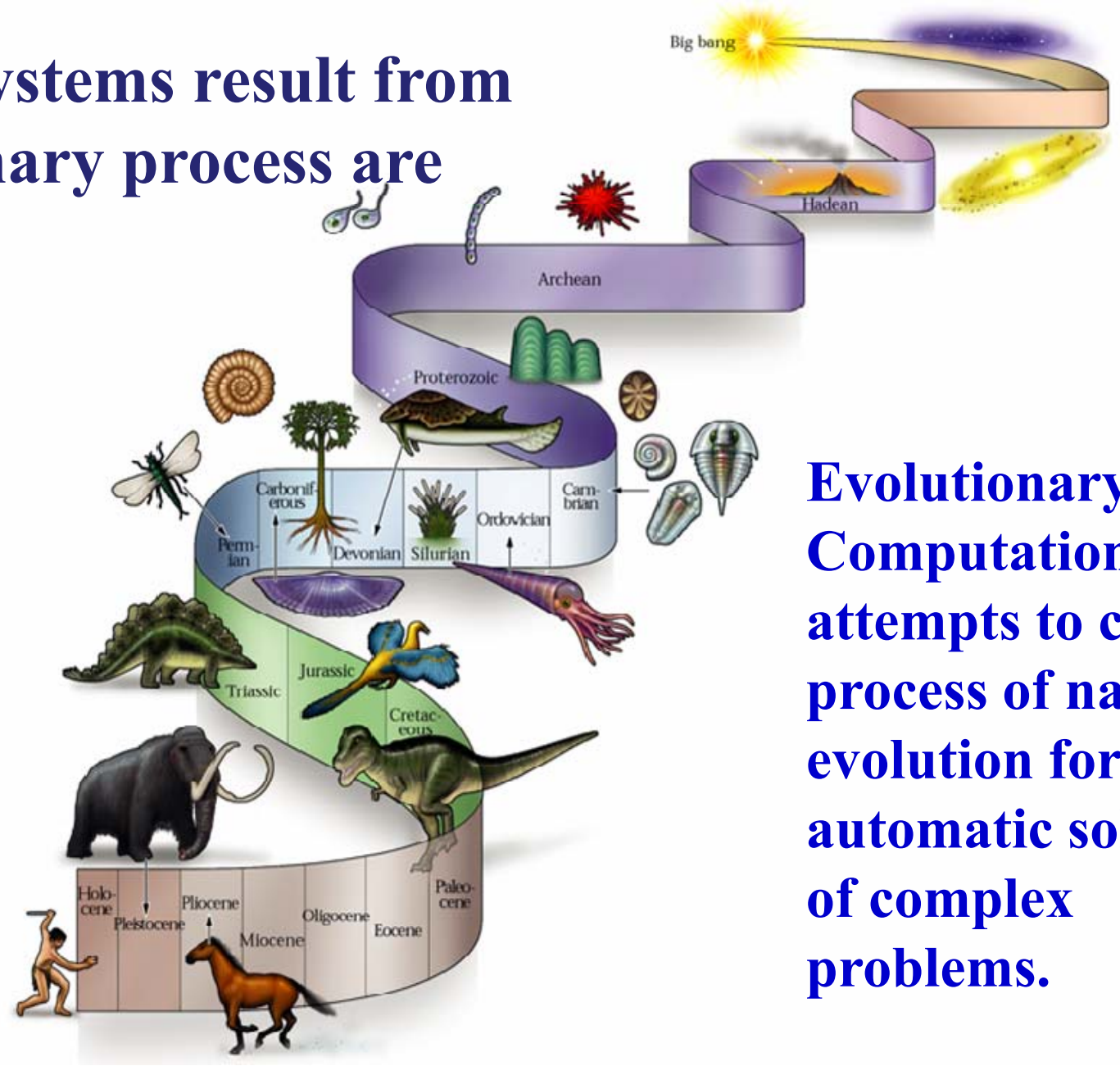
They are optimization algorithms,
search
and learning
inspired in the process of
Natural and Genetic Evolution



Evolutionary Computation: A New Way to Search for Solutions

Biological systems result from an evolutionary process are

- **robust**
- **complex**
- **adaptive**



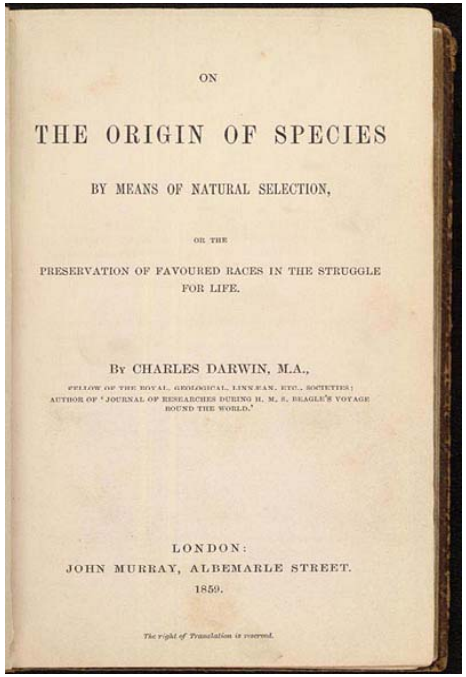
Evolutionary Computation attempts to copy process of natural evolution for automatic solution of complex problems.

Evolutionary Computation: A New Way to Search for Solutions

All species derive from common ancestor

Charles Darwin, 1859

On the Origins of Species



ON

THE ORIGIN OF SPECIES

BY MEANS OF NATURAL SELECTION,

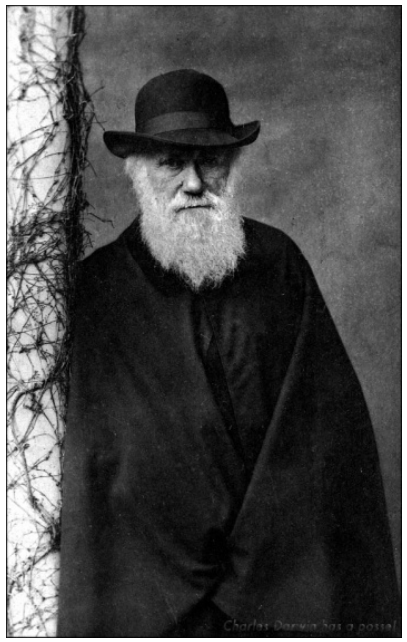
OR THE

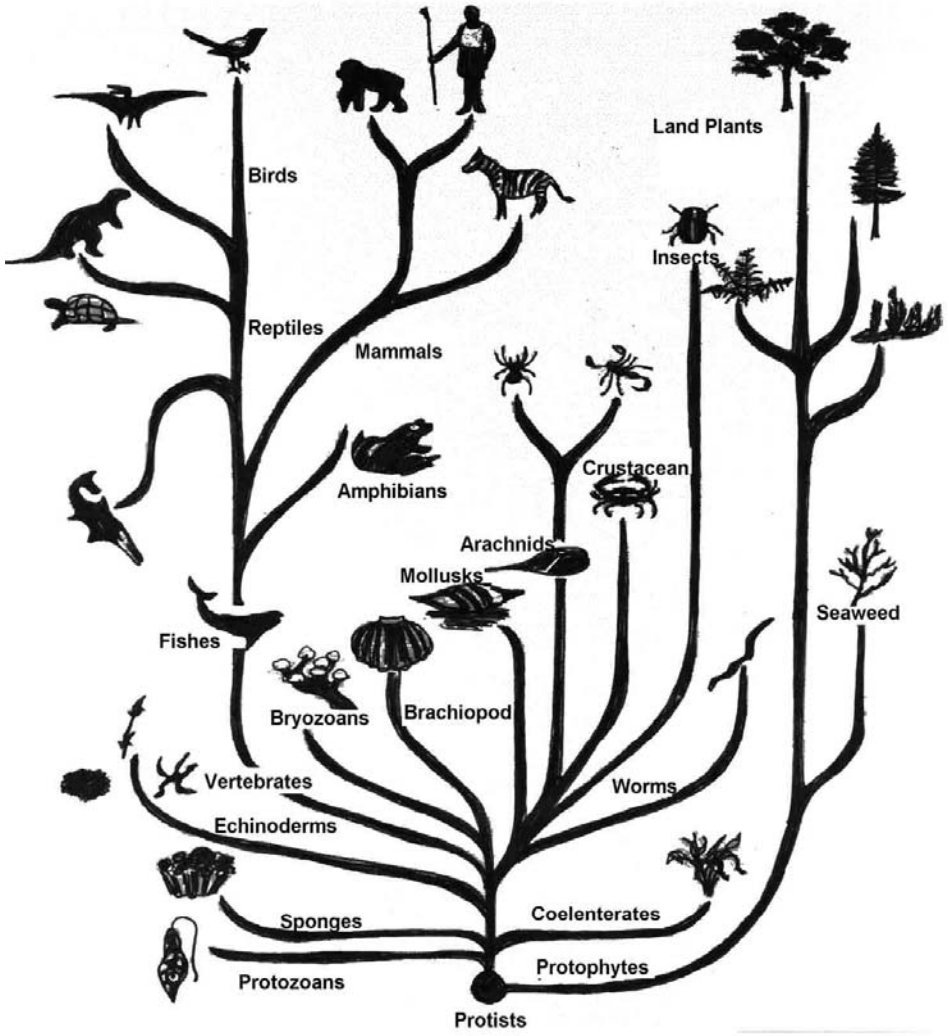
PRESERVATION OF FAVOURED RACES IN THE STRUGGLE FOR LIFE.

By CHARLES DARWIN, M.A.,

FELLOW OF THE ROYAL, GEOLOGICAL, LINNÆAN, ETC., SOCIETIES; AUTHOR OF 'JOURNAL OF RESEARCHES DURING H. M. S. BEAGLE'S VOYAGE ROUND THE WORLD.'

LONDON:
JOHN MURRAY, ALBEMARLE STREET.
1859.





Evolutionary Computation: A New Way to Search for Solutions

The 4 Pillars of Evolution

Population

Group of several individuals

Diversity

Individuals have different characteristics

Heredity

Characteristics are transmitted over generations

Selection

- Individuals make more offspring than the environment can support
- Better at food gathering = better at surviving = make more offspring

Evolutionary Computation: A New Way to Search for Solutions

Artificial Evolution

Automatic generation of solutions to hard problems

Similarities between natural and artificial evolution:

- Phenotype (computer program, object shape,, robot, etc.)
- Genotype (genetic representation of the phenotype)
- Population
- Diversity
- Selection
- Inheritance

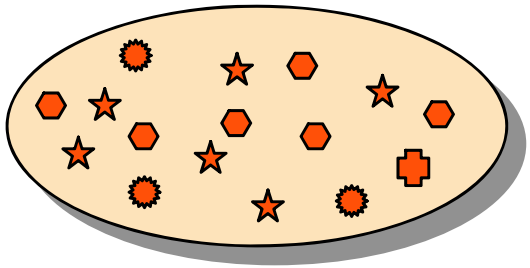
Differences between natural and artificial evolution:

- **Fitness is measure of performance of the individual solution to the problem**
- **Selection of the best according to performance criterion (fitness function)**
- **Expected improvement between initial and final solution**

Evolutionary Computation: A New Way to Search for Solutions

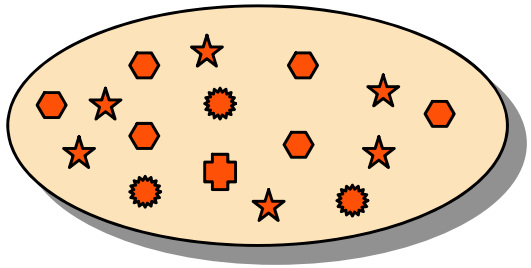
The ingredients

t

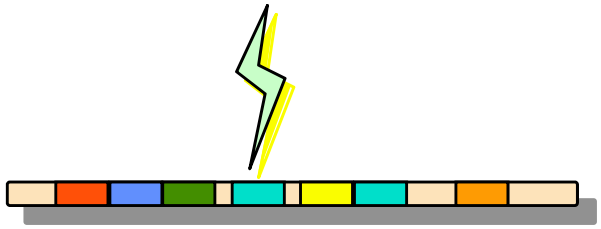


reproduction

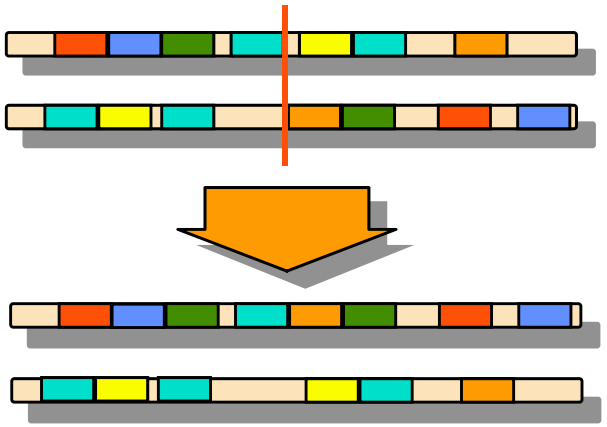
selection



t + 1



mutation



Crossover(or recombination)

Evolutionary Computation: A New Way to Search for Solutions

Genetic Algorithm Structure

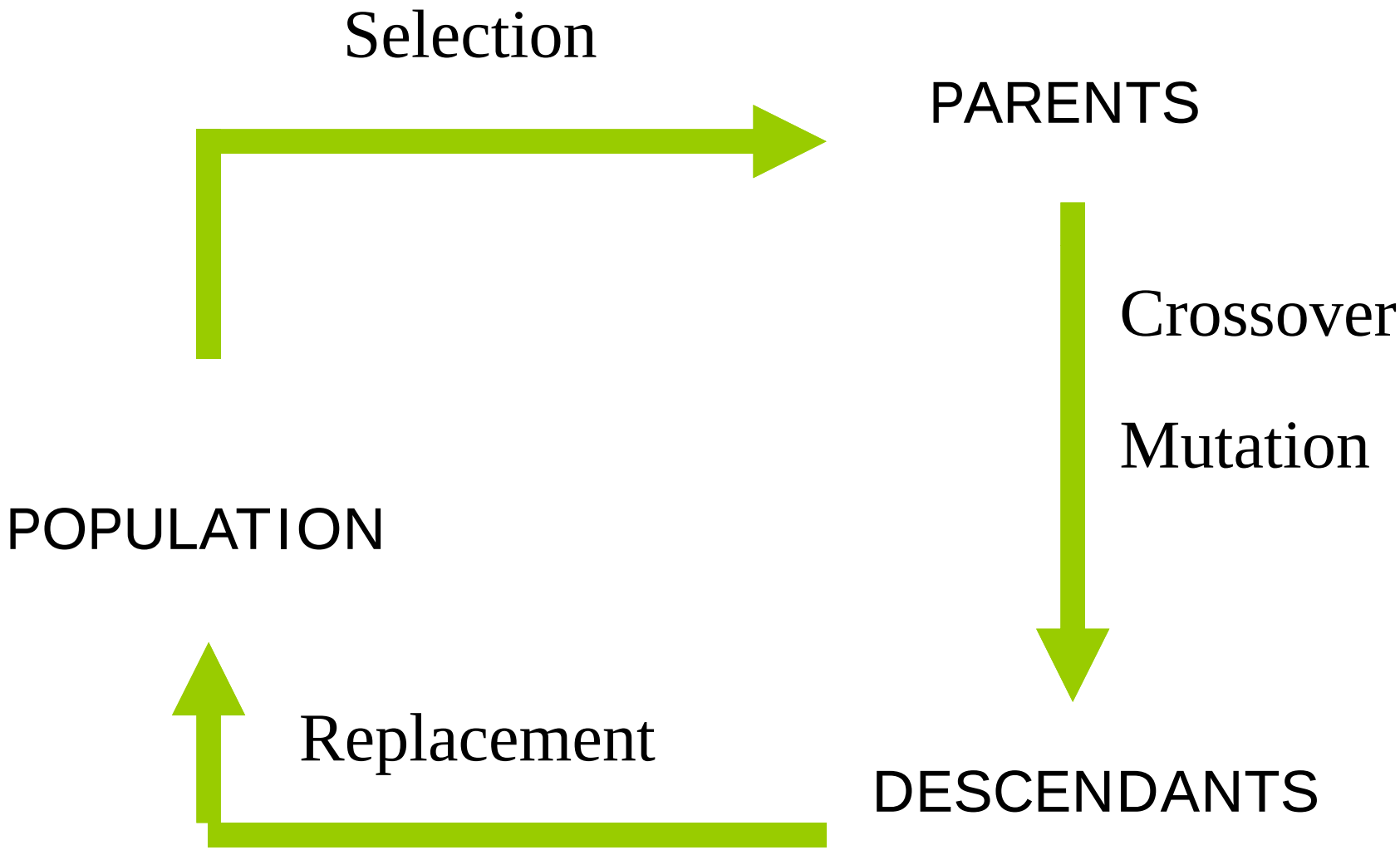
Basic Genetic Algorithms

```
Beginning (1)
    t = 0
    Initialization P(t)
    evalution P(t)

    While (the stop contition is not verified) do
    Beginning (2)
      t = t + 1
      selection P'(t) from P(t-1)
      P''(t) ← crossover P'(t)
      P'''(t) ← mutation P''(t)
      P(t) ← replacement (P(t-1),P'''(t))
      evaluation P(t)
    Final(2)
Final(1)
```

Evolutionary Computation: A New Way to Search for Solutions

The evolution cycle



Evolutionary Computation: A New Way to Search for Solutions

HOW TO CONSTRUCT A GA?

- Representation
- Initial population
- Fitness function (How to evaluate a GA?)
- Chromosomes selection for parents
- Design of crossover operator
- Design of mutation operator
- Chromosomes replacement
- Stop condition

PROBLEM DEPENDENT

ALGORITHM COMPONENTS

Evolutionary Computation: A New Way to Search for Solutions

Genetic Representation

Choice of representation benefits from <u>domain knowledge</u>:

- Encoding of relevant parameters
- Appropriate resolution of parameters
- Must match genetic

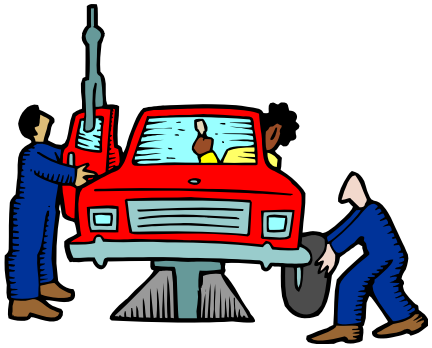
- operators of recombination and mutation
- Set of possible chromosomes should include optimal solution to the problem

Evolutionary Computation: A New Way to Search for Solutions

Discrete Representations

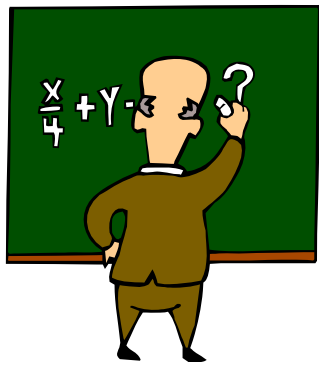
A sequence of *l* discrete values drawn from alphabet with cardinality *k*

- E.g., binary string of 8 positions (l=8, k=2): 01010100
- Can be mapped into several phenotypes:



to integer *i* using

binary code



01010100

84

0.328125

to real value *r* in range [min, max]:

$r = min + (i/255)(max-min)$

Job	A.M.	P.M.
1	x	
2		x
3	x	
4		x
5	x	
6		x
7	x	
8	x	

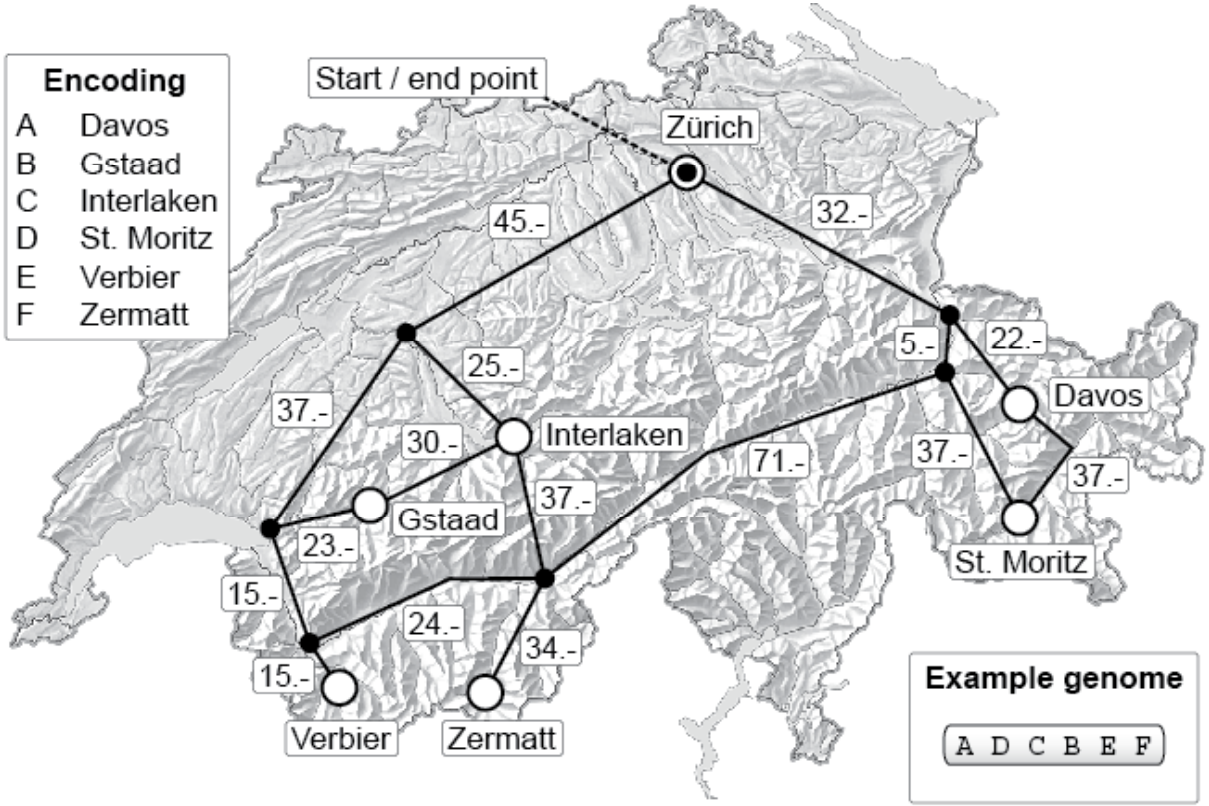
to job schedule:

