

# Introduction to Evolutionary Computation

Inteligencia Artificial en los Sistemas de Control Autónomo  
Máster en Ciencia y Tecnología desde el Espacio

Departamento de Automática

## Objectives

- Describe local search algorithms
- Introduce artificial evolution
- Justify the utility of artificial evolution from an engineering perspective
- Overview the components of an Evolutionary Algorithm

## Bibliography

- Russell, S., Norvig, P. Artificial Intelligence: A modern approach, chapter 4. 3rd edition. Ed. Prentice-Hall. 2009
- Luke, S. Essentials of Metaheuristics. 2nd edition. Ed. Lulu, 2010. ([Link](#))

# Table of Contents

## 1. Local search algorithms

- Introduction
- Hill-climbing search
- Simulated annealing search
- Local beam search

## 2. Evolutionary Computation background

- Historical review
- Theory of Evolution
- Molecular Genetics
- Theory of Evolution from an algorithmic perspective

## 3. Evolutionary Algorithms

- Evolution as optimization
- AI, search and optimization

## ■ Metaheuristics

- Basics
- Exploration and exploitation

## 4. EAs components

- Components of an EA
- Representation
- Evaluation
- Selection
- Genetic operators

## 5. Case studies

- EAs examples
- Case study I: Transonic wing shape optimization
- Case study II: Mars orbital insertion
- Case study III: 9<sup>th</sup> GTOC

# Local search algorithms

## Introduction (I)

- In many optimization problems, the path to the goal is irrelevant; the goal itself is the solution
- State space = set of “complete” configurations. Find configuration satisfying constraints, e.g., n-queens
  - In such cases, we can use local search algorithms
- Keep a single “current” state, try to improve it
  - Work with one current state and generally moves to the neighboring state
- The paths followed by the search are not retained
  - They use little memory
  - You can find reasonable solutions in large state spaces or infinite

# Local search algorithms

## Introduction (II)

Example: Put  $n$  queens on a  $n \times n$  board with no two queens on the same row, column or diagonal



## Local search algorithms

- Hill-climbing search
  - Simulated annealing search
  - Local beam search
  - Genetic Algorithms

# Local search algorithms

## Hill-climbing (I)

It's just a loop that moves in the direction of increasing value

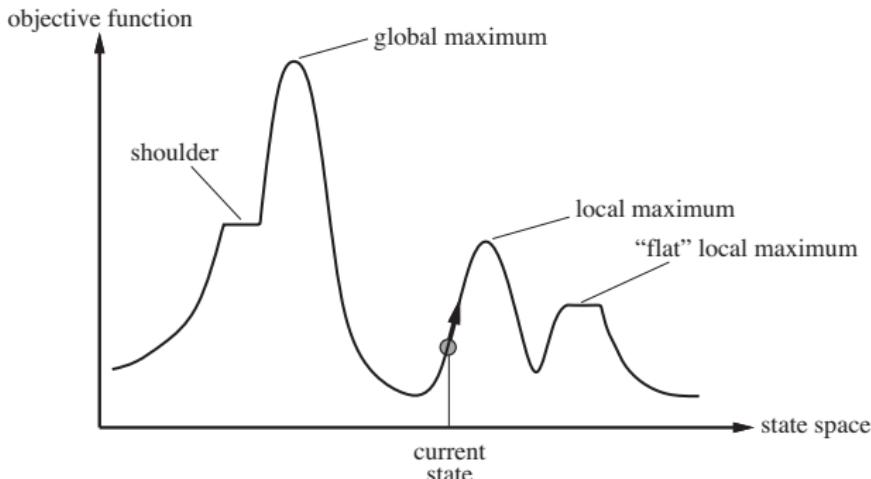
- Ends when it reaches a peak where no neighbor has a higher value
  - The search tree is not kept, just a data structure of the current node to check the goal condition and its objective function value

“Like climbing Everest in the thick fog with amnesia”

# Local search algorithms

## Hill-climbing (II)

Problem: depending on initial state, can get stuck in local maxima

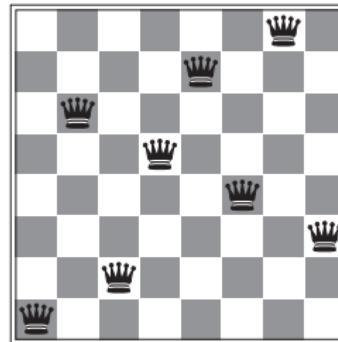


# Local search algorithms

## Hill-climbing (III)

Example: 8-queens problem

18	12	14	13	13	12	14	14
14	16	13	15	12	14	12	16
14	12	18	13	15	12	14	14
15	14	14	14	13	16	13	16
14	14	17	15	15	14	16	16
17	15	16	18	15	15	15	15
18	14	15	15	15	14	15	16
14	14	13	17	12	14	12	18



- $h = \text{number of pairs of queens}$
- $h = 17$  for the above state
- Values of all successors, top successors have  $h = 12$

Local minimum with  $h = 1$  after 5 steps

## Local search algorithms

## Hill-climbing (IV)

The algorithms gets stuck for several reasons

- Local maximum
  - Ridges
  - Plateau

In the 8-queens, it gets stuck in 86 % and solve 14 %

- If we allow lateral movements, 94 % success

## Variants

- Stochastic hill-climbing (randomly chooses upward movements)
  - Random restart (the initial state is generated randomly)

# Local search algorithms

## Simulated annealing search

- Simulated annealing is the process of tempering metals by heating and then cooling them gradually
- Idea: escape local maxima by allowing some “bad” moves but gradually decrease their frequency
- It combines hill-climbing with random generation successor
- One can prove: if  $T$  decreases slowly enough, then SA will find a global optimum with probability approaching 1
- Widely used in VLSI layout, airline scheduling, etc

(Animation)

# Local search algorithms

## Local beam search

Idea: Keep track of  $k$  states rather than just one

- Start with  $k$  randomly generated states
- At each iteration, all the successors of all  $k$  states are generated
- If any one is a goal, then stop; else select the  $k$  best successors from the complete list and repeat

Alternatively stochastic LBS randomly choose  $k$  successors, with the probability of choosing a successor as an increasing function of its value

# Biological background

## Historical review (I)

Anaximander of Miletus (610 – 546 BC)

- First animals come from water
- Man come from fishes

Plato (428/427 – 348/347 BC)

- Demiurgo created the cosmos
- Theory of Ideas

Aristotle (384 – 322 BC)

- Spontaneous generation
- Strong influence in Europe



# Biological background

## Historical review (II)



Creationism: God created all the species

- Literal interpretation of the Genesis
- Species are hierarchical
- Man has a superior position

Main school in Europe for centuries

# Biological background

## Historical review (III)

Georges Louis Leclerc (1707 - 1788)

- Speculated that species change
- Noticed the similarities between men and apes
- Could not provide a theory

Jean-Baptiste Lamarck (1744 - 1829)

- First to propose a theory of evolution
- Transmutation of Species
- Use strengthens/weakens organs
- Heritability of acquired characteristics

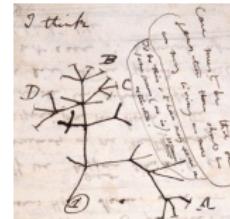


# Biological background

## Historical review (IV)

Charles Darwin (1809-1882)

- Published in “On the Origin of the Species” in 1859  
("On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life")
- Introduced natural selection ... and applies it to human being
  - Natural selection = Variability + selection
- Darwin did not explain the source of variation



# Biological background

## Historical review (V)

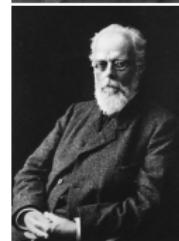
Gregor Mendel (1822 - 1884)

- Mendelian inheritance
- Recessive and dominant traits



August Weismann (1834 - 1914)

- Germ plasm theory
- Germ and somatic cells
- End of Lamarckism



J. Watson (1928) and F. Crick (1916 - 2004)

- Discovery of DNA
- Central Dogma of molecular biology



James Watson

Francis Crick

# Biological background

## Theory of Evolution

Neo-Darwinism: Darwin + Mendel + Weismann

- ... also called Theory of Evolution
- Variability + selection = evolution

There is variation among individuals

- Sexual reproduction, mutation and gene flow

There is a selection of those individuals

- Natural selection
- Artificial selection
- Sexual selection
- Genetic drift (deriva genética) (Link)

The fittest is the one that survives (not the strongest!)