- job=gene position
- time=gene value

Evolutionary Computation: A New Way to Search for Solutions

Sequence Representation

It is a particular case of discrete representation used for class of Traveling Salesman Problems (plan a path to visit n cities under some constraints). E.g., planning ski holidays with lowest transportation costs

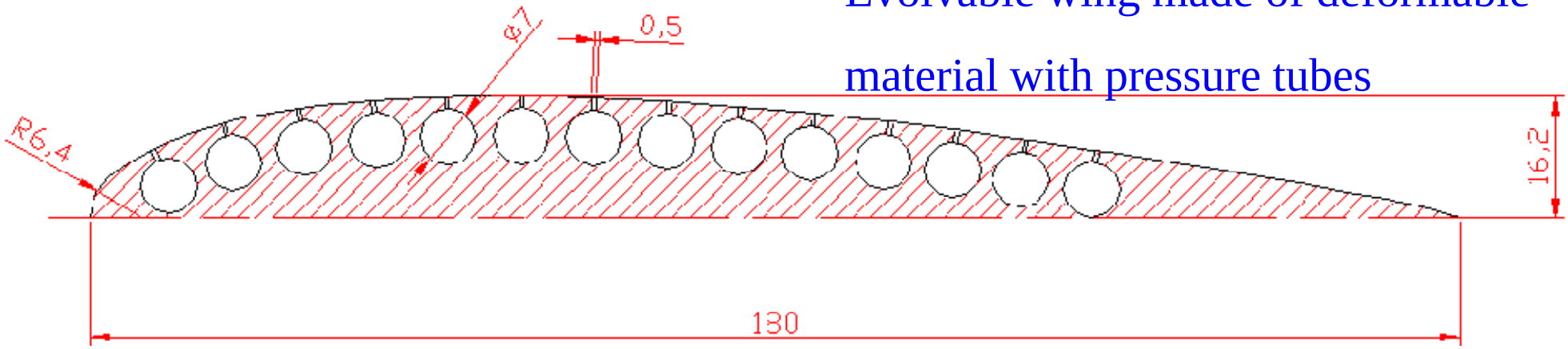


Evolutionary Computation: A New Way to Search for Solutions

Real-Valued Representation

Genotype is sequence of real values that represent parameters

- Used when high-precision parameter optimization is required
- For example, genetic encoding of wing profile for shape optimization



Genotype= pressure values of 14 tubes

Evolutionary Computation: A New Way to Search for Solutions

Tree-based Representation

Genotype describes a tree with branching points and terminals

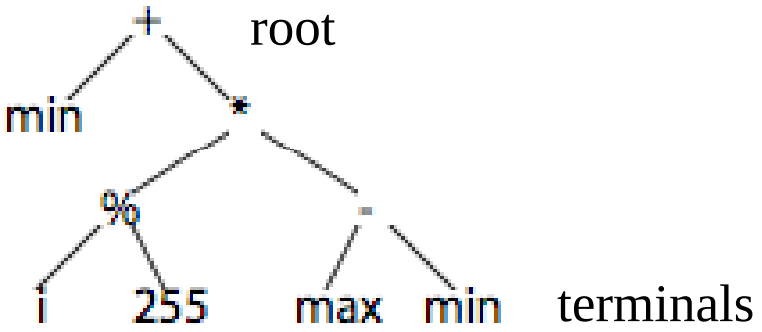
Suitable for encoding hierarchical structures

E.g., used to encode computer programs , made of:

- Operators (Function set: multiplication, If-Then, Log, etc.)
- Operands (Terminal set: constants, variables, sensor readings, etc.)

Expression r=min+(i/255)(max-min)

Nested list (+, min, (*, (/, i, 255), (-, max, min)))



Evolutionary Computation: A New Way to Search for Solutions

Initial Population

Sufficiently large to cover problem space (!), but sufficiently small for evaluation costs (typical size: between 10s and 1000s individuals)

Uniform sample of search space:

- **Binary strings: 0 or 1 with probability 0.5**
- **Real-valued representations: uniform on a given interval if bounded phenotype (e.g., +2.0, -2.0); otherwise best guess or binary string with dynamic mapping resolution (Schraudolph and Belew, 1992; Dürr et al, 2007)**
- **Trees are built recursively starting from root: root is randomly chosen from function set; for every branch, randomly choose among all elements of function set and of terminal set; if terminal is chosen, it becomes leaf; set maximum depth of tree.**

Evolutionary Computation: A New Way to Search for Solutions

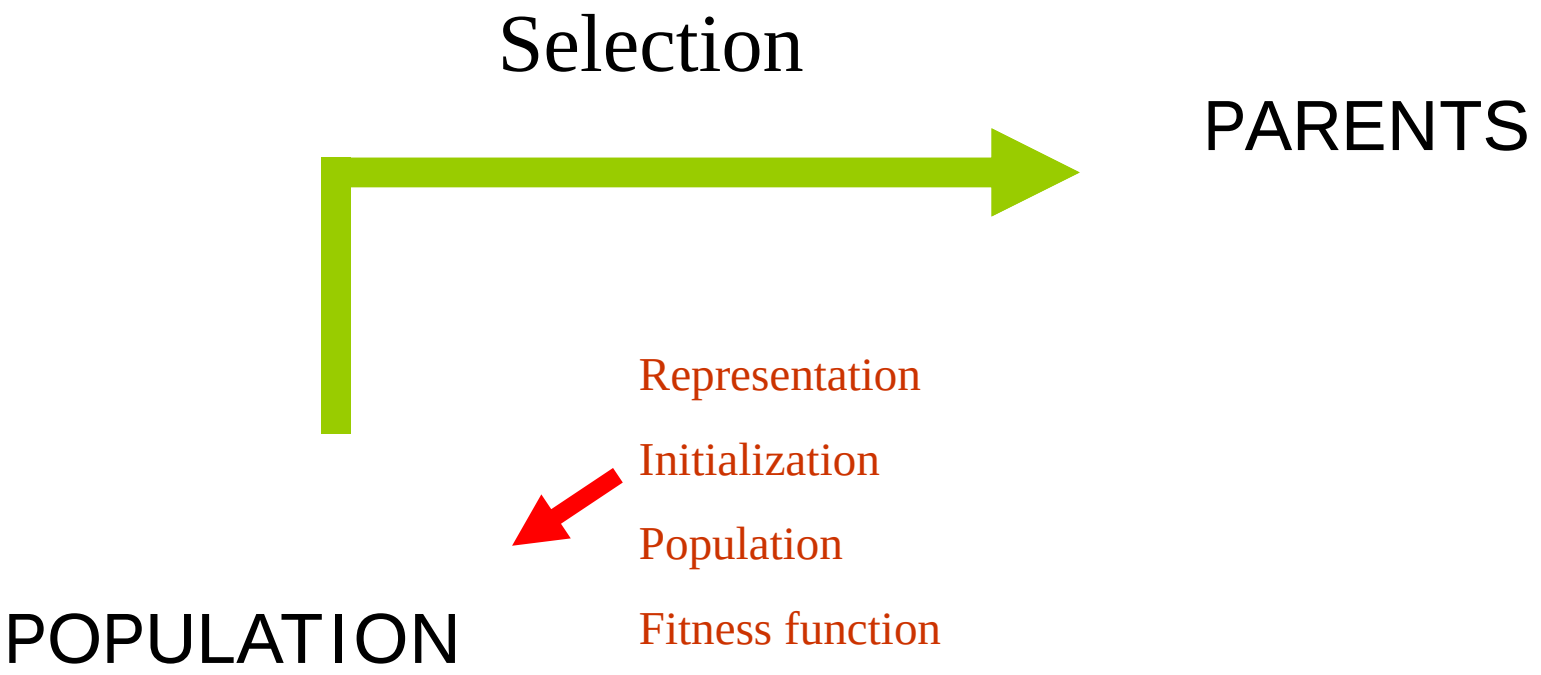
Fitness Function

Evaluates **performance** of phenotype with a numerical score

- Choice of components; e.g., lift and drag of wing
- Combination of components; e.g. (lift + 1/drag) or (lift - drag)
- Extensive test of each phenotype
- Warning! You Get What You Evaluate (example in application, later)
- Constraint problems can introduce a penalization in the fitness function.
- With multiple objectives we find a pareto (set of non-dominated solutions).

Evolutionary Computation: A New Way to Search for Solutions

HOW TO CONSTRUCT A GA?



Evolutionary Computation: A New Way to Search for Solutions

Selection

A method to make sure that better individuals make comparatively more offspring

Used in artificial evolution and breeding

- Selection pressure is inversely proportional to nr. of selected individuals
- High selection pressure = rapid loss of diversity and premature convergence
- Make sure that also less performing individuals can reproduce to some extent

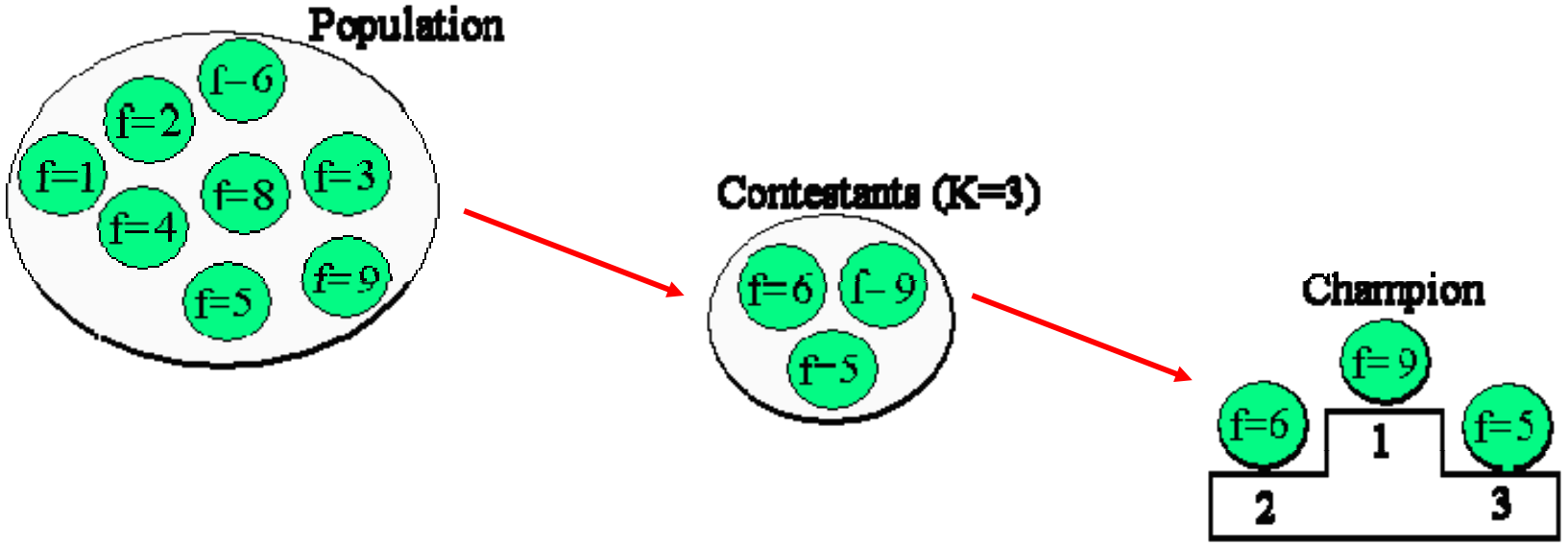
Evolutionary Computation: A New Way to Search for Solutions

Strategy of selection: Tournament selection

For each parent:

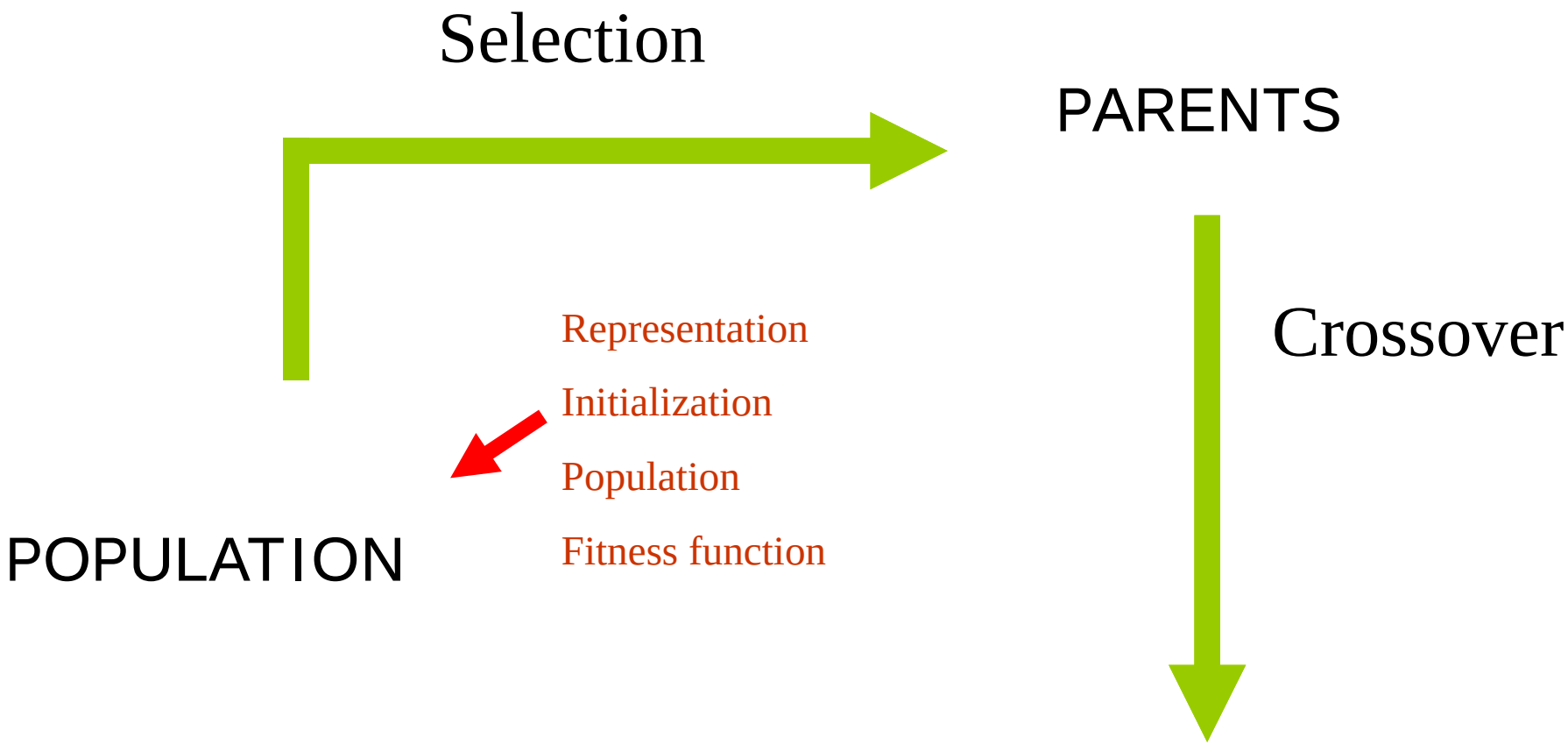
- **Random selection of *k* individuals, with replacement**
- **Selection of the best**

  k **is called the tournament size. A high** ***k*** **value, a high selective pressure and vice versa.**



Evolutionary Computation: A New Way to Search for Solutions

HOW TO CONSTRUCT A GA?



Evolutionary Computation: A New Way to Search for Solutions

Crossover operator

Emulates recombination of genetic material from two parents during meiosis

Exploitation of synergy of sub-solutions (building blocks) from parents

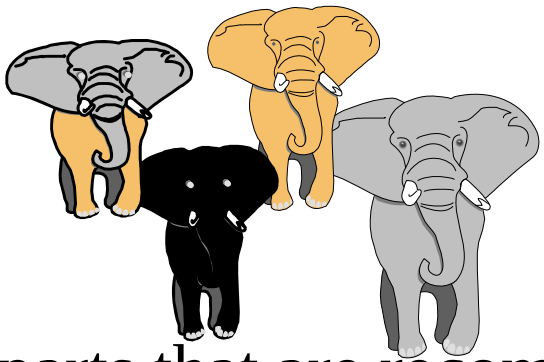
Applied to randomly paired offspring with probability p_c(pair)

- **The offspring must contain a heredity from the parents, associated to the parent features. In other case it would be a mutation operator.**
- **It depend on the representation.**
- **The recombination must produce valid chromosomes.**
- **It uses a probability for running on the two parents (Pc between 0.6 and 0.9, usually).**

Evolutionary Computation: A New Way to Search for Solutions

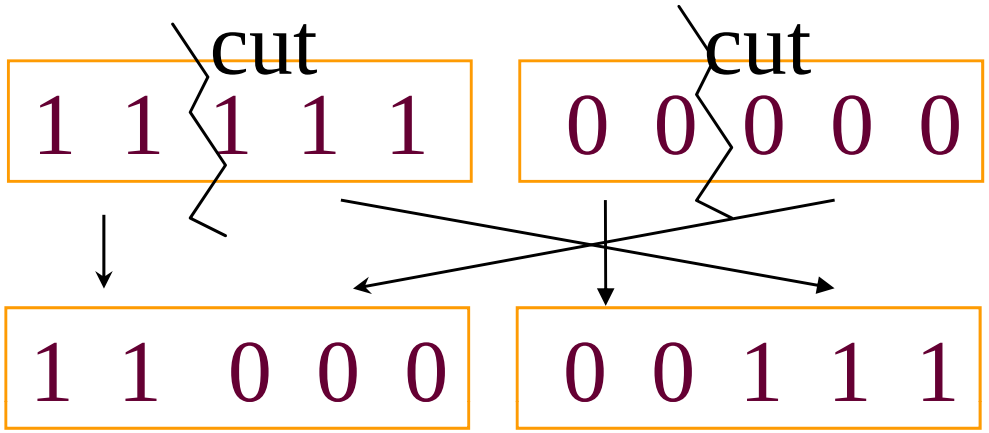
Example: Simple crossover on the binary representation

Population:

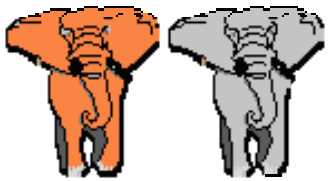


. . .

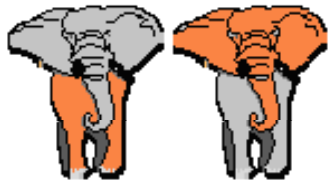
Each chromosome is divided into n parts that are recombined (example for n = 2)



parents



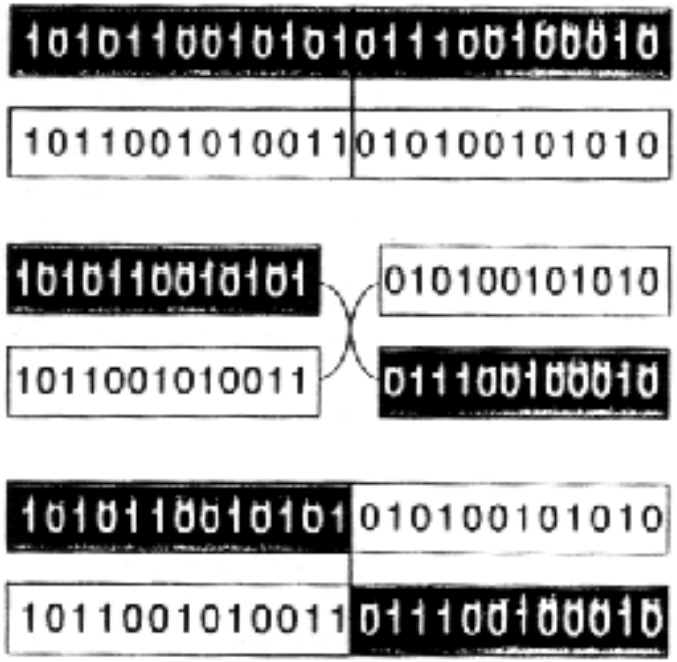
descendants



Evolutionary Computation: A New Way to Search for Solutions

Crossover

Classical image (John Holland): Biologica crosover



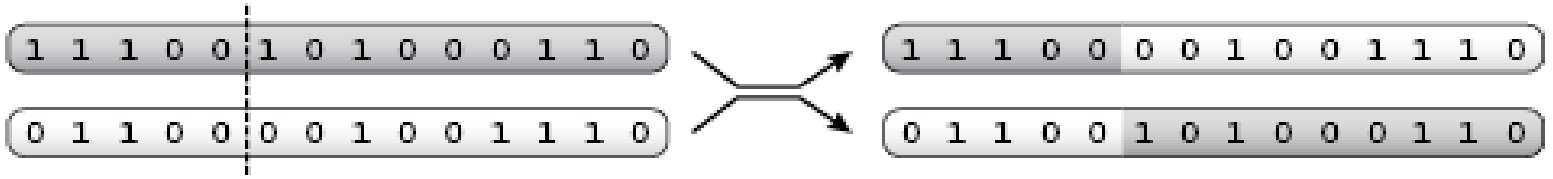
CROSSOVER is the fundamental mechanism of genetic rearrangement for both real organisms and genetic algorithms.