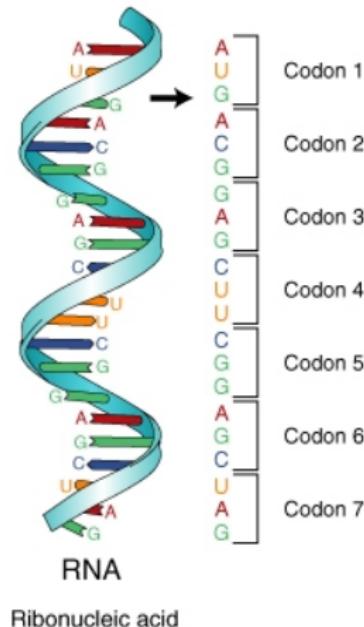
# Biological background

## Molecular Genetics (I)

Organisms are made by **proteins**

- Proteins are sequences of **aminoacids**
- They folder in a 3D structure
- 20 aminoacids

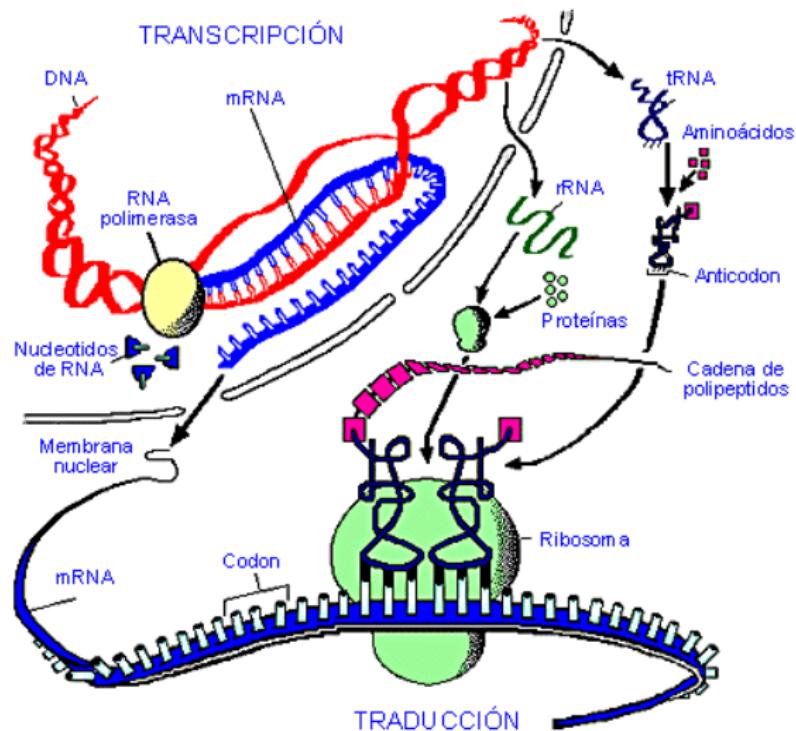
DNA codifies all the proteins in an organism



# Biological background

## Molecular Genetics (II)

### Protein synthesis: Creation of proteins from DNA (video)



# Biological background

## Molecular Genetics (III)

### Useful biological terms

Gene ADN fragment that codifies one protein

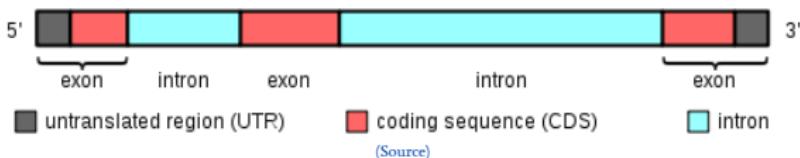
Allele The variant form of a gene

Genotype The sequence of DNA

Phenotype Characteristics of an individual

Exon Part of a gene that is transcribed

Intron Part of a gene that is not transcribed



# Biological background

## Theory of Evolution from an algorithmic perspective

Given a population ...