Chromosomes line up and then swap the portions of their genetic code beyond the crossover point.

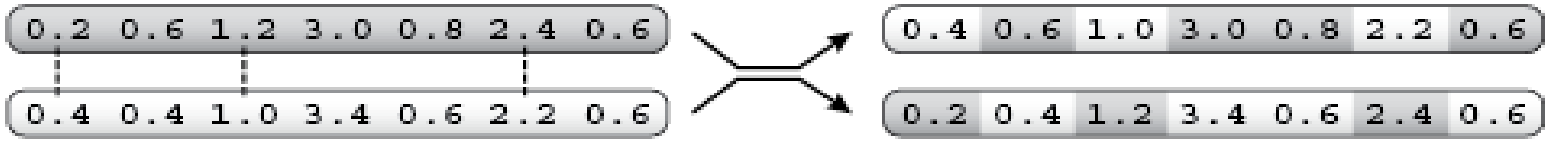
Evolutionary Computation: A New Way to Search for Solutions

Crossover

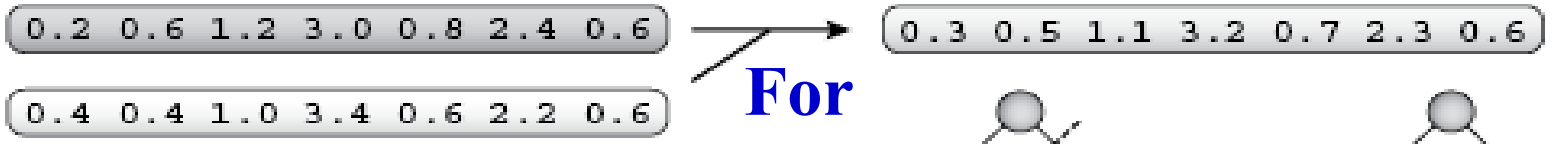
One point



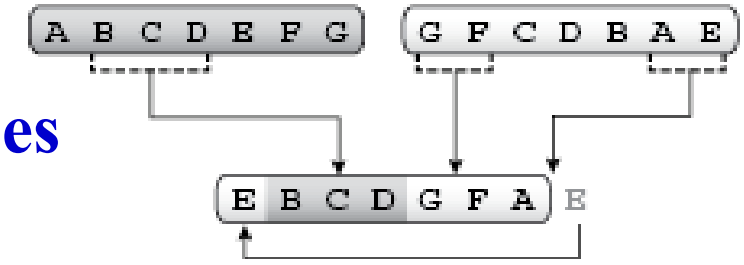
Uniform



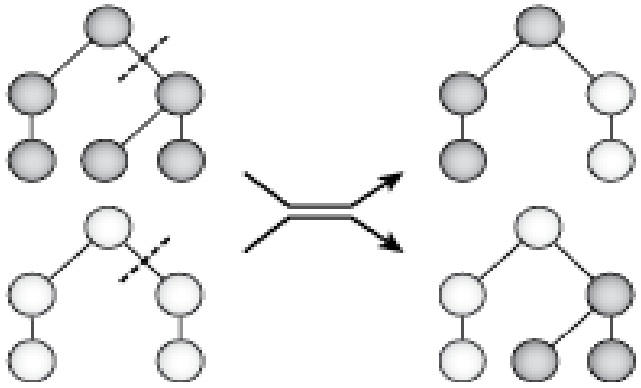
Arithmetic



For sequences

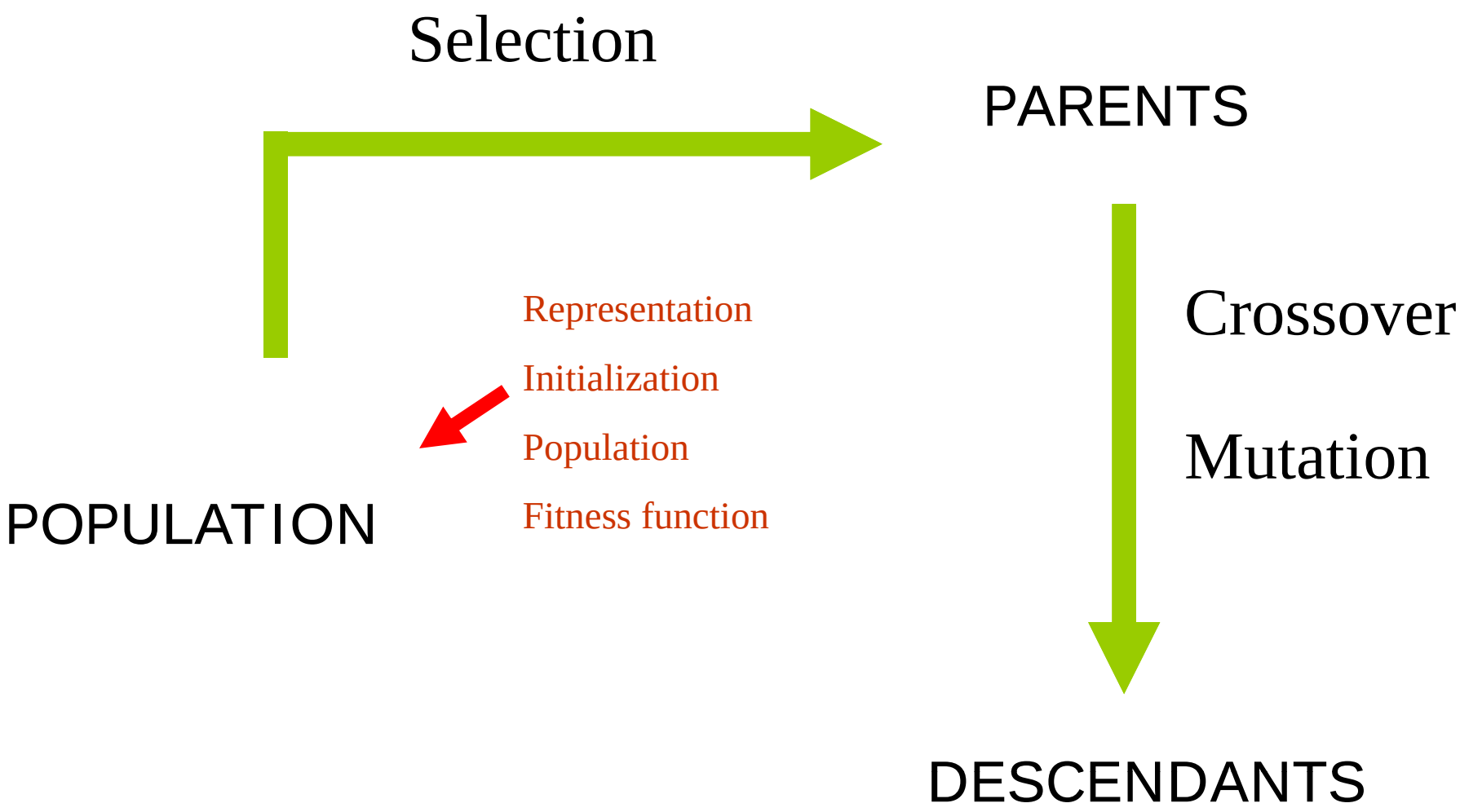


For trees



Evolutionary Computation: A New Way to Search for Solutions

HOW TO CONSTRUCT A GA?



Evolutionary Computation: A New Way to Search for Solutions

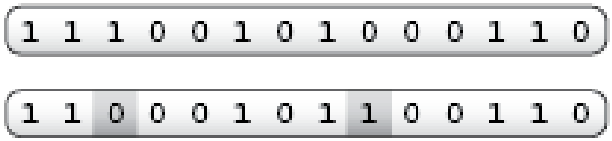
Mutation

Emulates genetic mutations

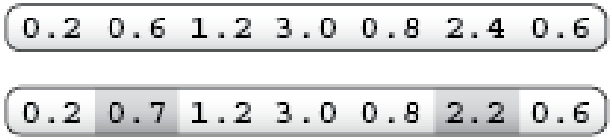
Exploration of variation of existing solutions

Applied to each character in the genotype with probability p_m(char)

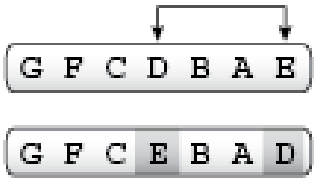
Binary genotypes



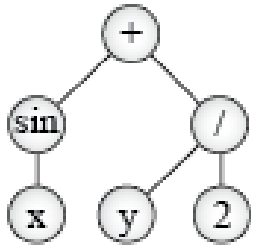
Real-valued genotypes

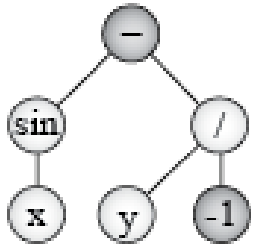


Sequence genotypes



For trees





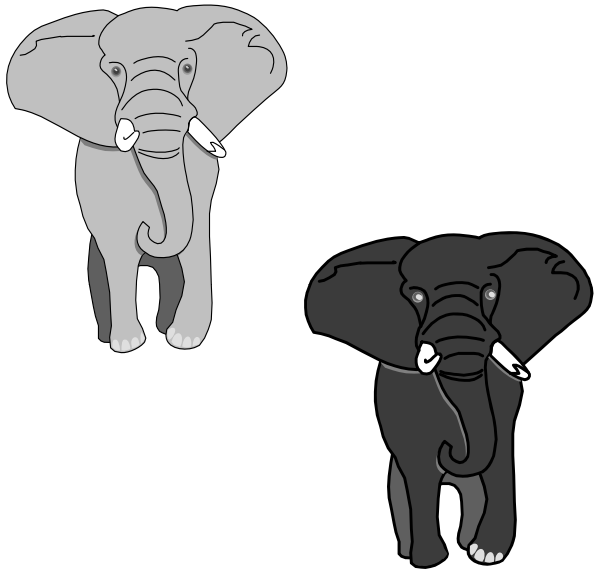
Evolutionary Computation: A New Way to Search for Solutions

Example: binary mutation

before 1 1 1 1 1

after 1 1 0 1 1

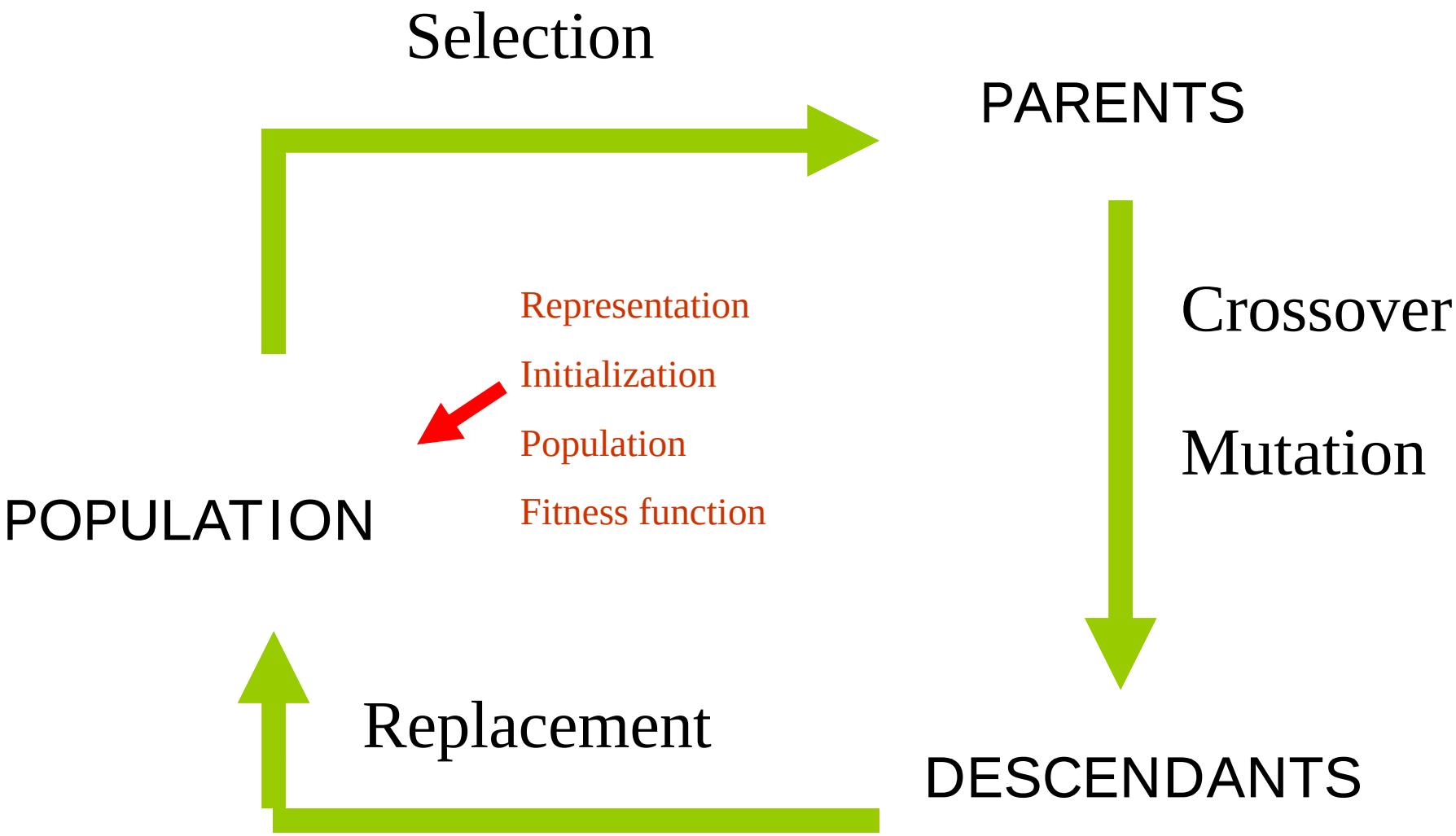
Mutated gen



The mutation happens with a low running probability per gen p_m

Evolutionary Computation: A New Way to Search for Solutions

HOW TO CONSTRUCT A GA?

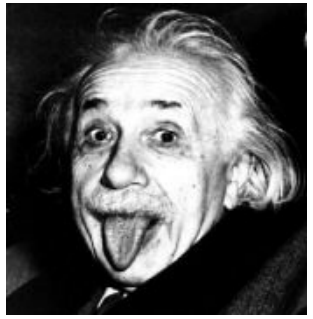


Evolutionary Computation: A New Way to Search for Solutions

Replacement

Generational replacement: old population is entirely replaced by offspring (most frequent method)

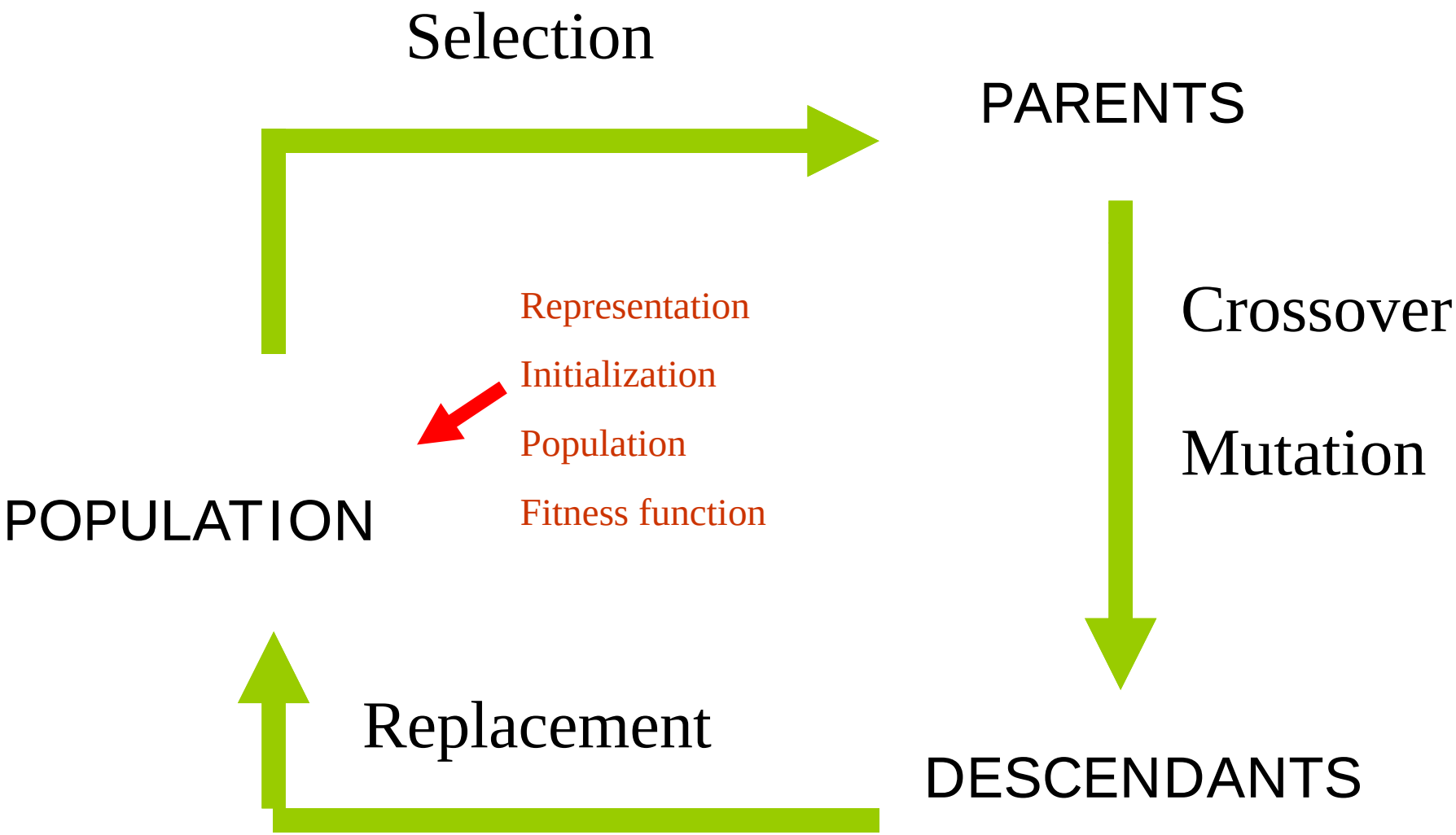
Elitism: maintain ***n*** best individuals from previous generation to prevent loss of best individuals by effects of mutations or sub-optimal fitness evaluation



Generational rollover: insert offspring at the place of worst individuals

Evolutionary Computation: A New Way to Search for Solutions

Components

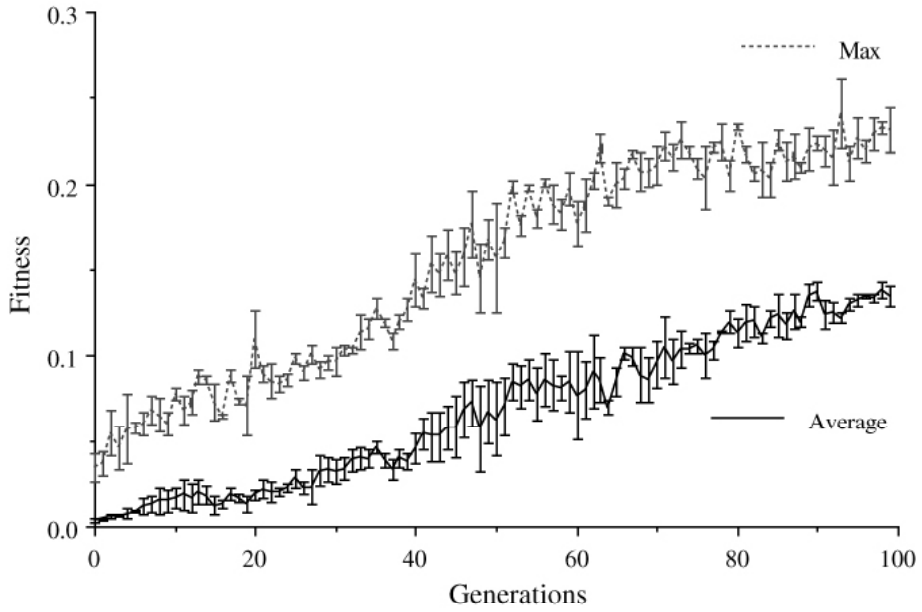


Evolutionary Computation: A New Way to Search for Solutions

Monitoring Performance

Track best and population average fitness of each generation

Multiple runs are necessary: plot average data and standard error



- Fitness graphs are meaningful only if the problem is stationary!
- Stagnation of fitness function may mean best solution found or premature convergence

Evolutionary Computation: A New Way to Search for Solutions

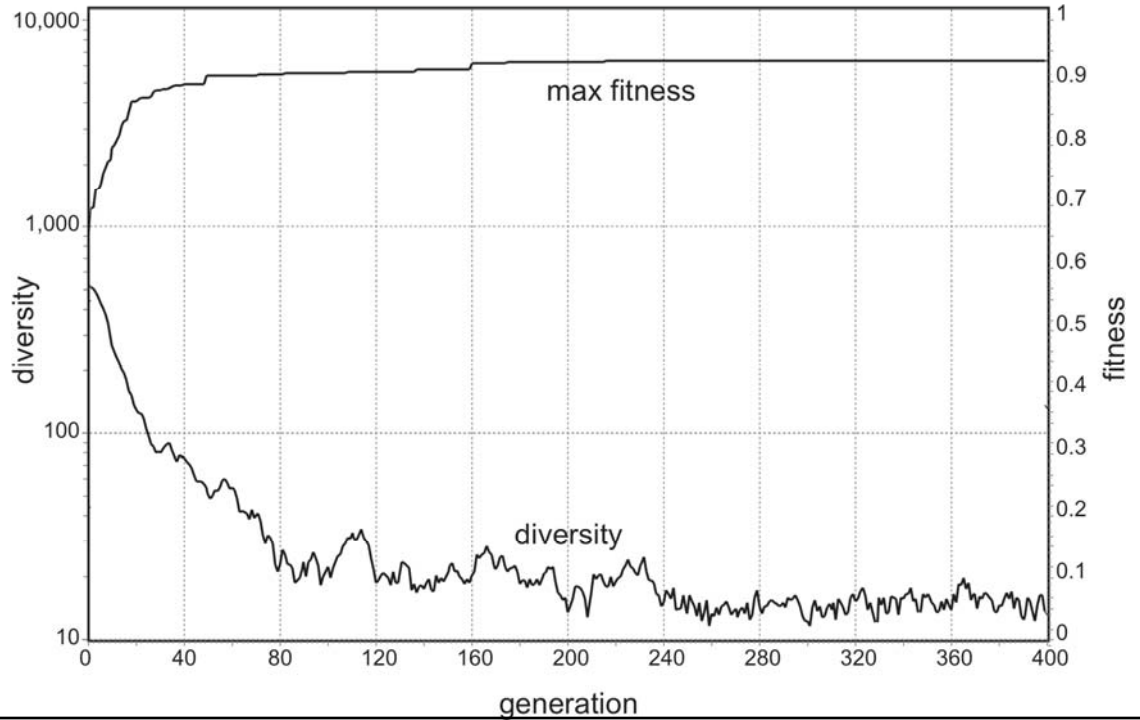
Measuring Diversity

Diversity tells whether the population has potential for further evolution.