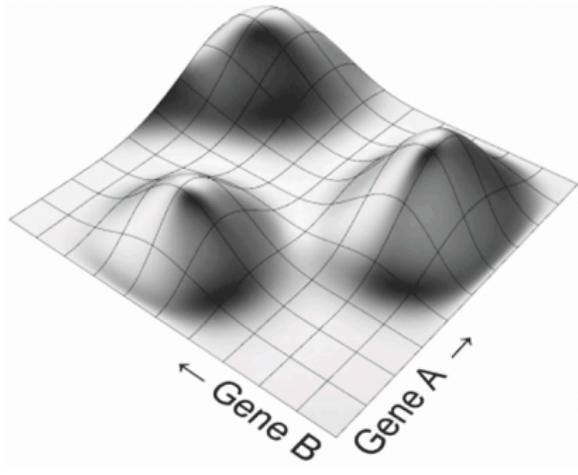
1. There are differences among individuals
2. Fittest individuals more likely to reproduce
3. Go to 1

We are interested in applying this to Engineering

How can we apply biological evolution to solve engineering problems?

# Evolutionary Algorithms

## Evolution as optimization



(Source)

Biological evolution is, in essence, an optimization algorithm

- ... it optimizes the survival probability
- Optimizing is to search the maximum

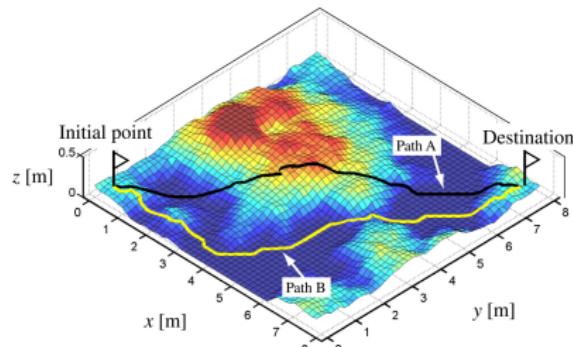
# Evolutionary Algorithms

## AI, search and optimization (I)

AI is much related to search a solution for a problem

- Search space
- Solution space

Almost any computational problem can be expressed as a search problem



(Source)

# Evolutionary Algorithms

## AI, search and optimization (II)

In AI, potential solutions are assessed

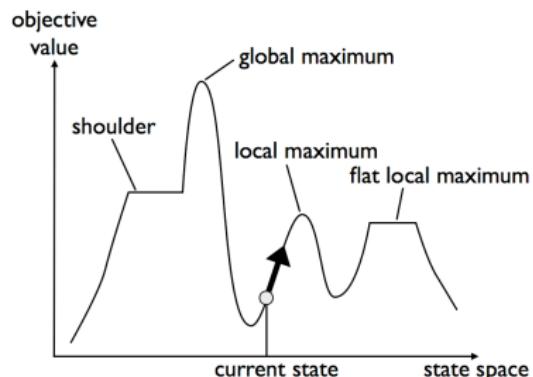
- Cost function
- Objective: Maximize cost function

The solution to any problem: **exhaustive search**

- Inviable in practice

How to find a solution efficiently?

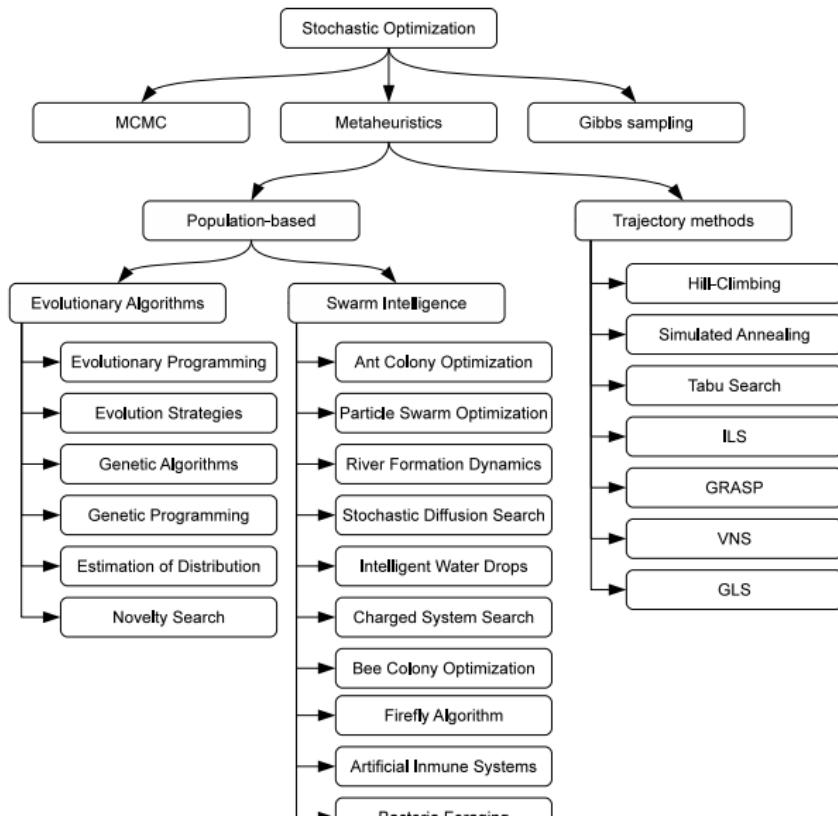
- With domain knowledge
- With randomness: **Metaheuristics**



(Source)

# Evolutionary Algorithms

## Metaheuristics



Again:

How can we apply biological evolution to solve engineering problems?

# Evolutionary Algorithms

## Basics (I)

Large number of Evolutionary Algorithms

- There is no “canonical” algorithm
- They all imitate biological evolution

They use a population

- Each individual represents a (potential) solution
- Multiple **representations**

Population is modified

- Mutation ( $1$  individual)
- Crossover ( $>1$  individuals)
- Multiple **genetic operators**