Measures of diversity depend on genetic representation.

E.g., for binary and real valued, use sum of Euclidean or Hamming distances

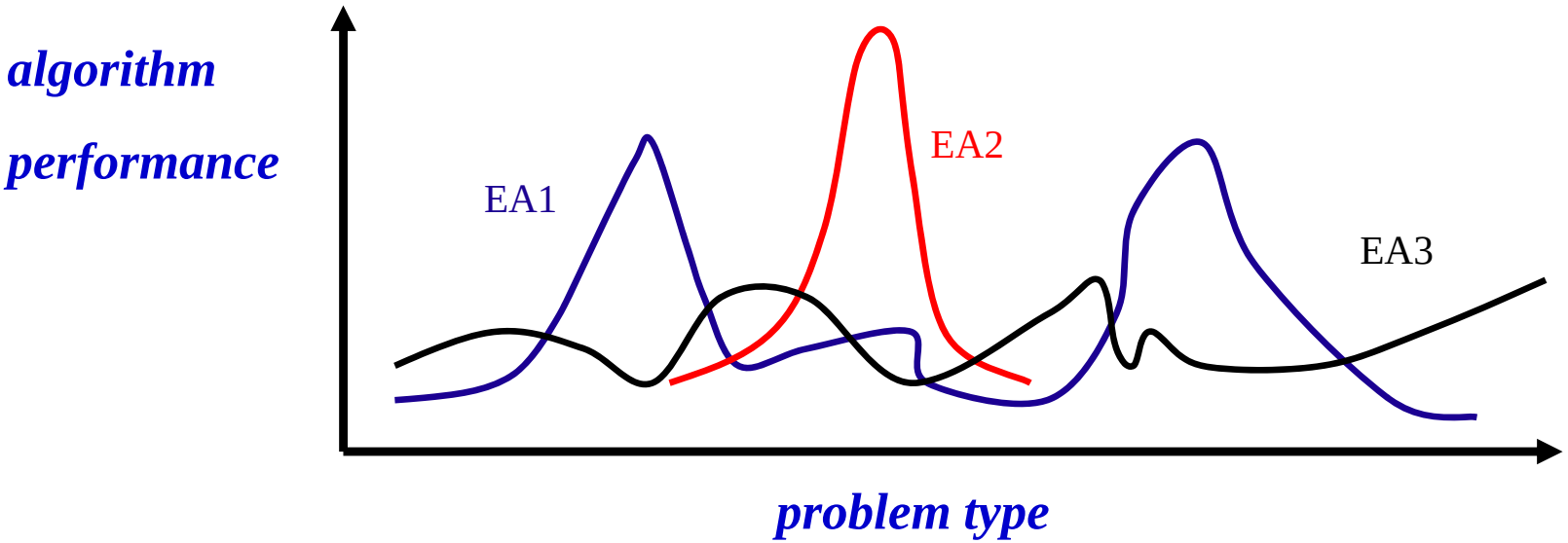
$$D_a(P) = \sum_{i,j \in P} d(g_i, g_j)$$



Evolutionary Computation: A New Way to Search for Solutions

Applicability

- **Evolutionary algorithms are used in a huge number of problems**
- **Biological inspiration is essential, but often distorted**
- **Different problems may require different algorithms**



Knowledge of problem domain can help to choose or assemble best algorithm

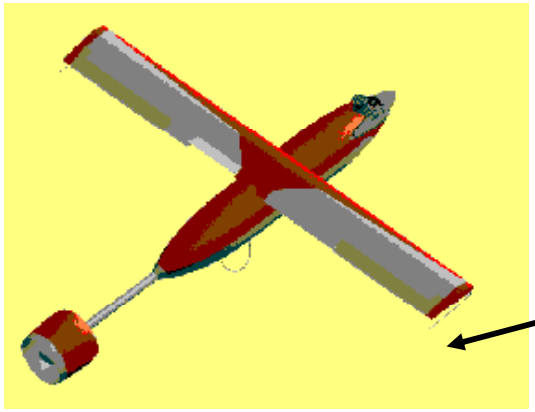
Evolutionary Computation: A New Way to Search for Solutions

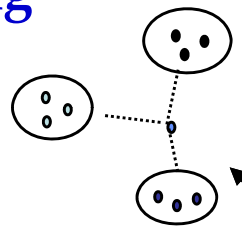
APPLICATIONS

Clasification

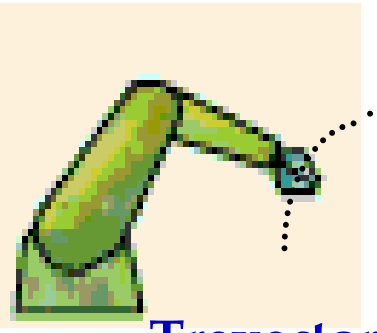
Learning

Structural optimization

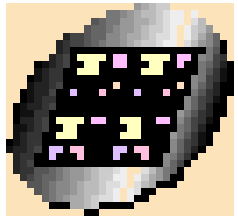




control

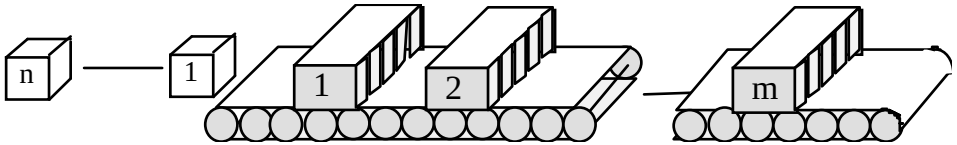


Trayectory generation



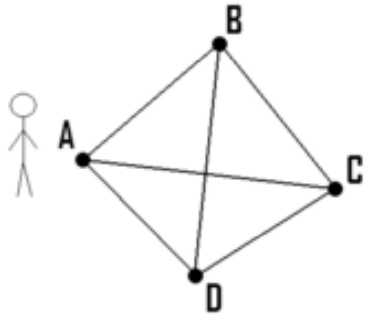
Circuits design VLSI

Planification

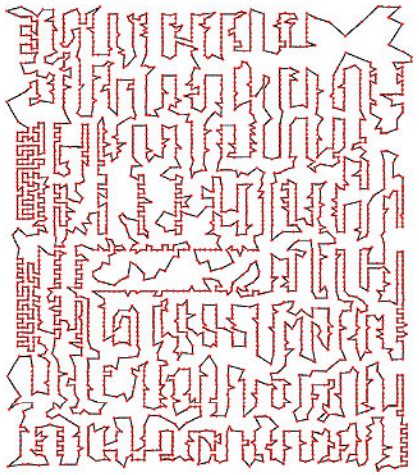


Evolutionary Computation: A New Way to Search for Solutions

Traveling Salesman Problem



- **Given a number of cities and the costs of traveling from any city to any other city, what is the cheapest round-trip route that visits each city exactly once and then returns to the starting city?**
  - **Trying all possible solutions means $n!$ permutations.**
  - **Using the techniques of dynamic programming, it can be solved in time $O(n^2 2^n)$**



- **The problem is of considerable practical importance. Example: printed circuit manufacturing: scheduling of a route of the drill machine to drill holes in a PCB.**

Evolutionary Computation: A New Way to Search for Solutions

EXAMPLE: TRAVELLING SALESMAN PROBLEM (TSP)

Order representation

(3 5 1 13 6 15 8 2 17 11 14 4 7 9 10 12 16)

17 cities

Objective: Sum of distance among cities.

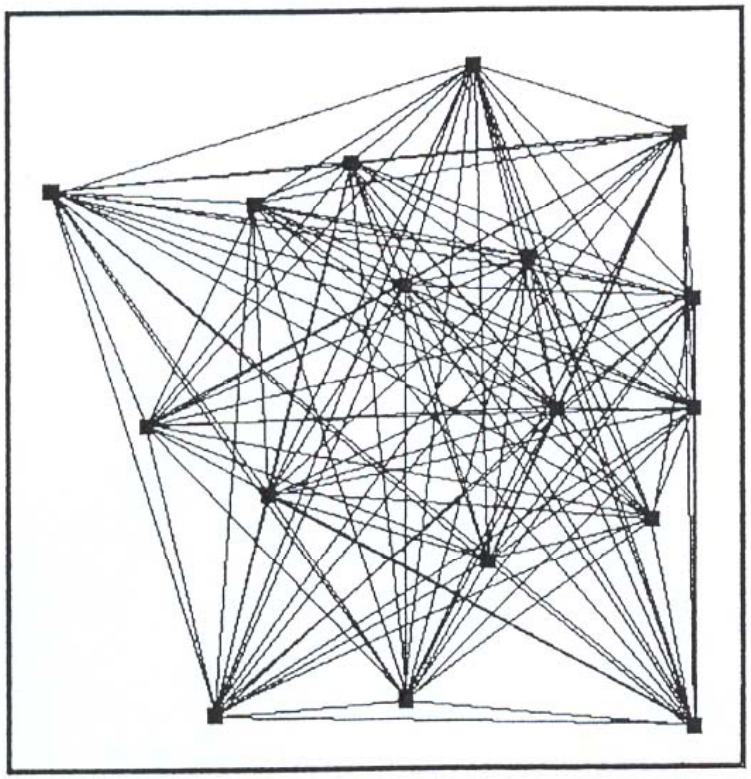
Population: 61 chromosomes - Elitism

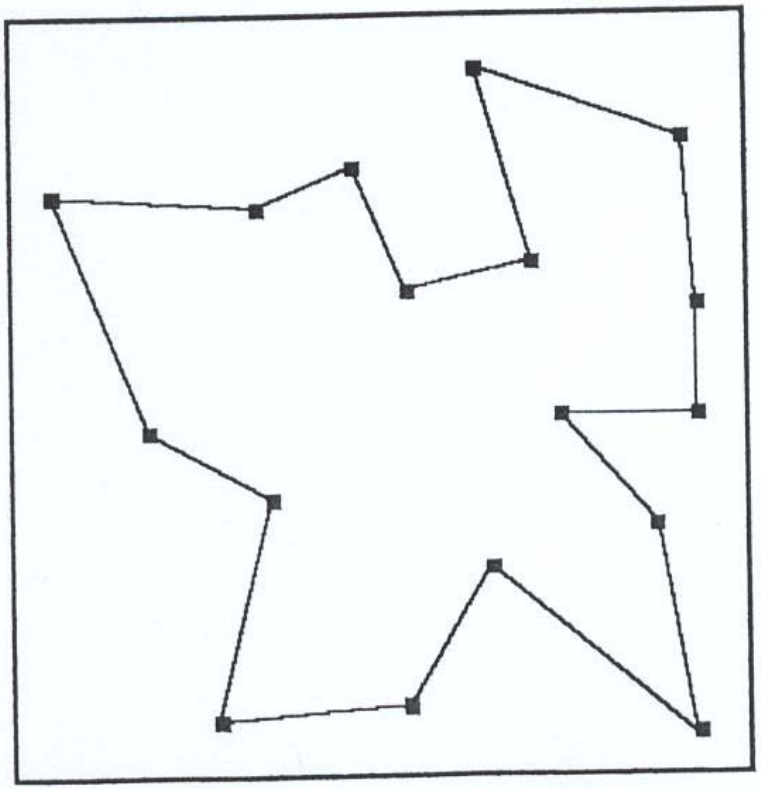
Crossover: OX ($P_c = 0,6$)

Mutation: List inversion ($P_m = 0,01$ – chromosome)