Selection that imitates natural selection

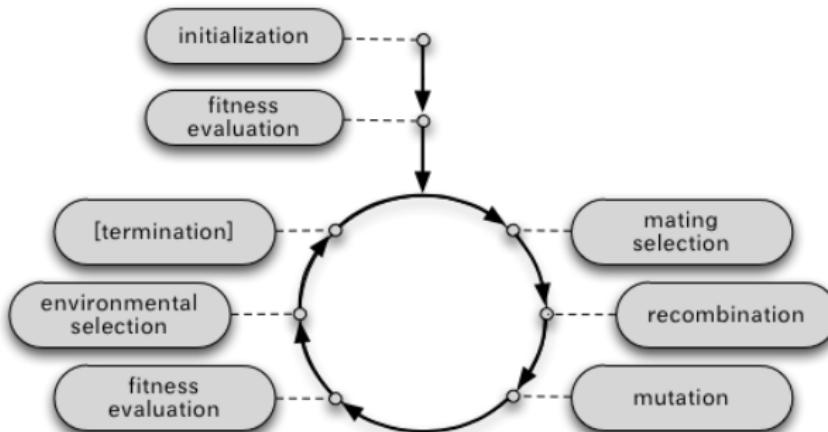
- Based on a **fitness** function

Iterative process

# Evolutionary Algorithms

## Basics (II)

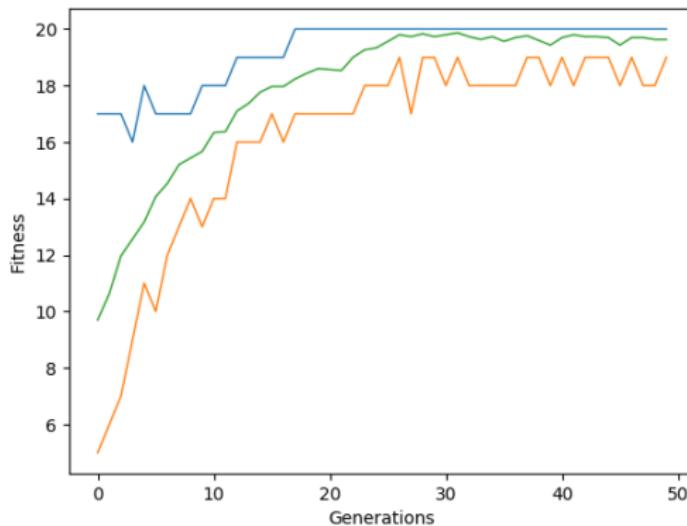
### Possible basic algorithm



# Evolutionary Algorithms

## Basics (III)

### Typical fitness behaviour



# Evolutionary Algorithms

## Basics (IV)

Initialization is usually random

- Random population
- Domain-dependent heuristics may be used
- Known solutions might be injected into the initial population

Termination criteria

- Get a desired fitness
- Maximum number of iterations (or generations)
- Loss of genetic diversity
- Lack of fitness improvement

# Evolutionary Algorithms

## Exploration and exploitation

### Balance between exploitation and exploration

- These are opposite objectives ⇒ Need of trade-off

#### Exploration: Search of new regions

- Explore the search space
- Performed, mostly, by mutation

#### Exploitation: Search of local (or global) maxima

- Exploit the acquired knowledge
- Performed, mostly, by crossover

# EAs components

## Components of an EA

### Common components in any EA

- Representation
- Evaluation
- Selection
- Genetic operators

## EAs components

## Representation (I)

Main difference among EAs is the representation

- Strings: Genetic Algorithms (GA)
  - Real vectors: Evolution Strategies (ES)
  - State machine: Evolutive Programming (EP)
  - Trees: Genetic Programming (GP)

These differences are, mostly, irrelevant

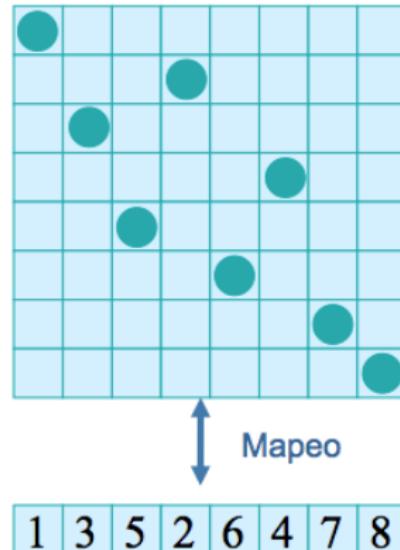
- Use the most natural representation
  - Use the most natural genetic operators according to the representation

## EAs components

## Representation (II)

## Example: 8 queens with a Genetic Algorithm

**Phenotype:** Board position  
**Genotype:** Integer vector



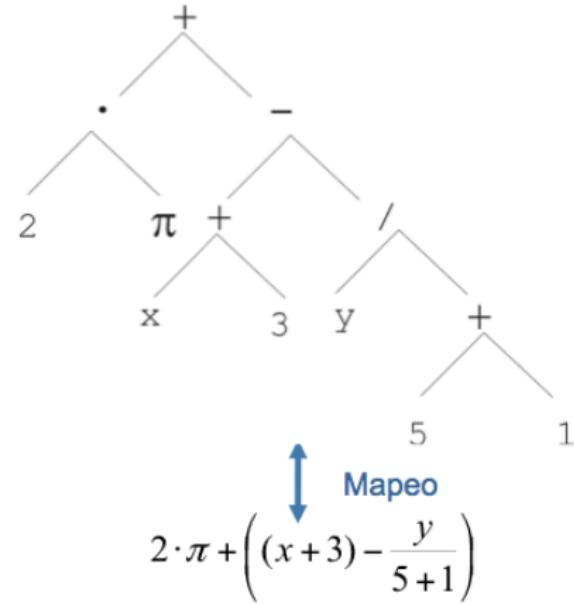
# EAs components

## Representation (III)

Example: Regression in Genetic Programming

**Phenotype:** Tree

**Genotype:** Formula



# EAs components

## Evaluation (I)

Individuals quality is assessed by a **fitness function**

- Individual = Potential solution

The fitness assigns a numerical value to a phenotype

- Caution, phenotype, not genotype
- Multiobjective algorithms use several fitnesses