Evolutionary Computation: A New Way to Search for Solutions

TSP



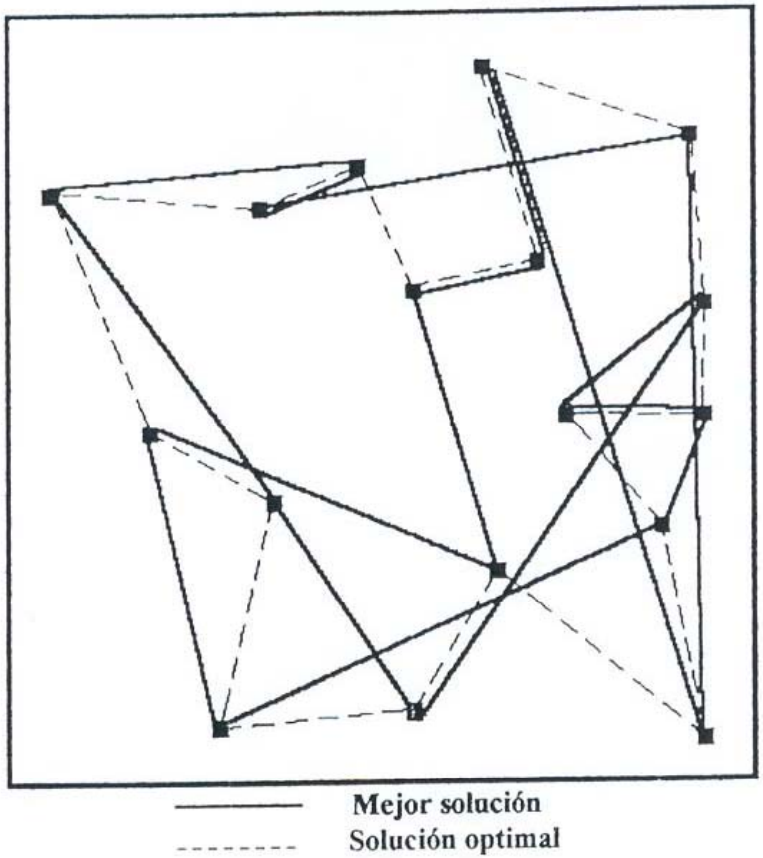


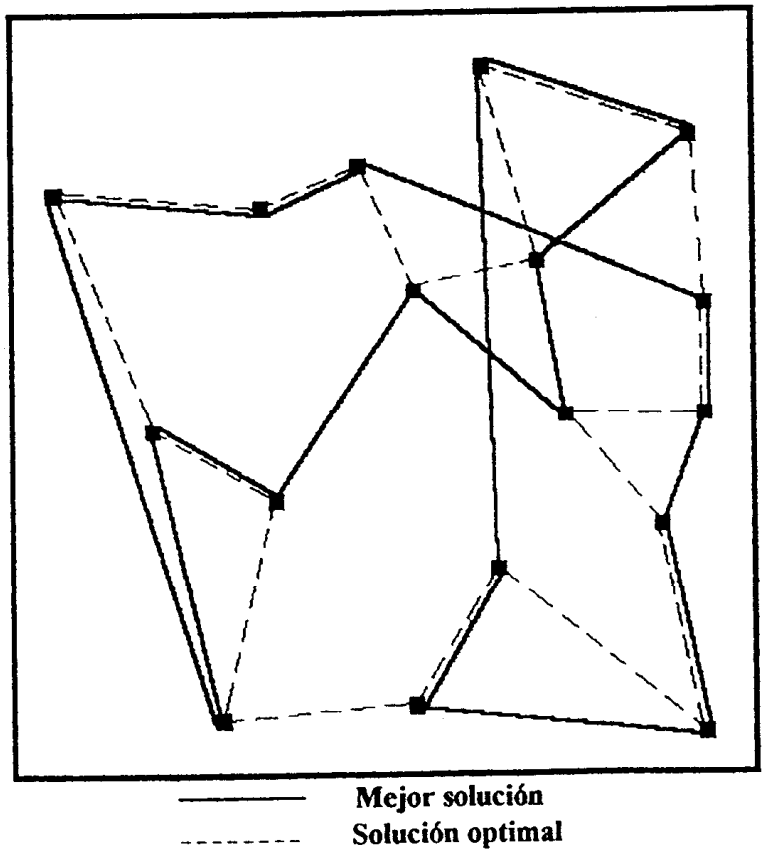
17! = 3.5568743 e14 possible solutions

Optimum solution: 226.64

Evolutionary Computation: A New Way to Search for Solutions

TSP





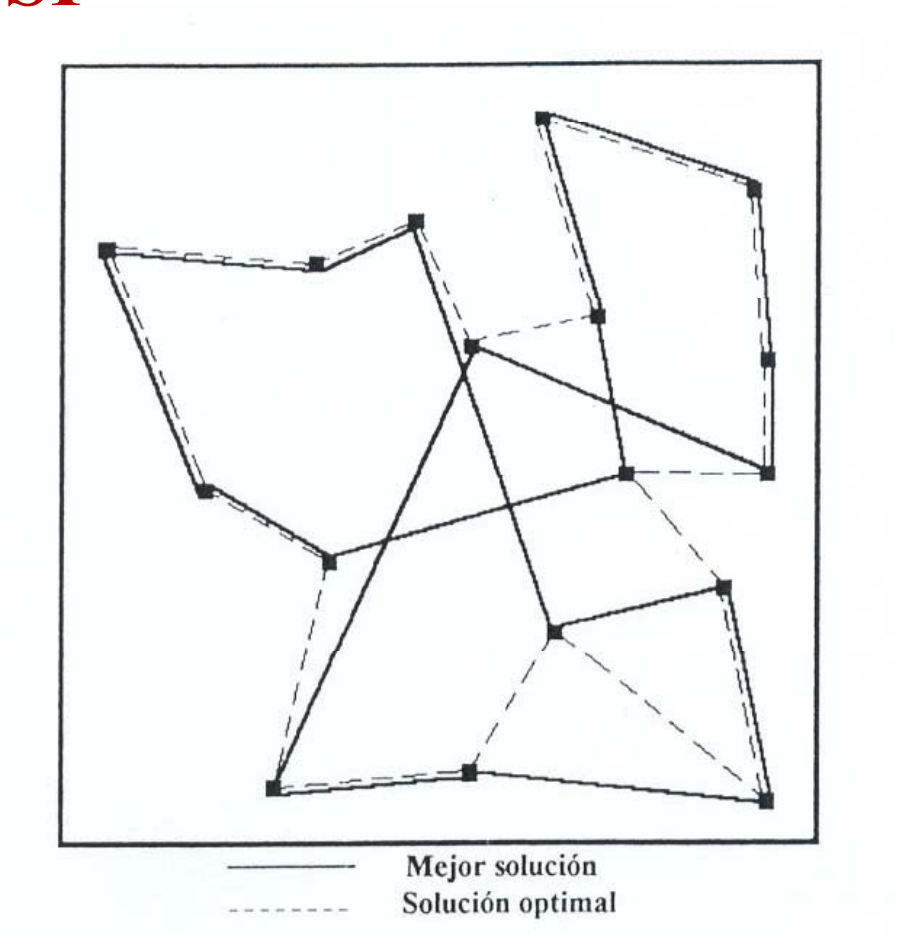
Iteration: 0 Cost: 403.7

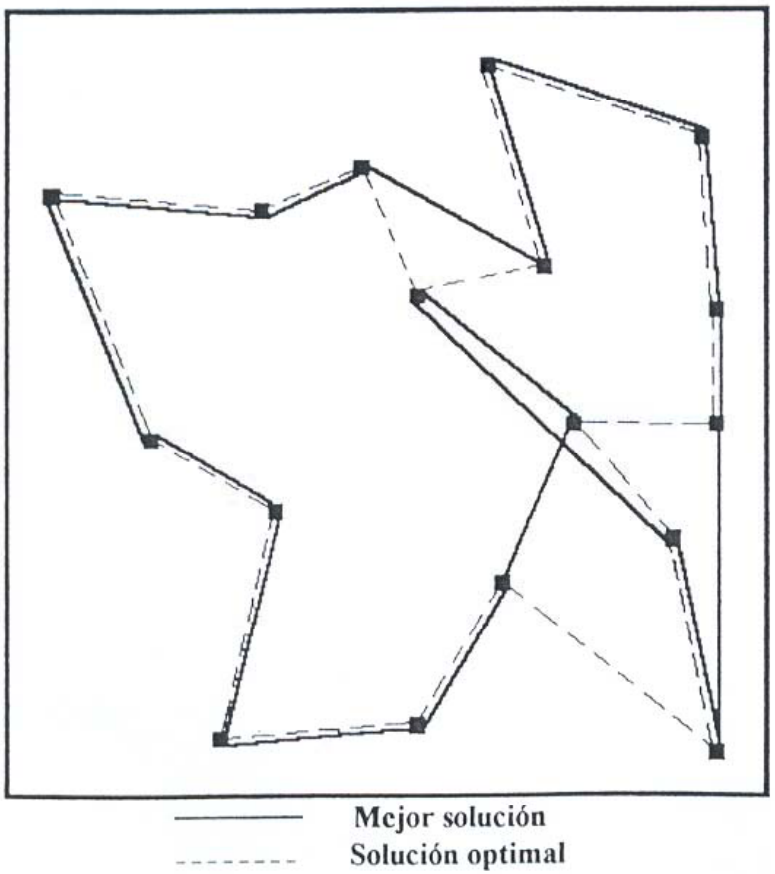
Iteration: 25 Cost: 303.86

Optimum solution: 226.64

Evolutionary Computation: A New Way to Search for Solutions

TSP





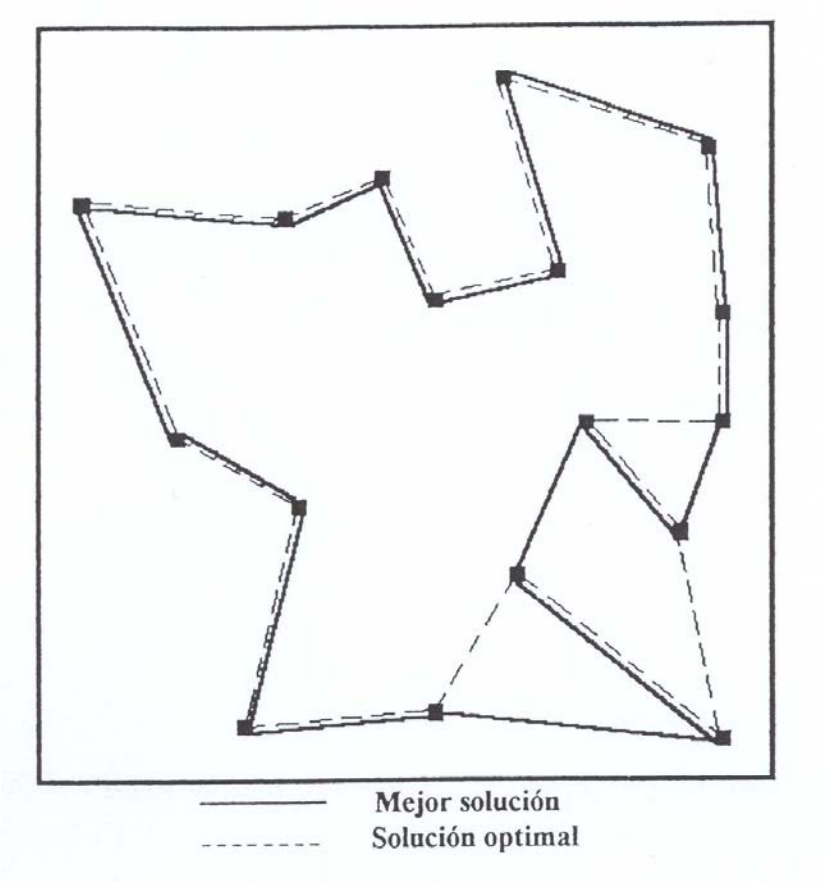
Iteration: 50 Cost: 293,6

Iteration: 100 Cost: 256,55

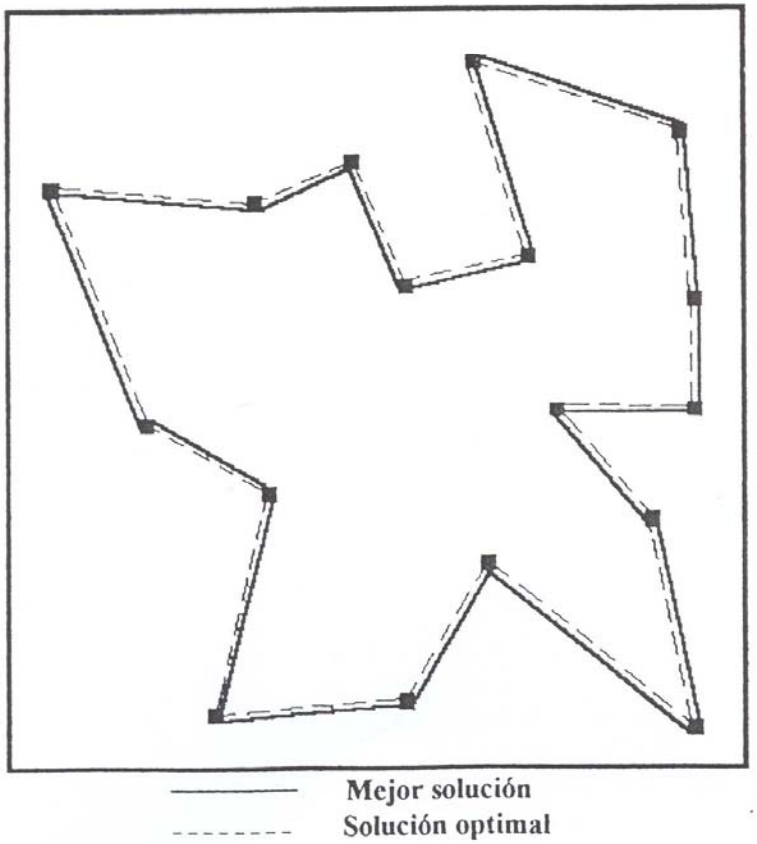
Optimum solution: 226,64

Evolutionary Computation: A New Way to Search for Solutions

TSP



Iteration: 200 Costo: 231,4

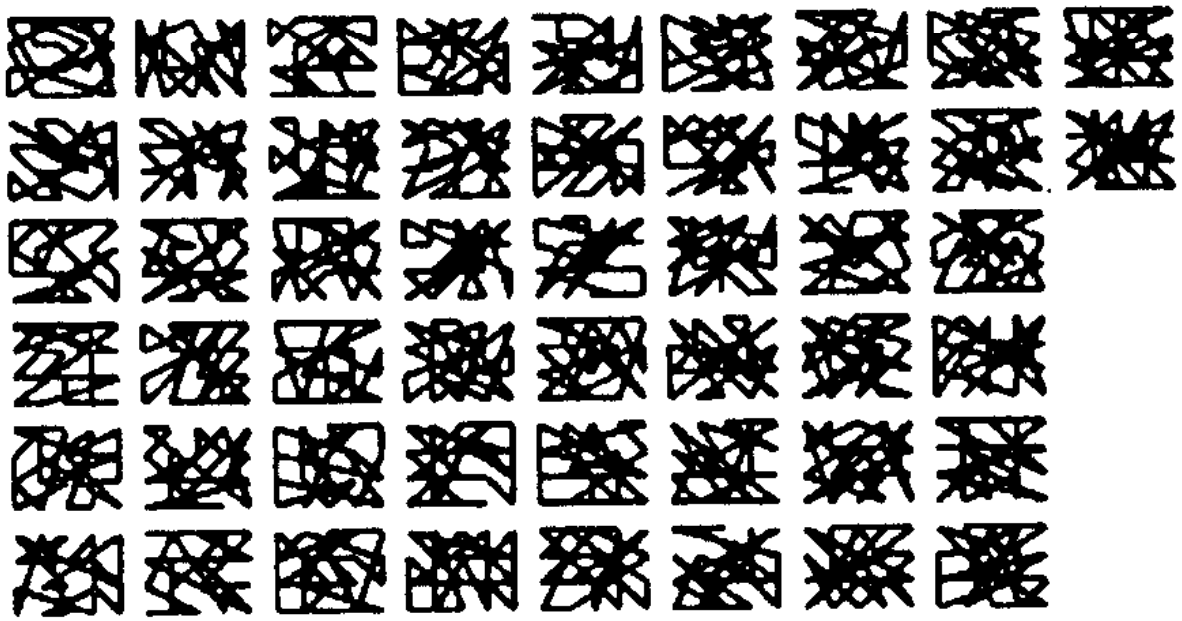


Iteration: 250

Optimum solution: 226,64

Evolutionary Computation: A New Way to Search for Solutions

TSP

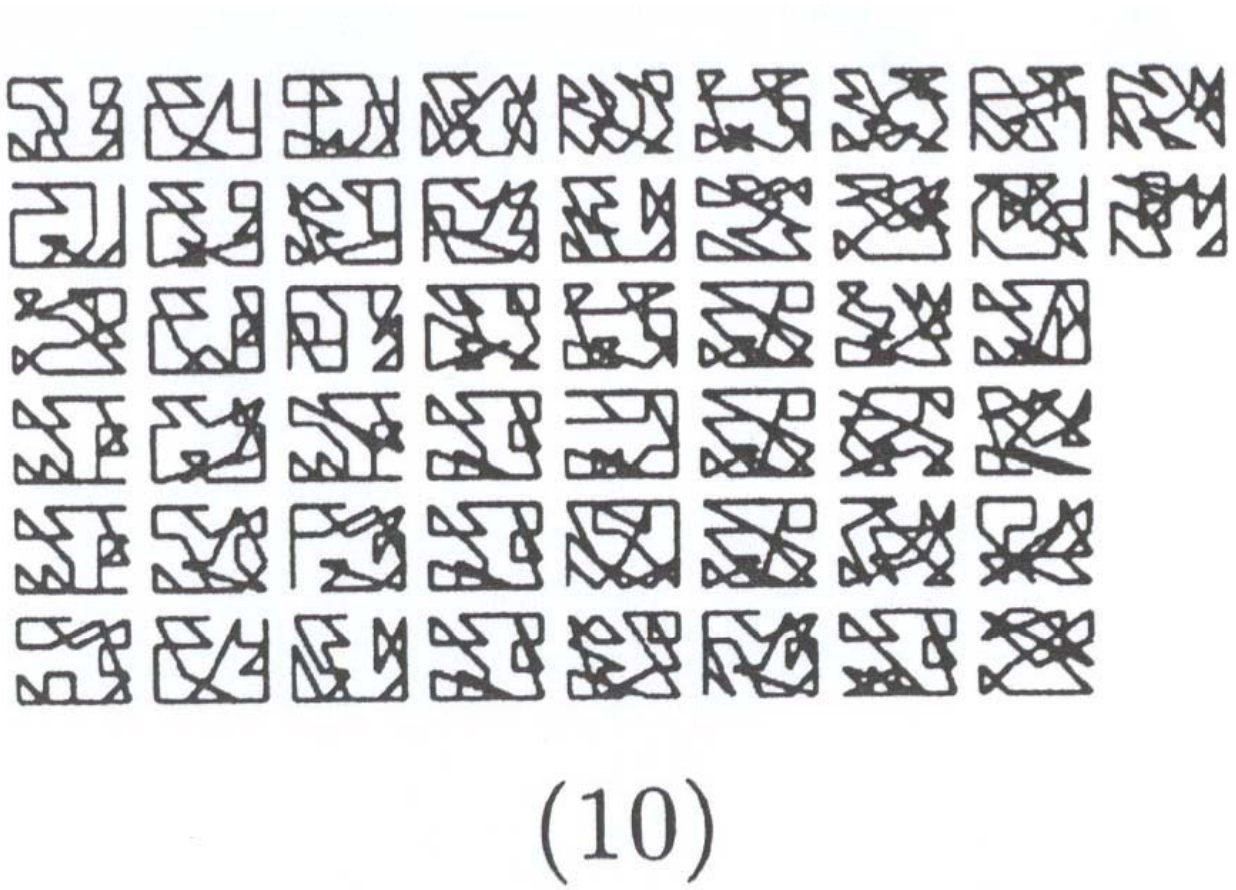


(0)

Visualization of the evolution with a population of size 50 and 70 iterations

Evolutionary Computation: A New Way to Search for Solutions

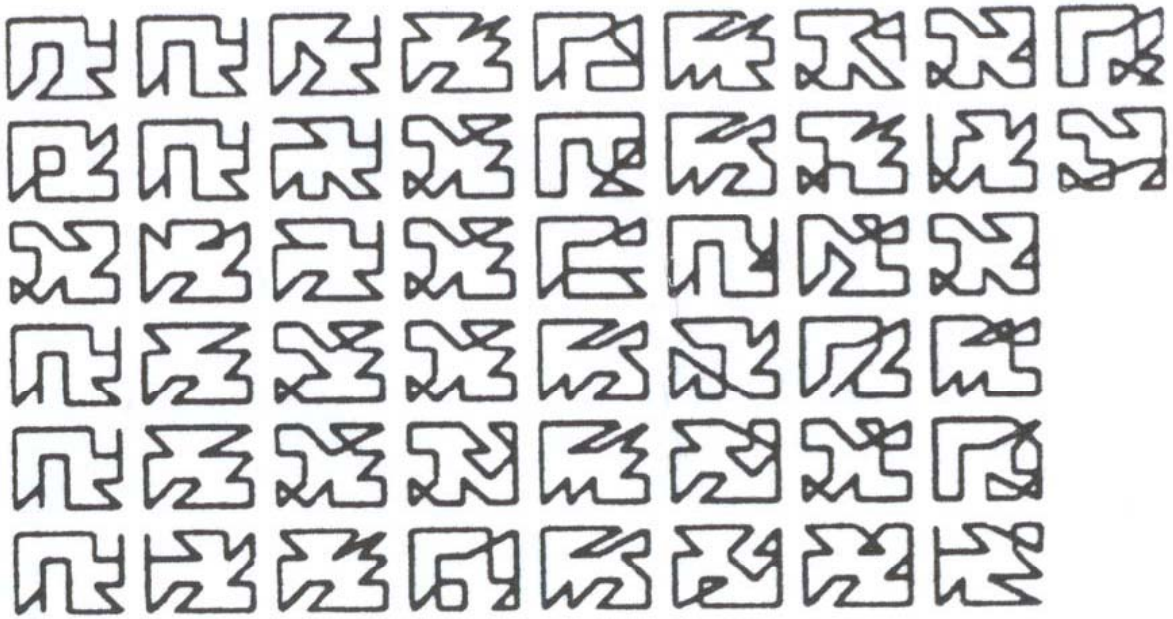
TSP



Visualization of the evolution with a population of size 50 and 70 iterations

Evolutionary Computation: A New Way to Search for Solutions

TSP

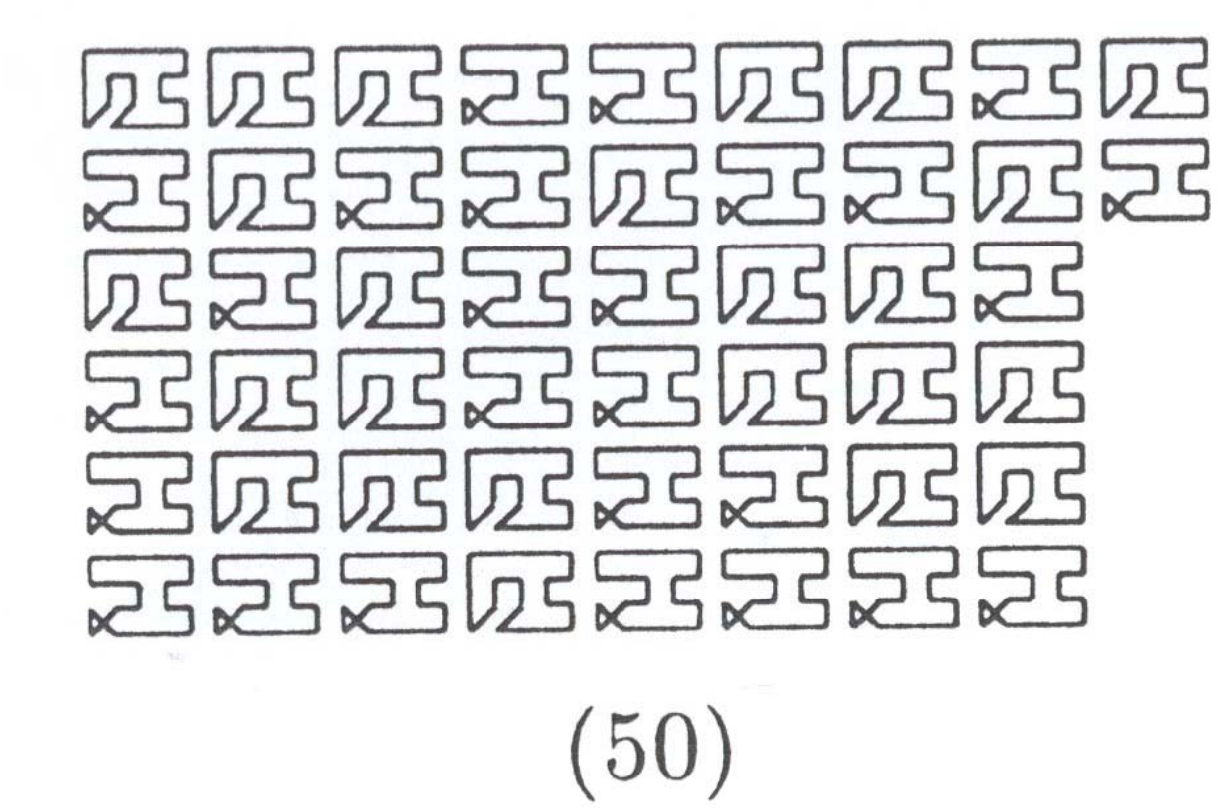


(30)

Visualization of the evolution with a population of size 50 and 70 iterations

Evolutionary Computation: A New Way to Search for Solutions

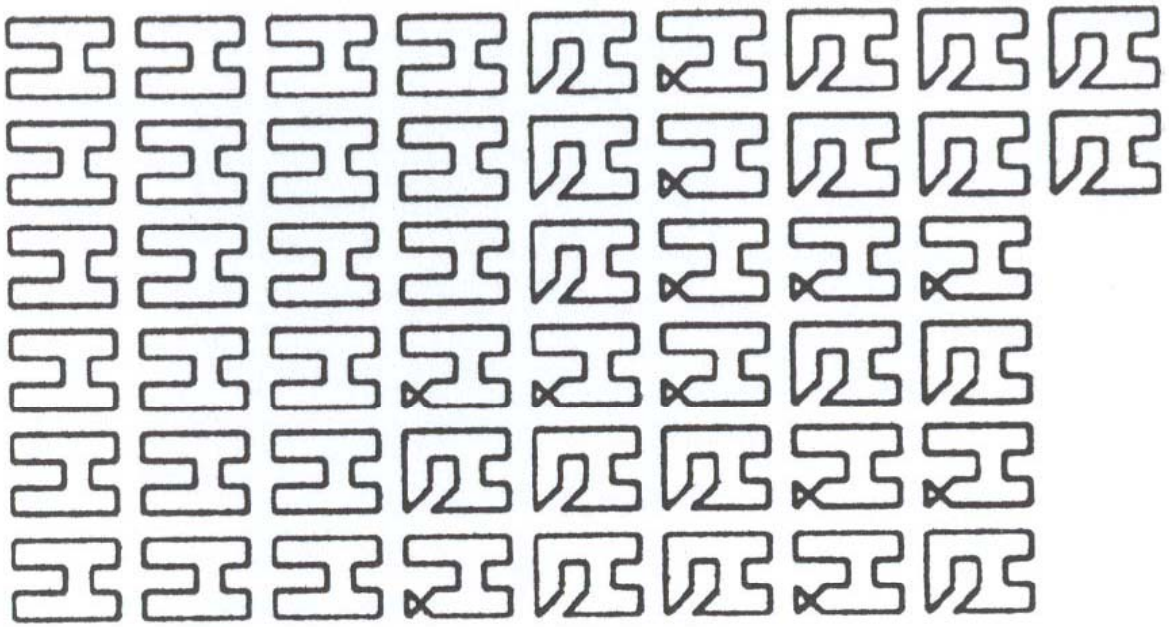
TSP



Visualization of the evolution with a population of size 50 and 70 iterations

Evolutionary Computation: A New Way to Search for Solutions

TSP

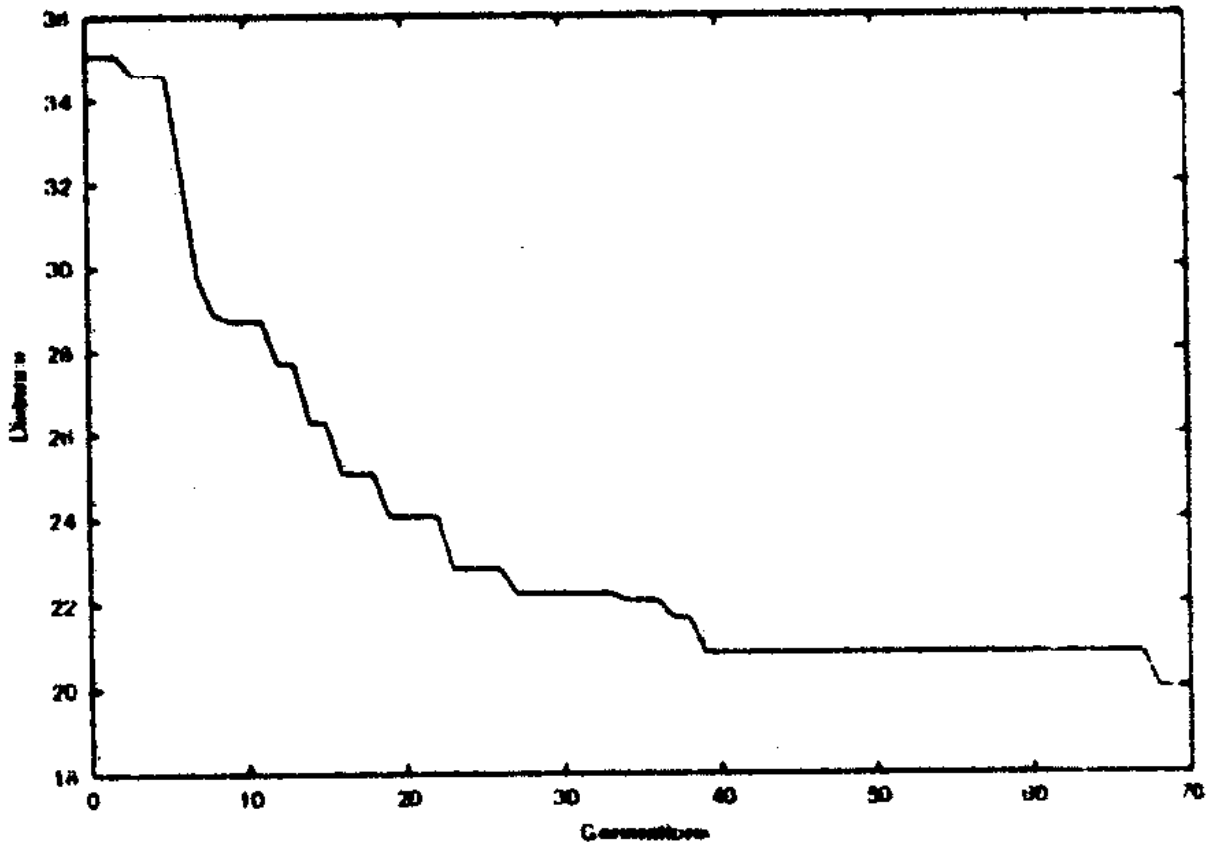


(70)

Visualization of the evolution with a population of size 50 and 70 iterations

Evolutionary Computation: A New Way to Search for Solutions

TSP



Visualization of the evolution with a population of size 50 and 70 iterations

Evolutionary Computation: A New Way to Search for Solutions

SOFTWARE AND IMPLEMENTATIONS

EO Evolutionary Computation Framework

EO is a template-based, ANSI-C++ compliant evolutionary computation library. It contains classes for almost any kind of evolutionary computation you might come up to at least for the ones we could think of. It is component-based, so that if you don't find the class you need in it, it is very easy to subclass existing abstract or concrete classes.

http://eodev.sourceforge.net/

Maintained by J.J. Merelo, Grupo Geneura, Univ. Granada
<jjmerelo@gmail.com>

Evolutionary Computation: A New Way to Search for Solutions

SOFTWARE AND IMPLEMENTATIONS

JCLEC JAVA Library

JCLEC is a software system for Evolutionary Computation (EC) research, developed in the Java programming language. It provides a high-level software environment to do any kind of Evolutionary Algorithm (EA), with support for genetic algorithms (binary, integer and real encoding), genetic programming (Koza style, strongly typed, and grammar based) and evolutionary programming.

http://jclec.sourceforge.net/

Maintained: Sebastián Ventura, Universad de Córdoba (sventura@uco.es)

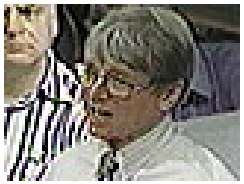
S. Ventura, C. Romero, A. Zafra, J.A. Delgado, C. Hervás-Martínez. JCLEC: A Java Framework for Evolutionary Computing. Soft Computing, 2008.

Evolutionary Computation: A New Way to Search for Solutions

Artificial Evolution

There are 4 classic paradigms:

Genetic Algorithms. 1975, Michigan University



John Holland

Inventor of genetic algorithms

Professor of CS and Psychology at the U. of Michigan.

Evolution Strategies 1964, Technische Universität Berlin

Inventors of Evolution Strategies

Hans-Paul Schwefel

Universität Dortmund

Ing. Ingo Rechenberg

Bionics & Evolutiontechnique

Technical University Berlin

http://www.bionik.tu-berlin.de/

Evolutionary Programming. 1960-1966, Florida

Lawrence J. Fogel,

Natural Selection, Inc.

Inventor of Evolutionary Programming

Genetic Programming. 1989, Stanford University

John Koza

Stanford University.

Inventor of Genetic Programming

Thre exist other modelos based on population evolution

Evolutionary Computation: A New Way to Search for Solutions

Artificial Evolution

EVOLUTIONARY COMPUTATION

EVOLUTIONARY CLASSIC PARADIGMS

- **GENETIC ALGORITHMS**
  - **GENETIC PROGRAMMING**
- **EVOLUTION STRATEGIES**
  - **EVOLUTIONARY PROGRAMMING**

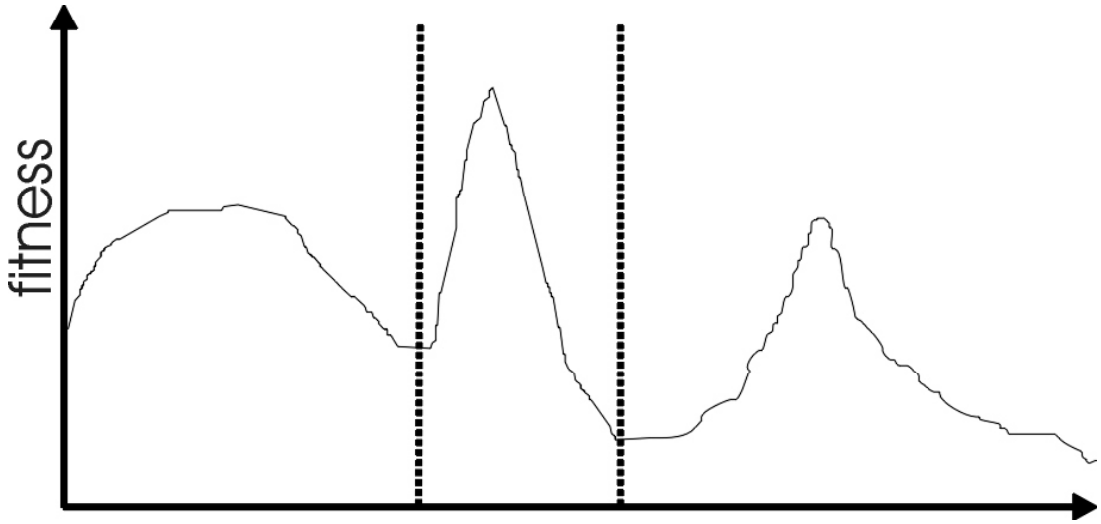
OTHER EVOLUTIONARY MODELS

- **ESTIMATION DISTRIBUTIION ALGORITHMS: PBIL, EDA, ...**
- **PARTICLE SWARM: SOCIAL ADAPTATION**
- **SCATTER SEARCH**
- **CULTURAL EVOLUTIOARY ALGORITHMS**
- **DIFFERENTIAL EVOLUTION**
- **MEMETIC ALGORITHMS**

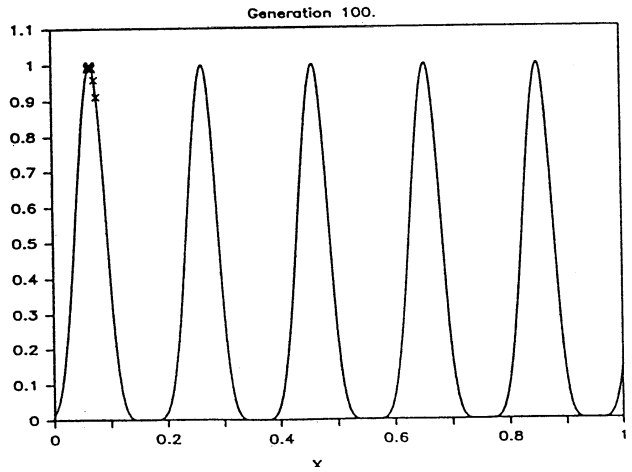
Evolutionary Computation: A New Way to Search for Solutions

Artificial Evolution – Advanced research – Niching Genetic Algorithms

- There are a lot of interesting problems with multiple optima.

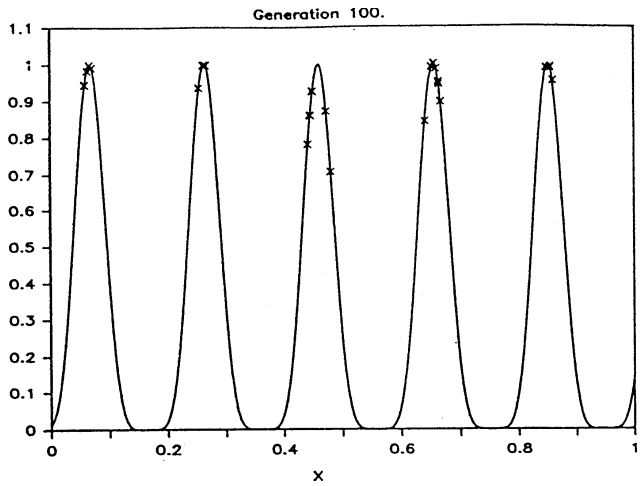


- In some problems we want to obtain a set of multiple solut



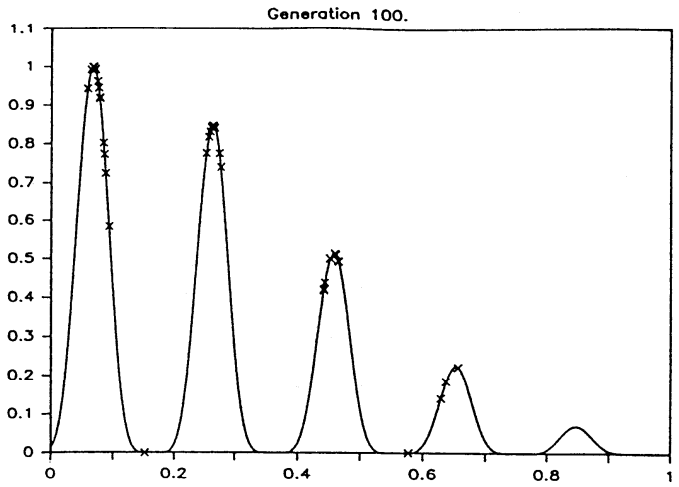
Evolutionary Computation: A New Way to Search for Solutions

Artificial Evolution – Advanced research – Niching Genetic Algorithms



Various global optima

Evolution with niches and without mutation



Various local optima

Evolution with niches and without mutation

We have a convergence towards different optima

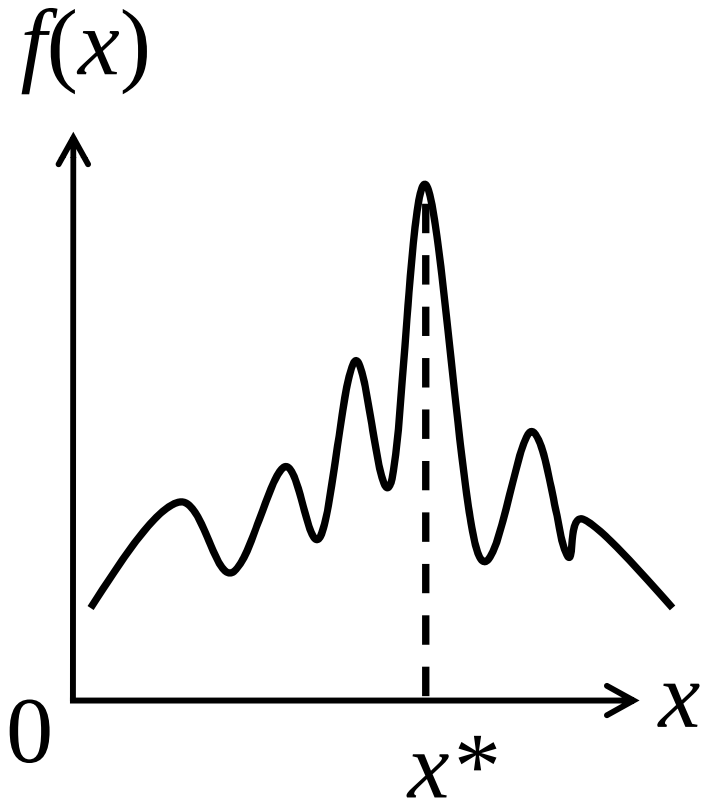
Pérez, E., Herrera, F. and Hernández, C. (2003). Finding multiple solutions in job shop scheduling by niching genetic algorithms. Journal of Intelligent Manufacturing, (14) Pp. 323-341.

Evolutionary Computation: A New Way to Search for Solutions

Artificial Evolution – Advanced research – Multiobjectivee Evolutionary Algorithms (MOEAs)

Single-objective optimization:

To find a single optimal solution x^* of a single objective function $f(x)$.



Multi-objective optimization:

To find a large number of Pareto optimal solutions with respect to multiple objective functions.

Evolutionary Computation: A New Way to Search for Solutions

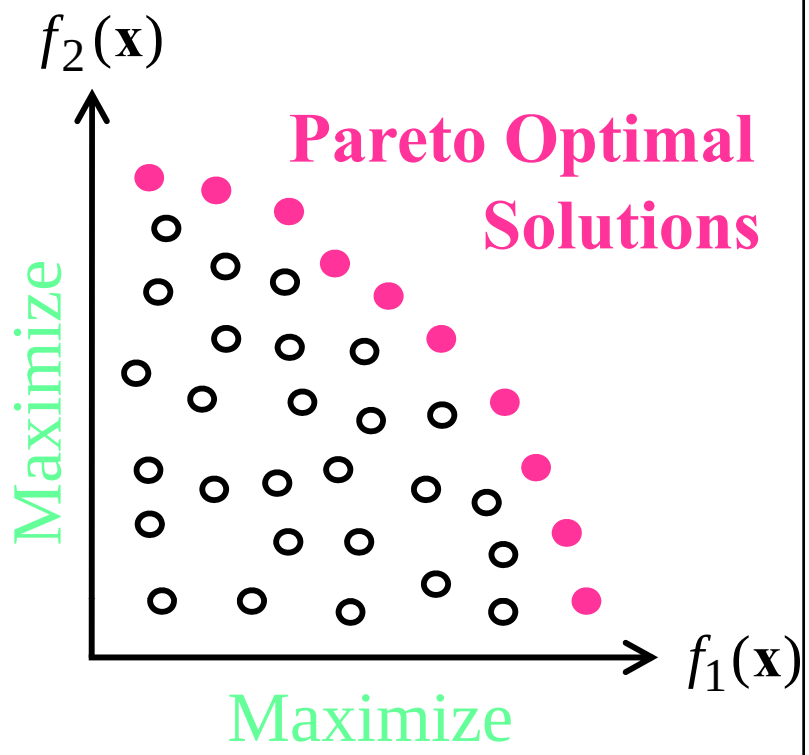
Artificial Evolution – Advanced research – Multiobjectivee Evolutionary Algorithms (MOEAs)

Multiobjective Optimization Problem

Maximize $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}),\ f_2(\mathbf{x}),\ ...,\ f_k(\mathbf{x}))$

subject to $\mathbf{x} \in \mathbf{X}$

Many Pareto-optimal solutions

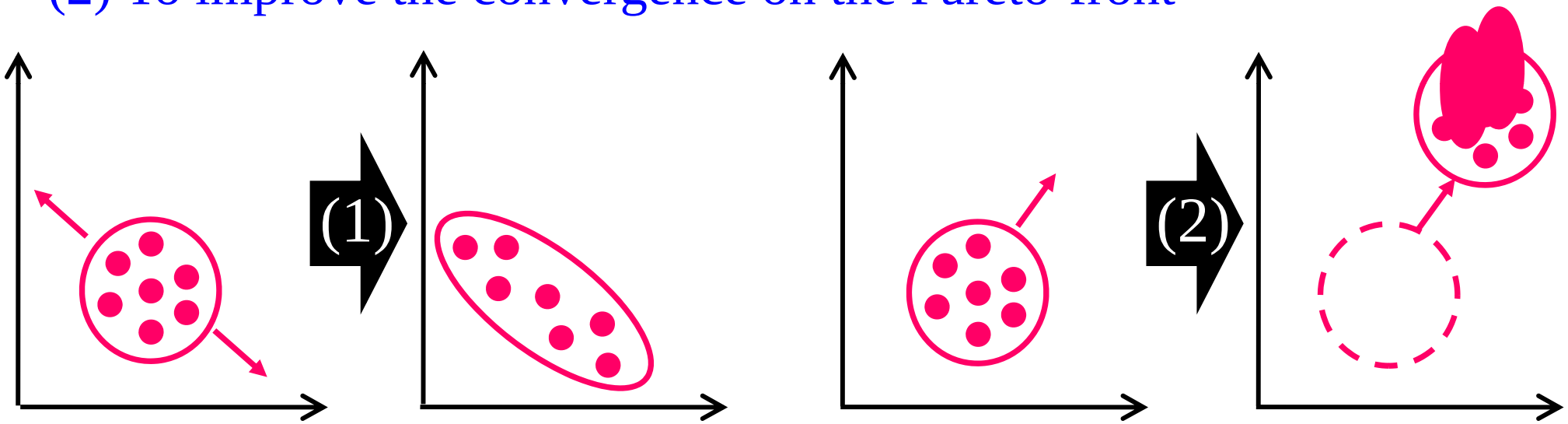


Evolutionary Computation: A New Way to Search for Solutions

Two Goals in the Design of MOEAs

(1) To increase the diversity of solutions

(2) To improve the convergence on the Pareto-front



Niching, Crowding

Pareto Ranking & Elitist Strategy

Nondominated Sorting Genetic Algorithm II

K. Deb, A. Pratap, S. Agarwal and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II.

IEEE Transactions on Evolutionary Computation 6:2 (2002) 182-197.

Evolutionary Computation: A New Way to Search for Solutions

Artificial Evolution – Advanced research – Memetic Algorithms

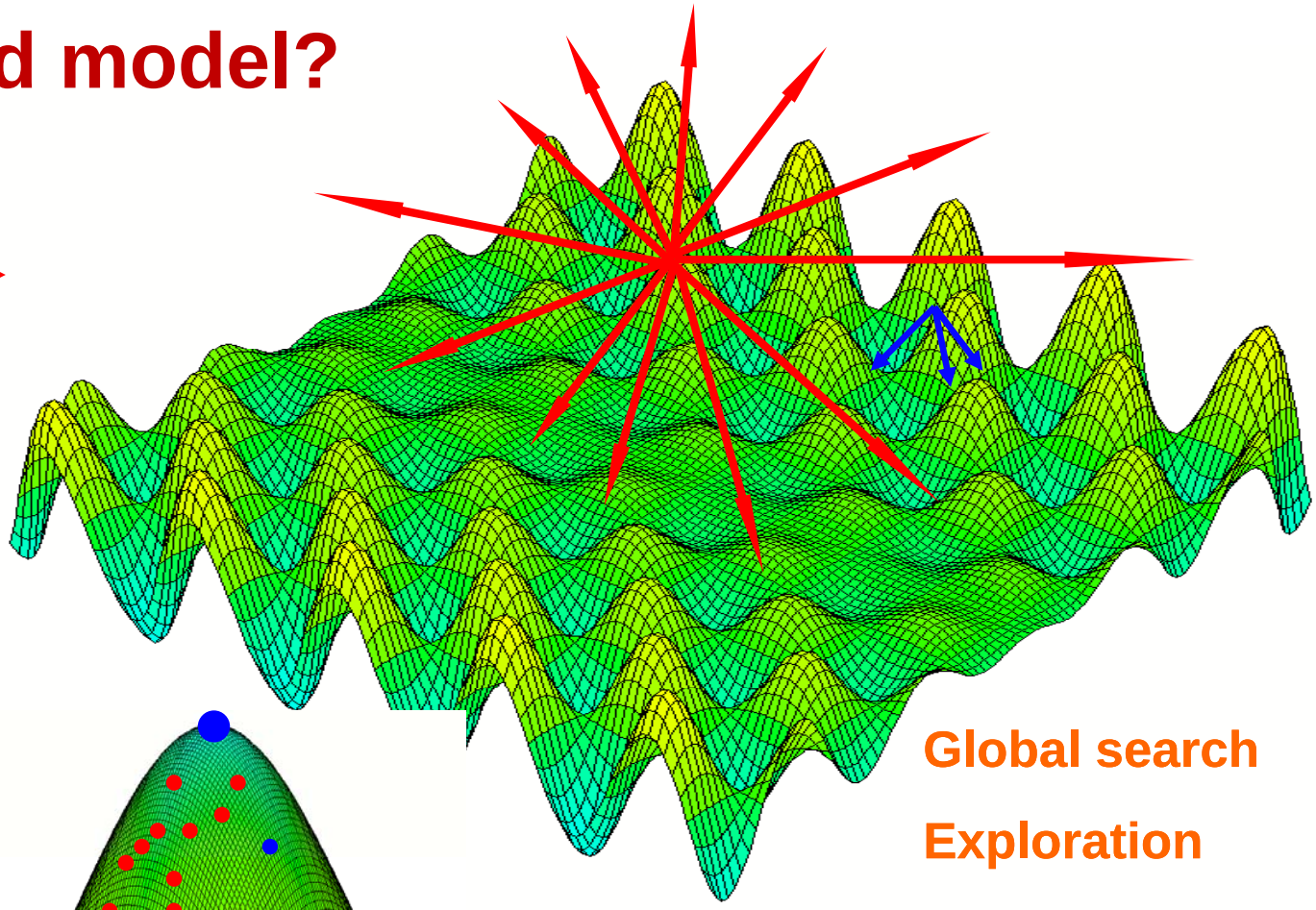
Algorithm based on the evolution of populations that use the knowledge on the problem in the search process (usually, the knowledge is in the form of local search algorithms acting on the population individuals).

Why this hybrid model?

Evolutionary Computation: A New Way to Search for Solutions

Why this hybrid model?

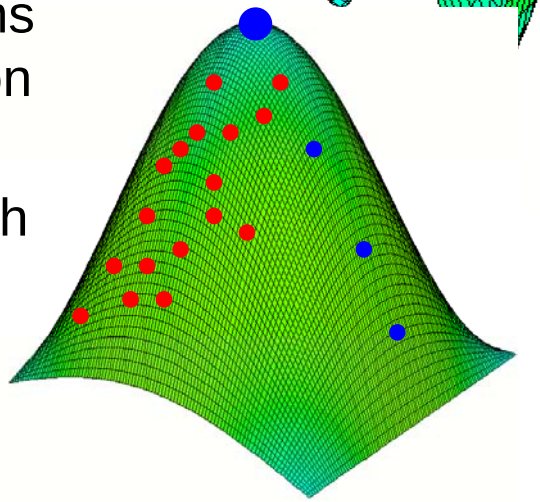
- Evolutionary algorithms have good exploratory features →
- Local search have bad exploratory features →



Global search

Exploration

- Evolutionary algorithms have not high exploitation features ●
- Local search have high exploitation features ●



Local search

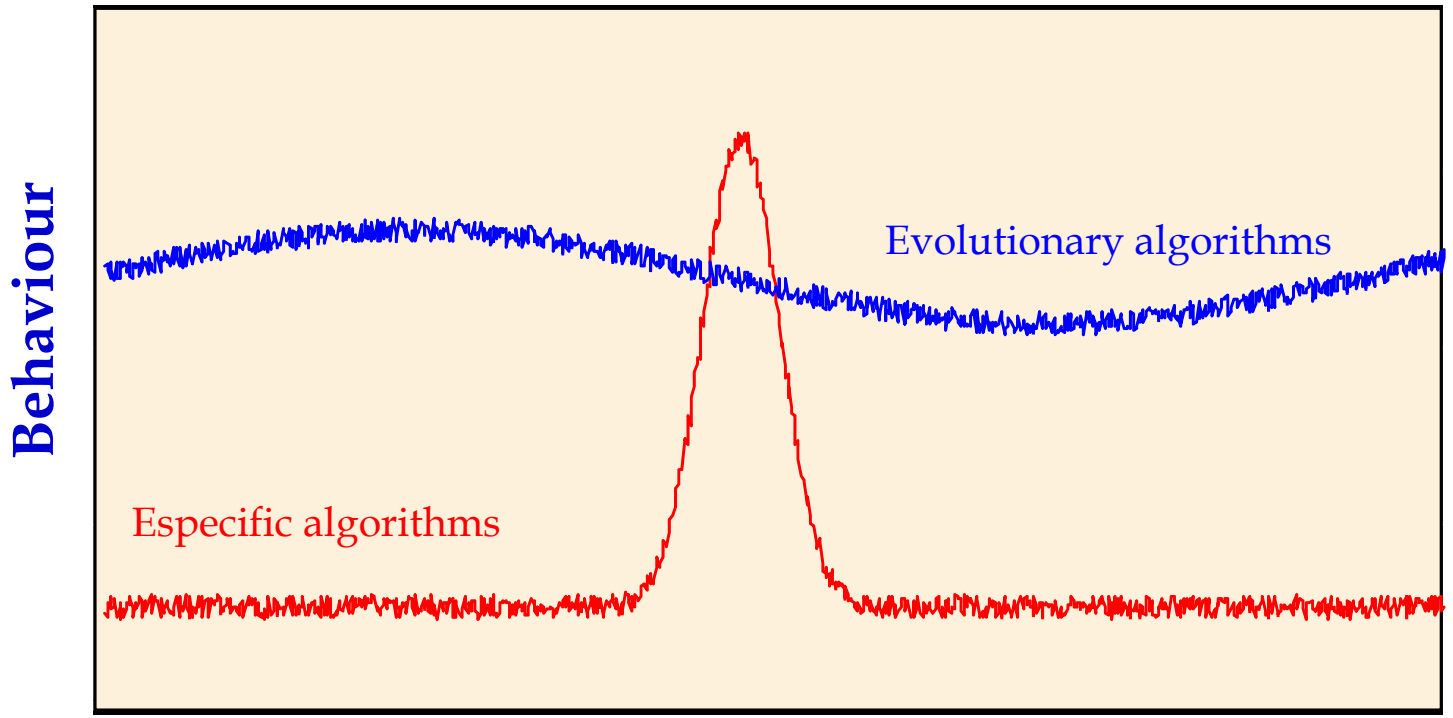
Exploitation

Evolutionary Computation: A New Way to Search for Solutions

Why this hybrid model?

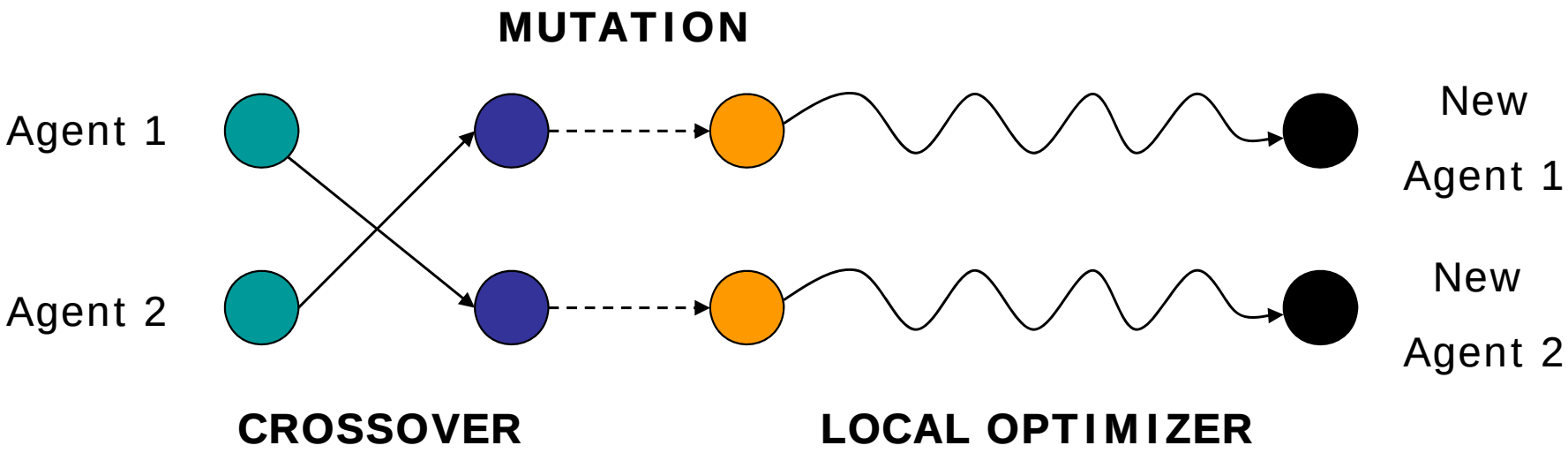
The limits of the EAs

On the behaviour of EAs



Evolutionary Computation: A New Way to Search for Solutions

Memetic Algoriths: Basic concepts



N. Krasnogor and J.E. Smith. A tutorial for competent memetic algorithms: model, taxonomy and design issues. IEEE Transactions on Evolutionary Computation 9(5):474- 488, 2005.

M. Lozano, F. Herrera, N. Krasnogor, D. Molina, Real-Coded Memetic Algorithms with Crossover Hill-Climbing. Evolutionary Computation 12:3 (2004) 273-302.

Evolutionary Computation: A New Way to Search for Solutions

CONCLUDING REMARKS

Evolutionary Algorithms

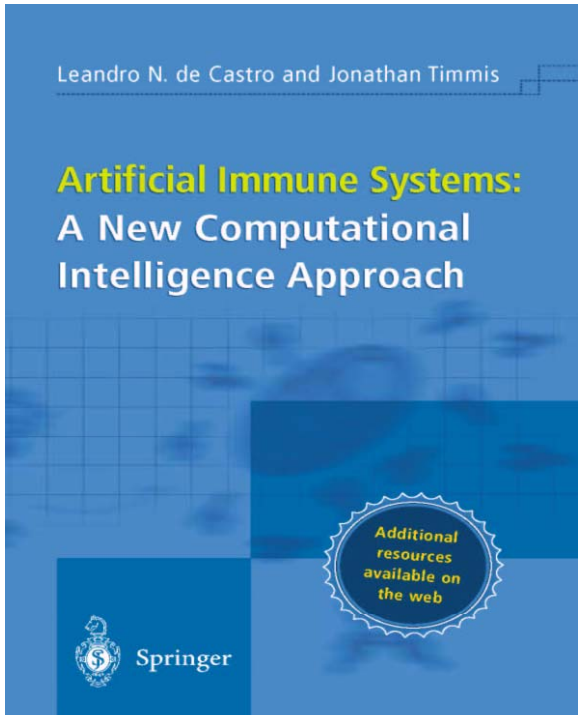
- Based in a biological **metaphor**: evolution
- high applicability
- very popular
- High performance and low cost
- Powerful algorithms for a lot of applications

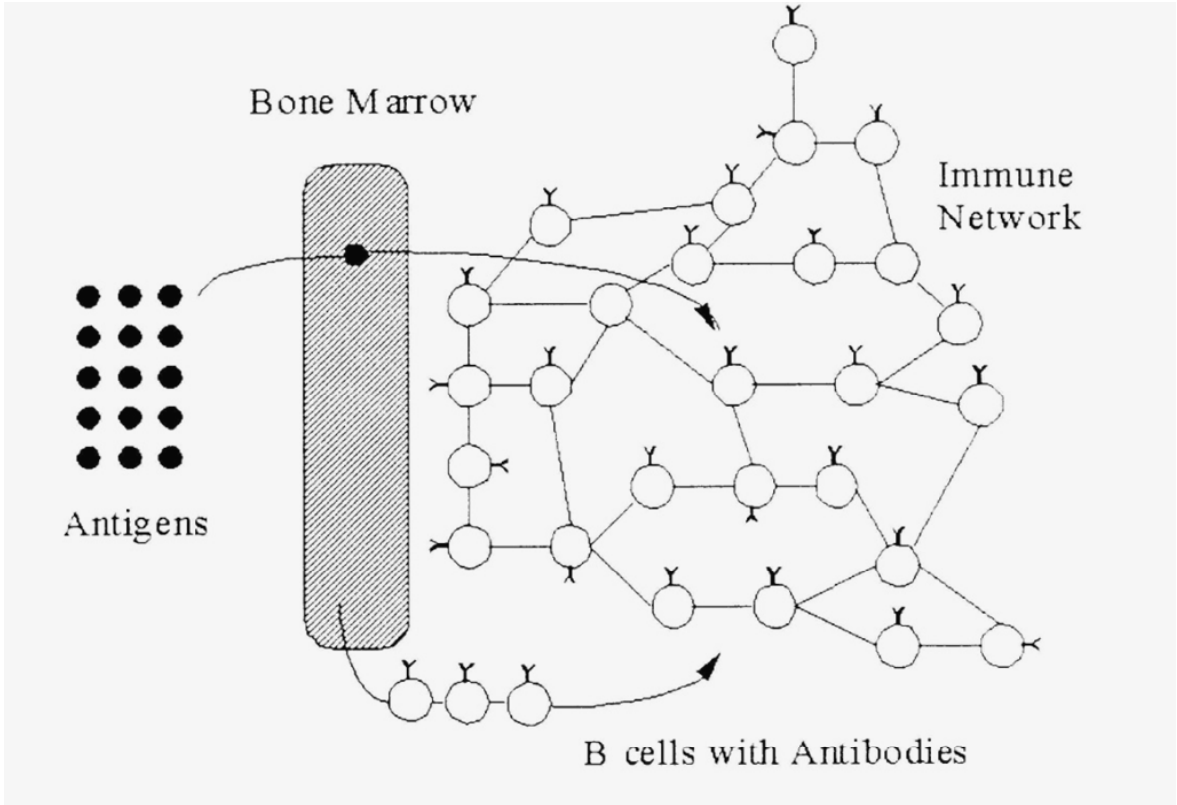
Contents

Artificial Immune System (AIS)

- ***"Artificial immune systems are computational systems inspired by theoretical immunology and observed immune functions, principles and models, which are applied to complex problem domains"***

  (de Castro & Timmis)



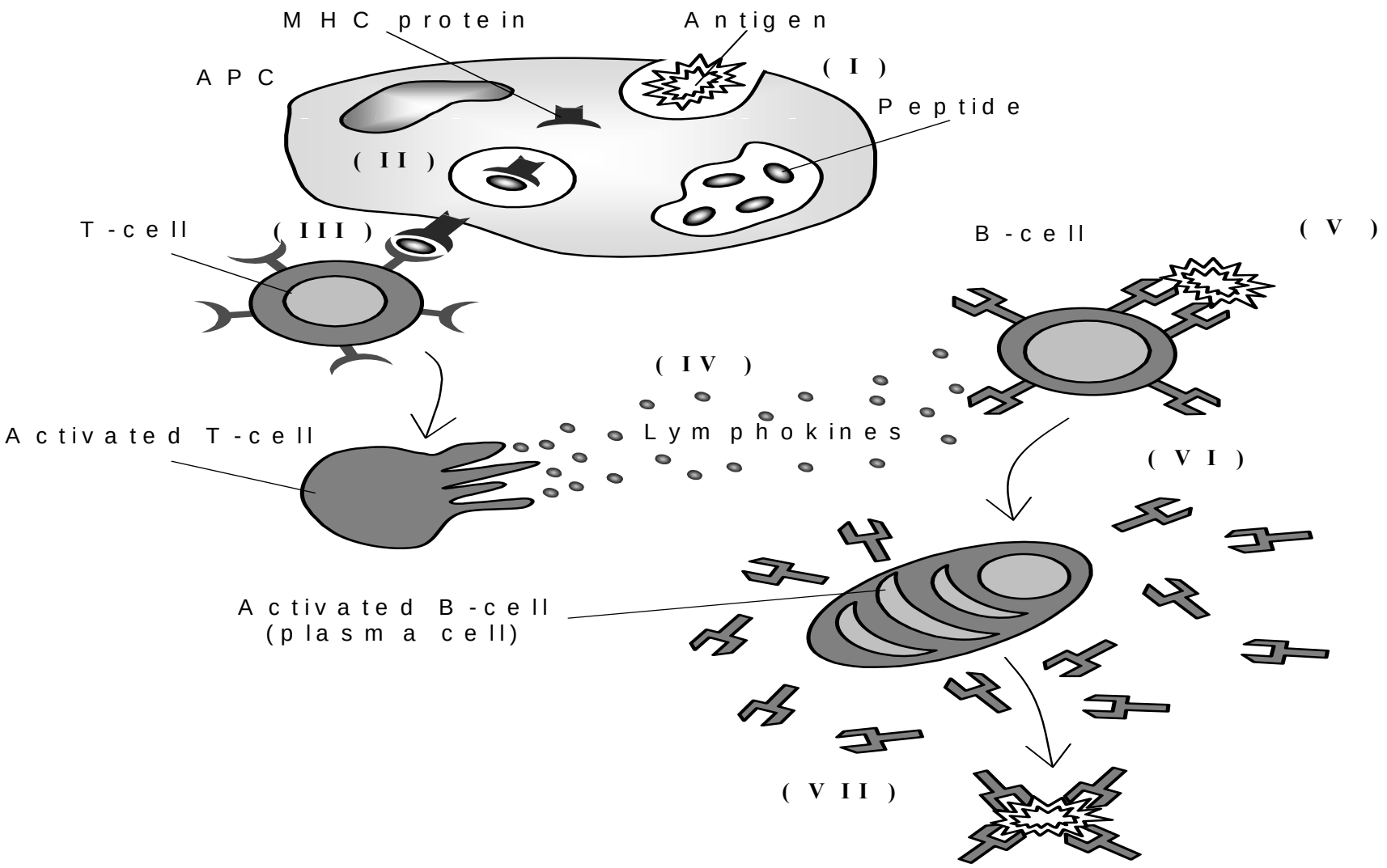


Artificial Immune System (AIS)

Why the immune system?

- ***Recognition*** **– Ability to recognize pattern that are (slightly) different from previously known or trained samples, i.e. capability of anomaly detection**
- ***Robustness*** **– Tolerance against interference and noise**
- ***Diversity*** **– Applicability in various domains**
- ***Reinforcement learning*** **– Inherent self-learning capability that is accelerated if needed through reinforcement techniques**
- ***Memory*** **– System-inherent memorization of trained pattern**
- ***Distributed*** **– Autonomous and distributed processing**

Artificial Immune System (AIS)



Artificial Immune System (AIS)

Self/Non-Self Recognition

- **Immune system needs to be able to differentiate between self and non-self cells**
- **Antigenic encounters may result in cell death, therefore**
  - **Some kind of *positive selection***
  - **Some element of *negative selection***

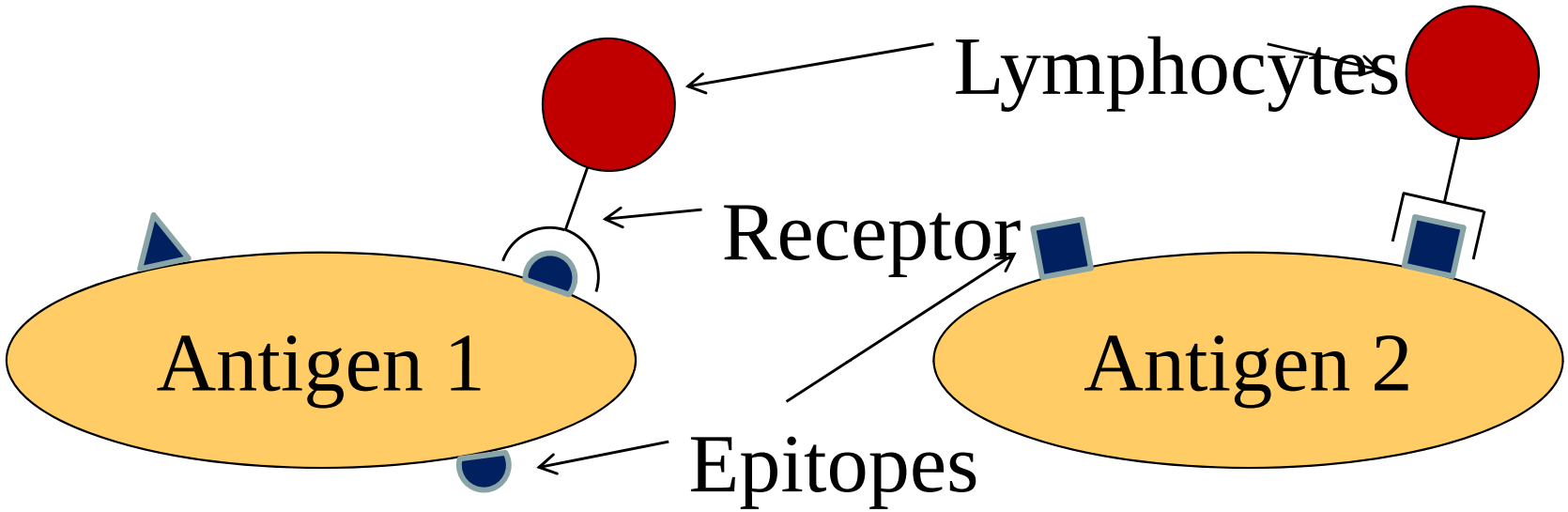
- **Primary immune response**
  - **Launch a response to invading pathogens**
    → unspecific response (Leucoytes)

- **Secondary immune response**
  - **Remember past encounters (immunologic memory)**
  - **Faster response the second time around**
    → specific response (B-cells, T-cells)

Artificial Immune System (AIS)

Immune Pattern Recognition

- **The immune recognition is based on the *complementarity* between the binding region of the receptor and a portion of the antigen called *epitope***
- **Antibodies present a single type of receptor, antigens might present several epitopes**
  - **This means that different antibodies can recognize a single antigen**



Artificial Immune System (AIS)

AIS – Application Examples

- Fault and anomaly detection
- Data mining (machine learning, pattern recognition)
- Agent based systems
- Autonomous control and robotics
- Scheduling and other optimization problems
- Security of information systems

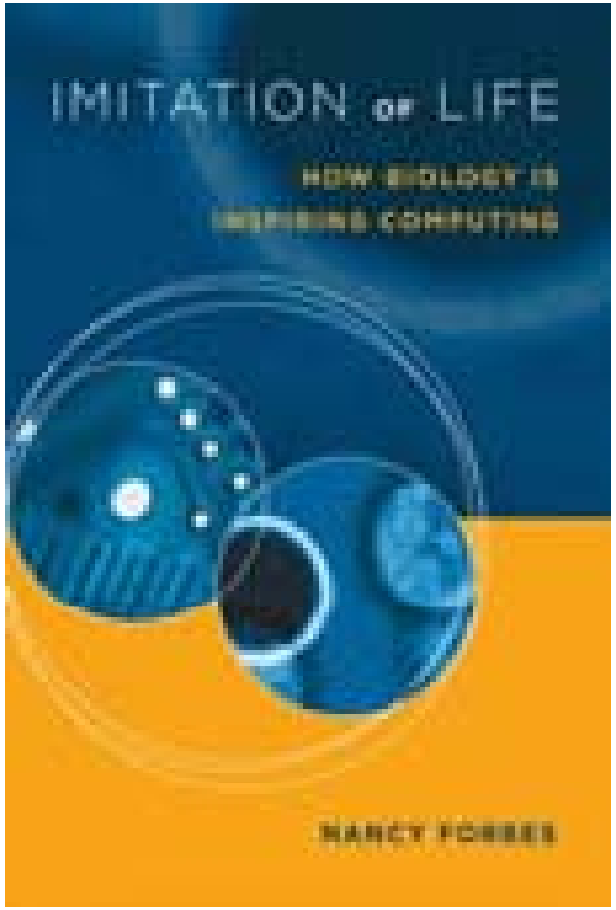
Contents

Bioinspired Research - Others

Nancy Forbes

IMITATION OF LIFE. How Biology Is Inspiring Computing.

The MIT,2004



Bioinspired Research - Others

Bio-inspired research – others

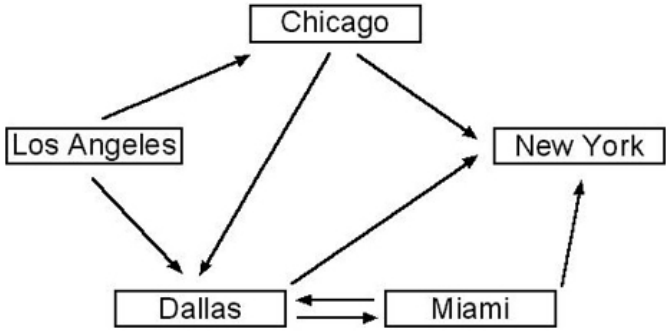
- Swarm intelligence (SI)
- Cellular automata
- Artificial life
- DNA Computing
- …

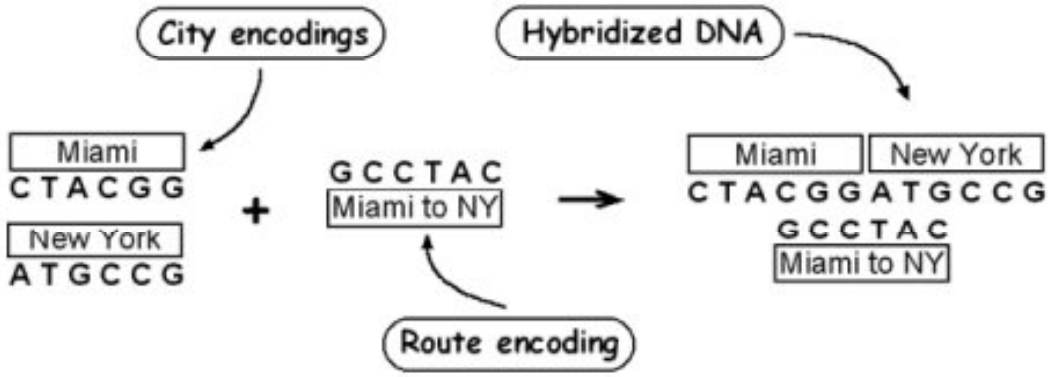
Bioinspired Research - Others

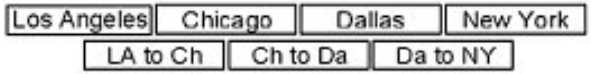
DNA Computing

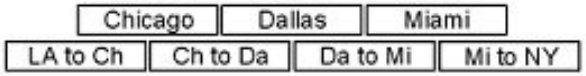
See also *DNA primer* on www.howstuffworks.com

Exploit replication and binding of biological DNA to solve large combinatorial problems: e.g. TSP (Adleman, 1994, Science)





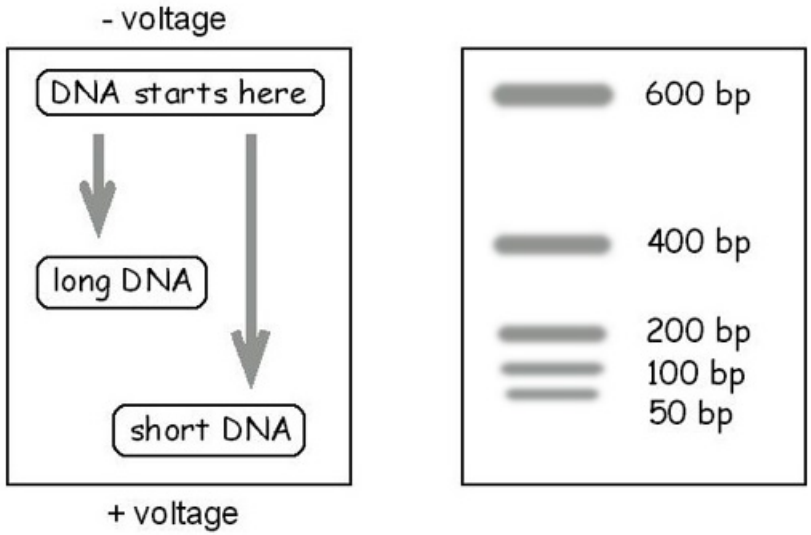


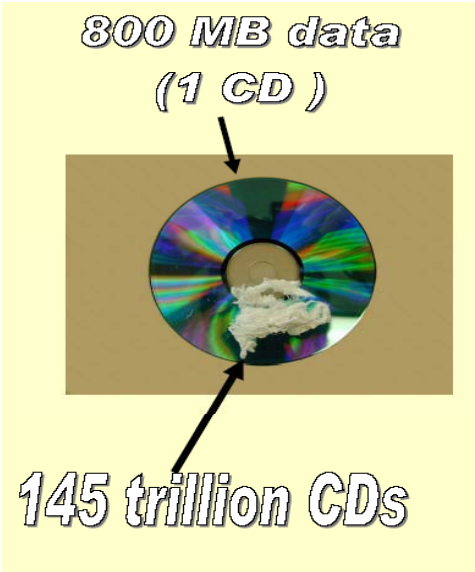


- **Create DNA**
- **Create DNA routes**
- **Make many copies with PCR**
- **Mix all DNA strings in test tube**

Bioinspired Research - Others

DNA Computing





Problems of "DNA computers":

-number of strands

- laborious process

- restricted to few problems

Select only strands with good size

(e.g., 5 cities x 6 bases = 30 bases)

Create baits:

compliment of city + magnetic bead

Fish out all strands starting with 1st city

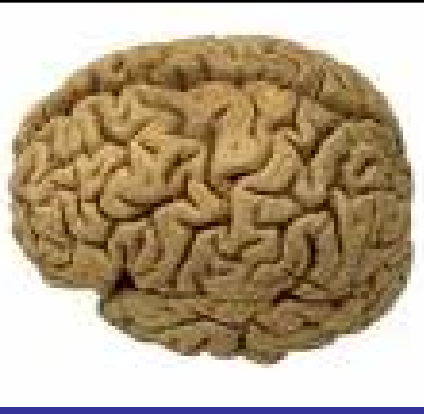
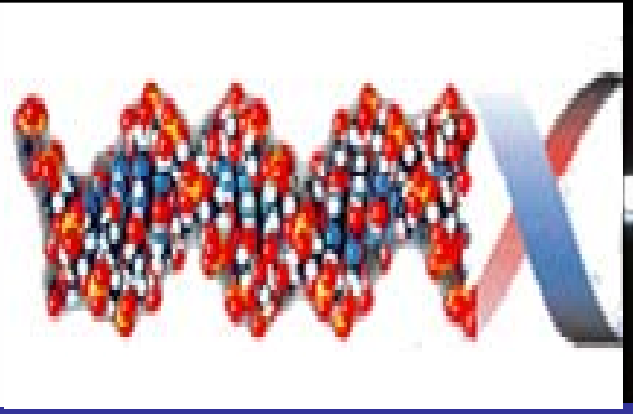
Out of those, fish out all strands with 2nd city

Out of those, fish out all strands with 3rd city

Out of those, fish out all strands with 4th city

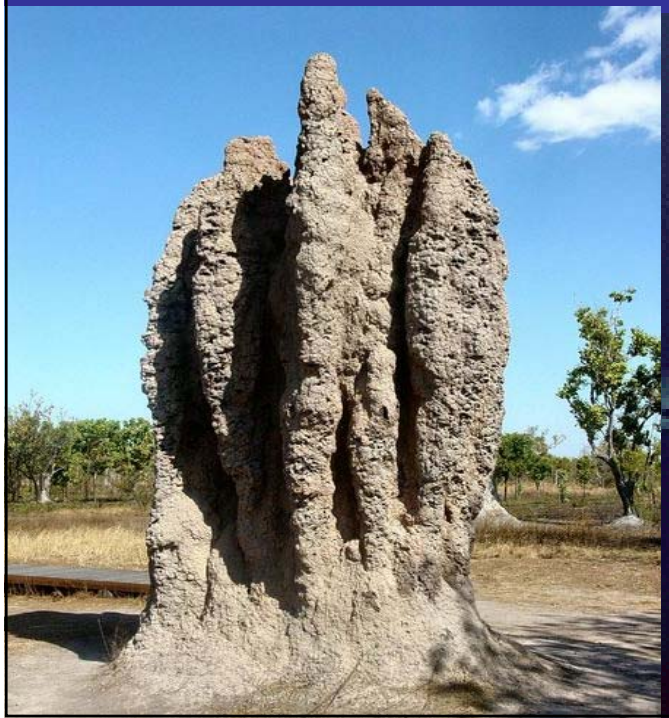
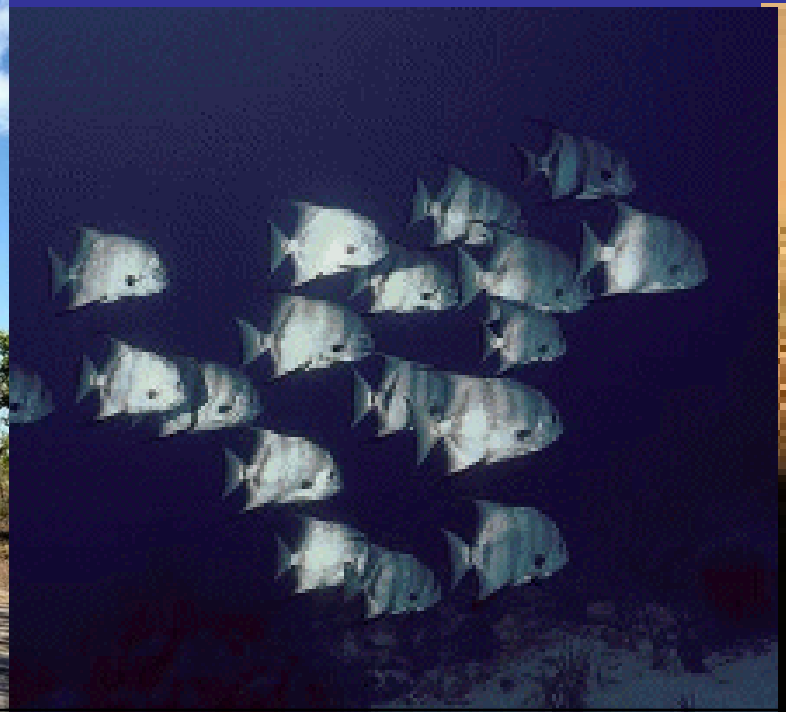
Repeat for all cities until…

Remaining strands represent the best route


Collective Intelligence: from neurons and genes to ants and agents

"Dumb parts, properly connected into a swarm, yield to smart results"

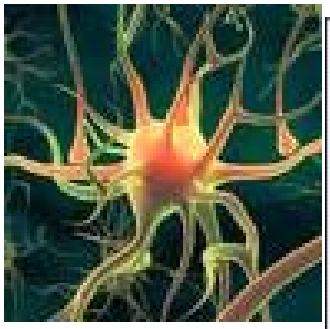

Contents

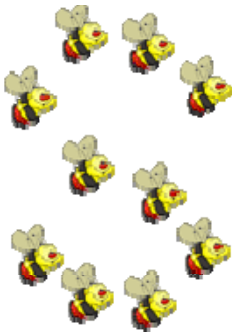
Final Comments

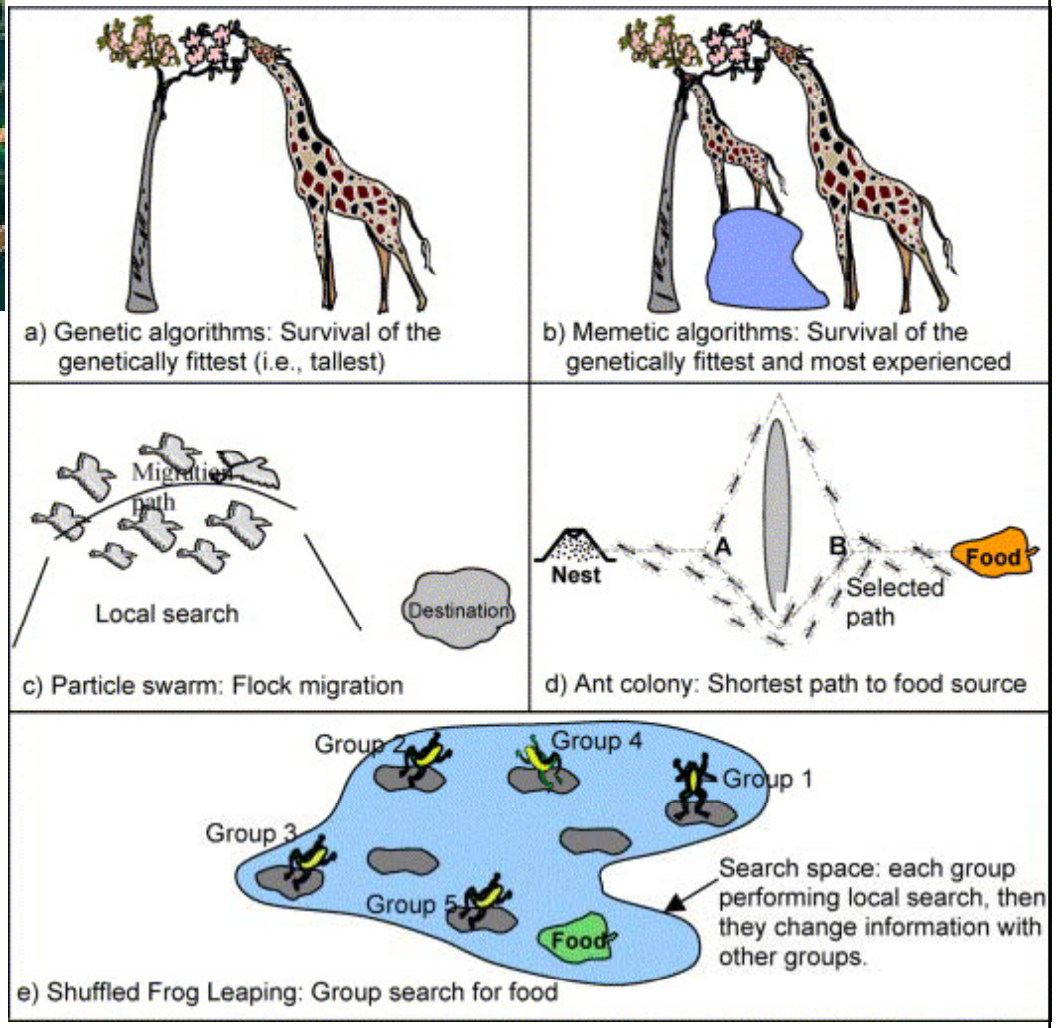
There are a lot of bio-inspired computation based areas tackling different problems, inspired by different biological mechanisms.

This is an exciting research area that we can consider at the beginning.

Success in AI will arrive from success from Bio-inspired Computing.

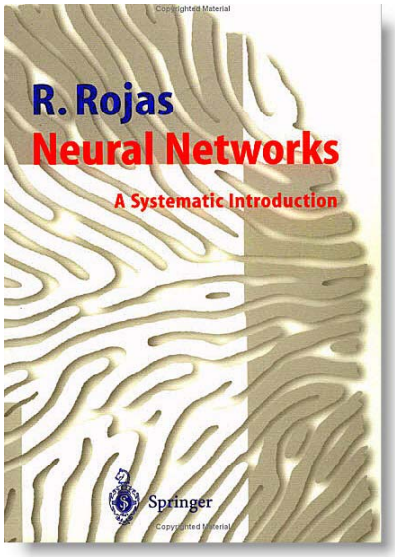


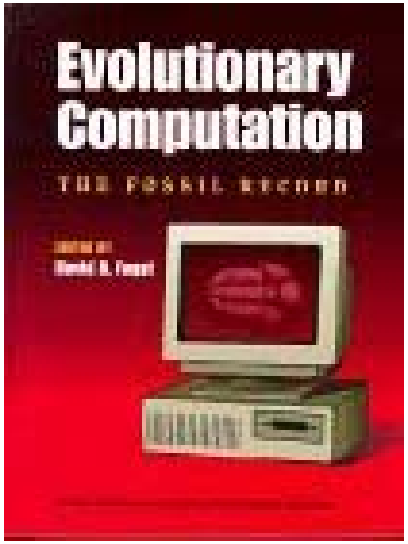


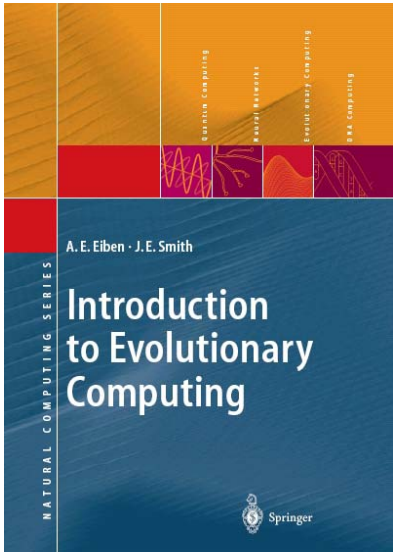


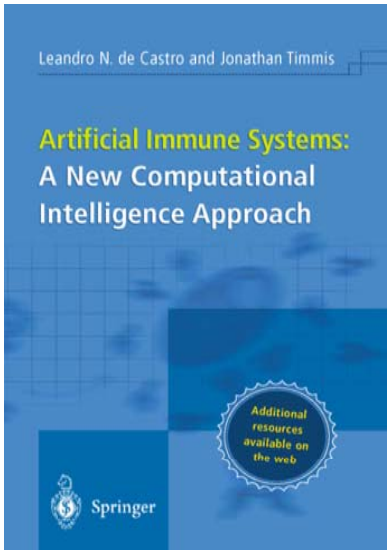
Final Comments

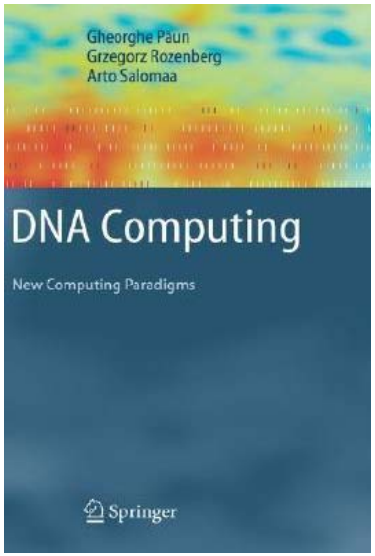
A few books

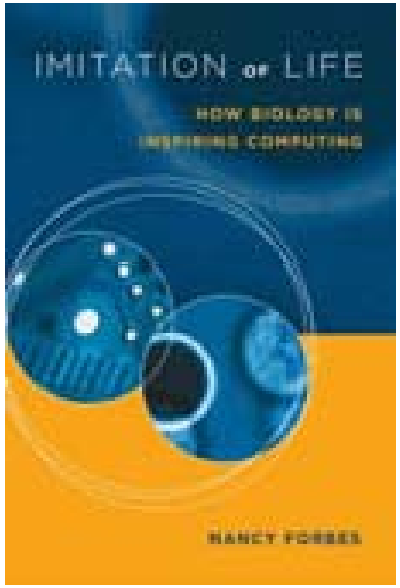












Computación Bioinspirada: Cuando la informática imita a los seres vivos

Thanks !!!

Any Questions?