Evaluation used to be a bottleneck

- Many times it involves simulating a system

Minimize number of evaluations

# EAs components

## Evaluation (II)

### Example: 8 queens

- The fitness may be the number of threaded pieces
- Objective: Minimize fitness (minimation problem)

### Example: Regression

- The fitness may be the quadratic average error
- Objective: Minimize fitness (minimation problem)

# EAs components

## Selection (I)

Selection operator “selects” individuals for reproduction

- Imitates natural selection
- Higher reproduction probability for high fitness individuals
  - Randomness helps avoiding local minima
- Selection is done in phenotypic space!
  - Selection does not take into account how representation is

Introduces selective pressure

# EAs components

## Selection (II)

High selective pressure reduces genetic diversity

- Faster evolution, higher probability of local maxima
- Eliminates low fitness individuals
  - Potentially valuable genetic material can be lost
- Selection operators: Tournament size  $n$ , roulette-wheel, rank-based, ...

### Tournament size $n$

1. Take randomly  $n$  individuals
2. Compute their fitness
3. Select the highest fitness

Variable selective pressure depending on  $n$

# EAs components

## Selection (III)

### Replacement strategy

- Select which individual replace

### Two basic strategies

- Generational algorithms: Replace all the offspring
  - Iterations are named **generations**
  - Time is usually measured in generations
- Steady-state: Replace part of the offscript
  - Criteria: Age, fitness, selection, etc
  - Lower memory consumption

### Hybrid strategy: Elitism

- Replace the population, except the  $n$  fittest individuals
- $n$  fittest individuals guaranteed to survive

# EAs components

## Genetic operators (I)

Genetic operators build new individuals

- Two basic operators: **mutation** and **crossover**

Open discussion (=research) about the role of mutation and crossover

- Mutation enhances exploration
- Crossover enhances exploitation

Both are used

- Historical constraints

# EAs components

## Genetic operators (II)

### Mutation operator

- It takes a genotype and returns another one
  - It has a stochastic behavior
  - Used to maintain genetic diversity
- Guarantees search space connectivity
- Mutation plays a disruptive role
  - Moves population to new regions

Example: 8 queens permutation operator



# EAs components

## Genetic operators (III)

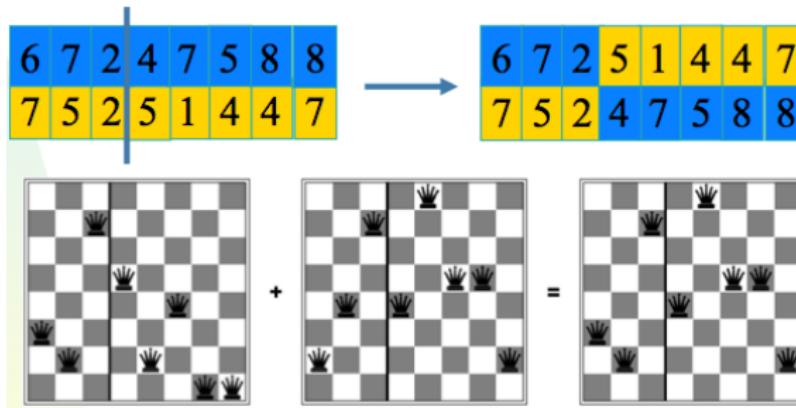
### Crossover operator

- Fuse information from the parents (sexual reproduction)
  - Randomness has a place
- Offspring uses to be worse than its parents
  - With luck, good components of the parents are joined ...
  - ... and this is something that happens
- Crossover has a constructive role
  - Join preexistent components
  - Does not generate new genetic material
  - Encourages exploitation

# EAs components

## Genetic operators (IV)

Example: 8 queens with one-point crossover



# EAs examples

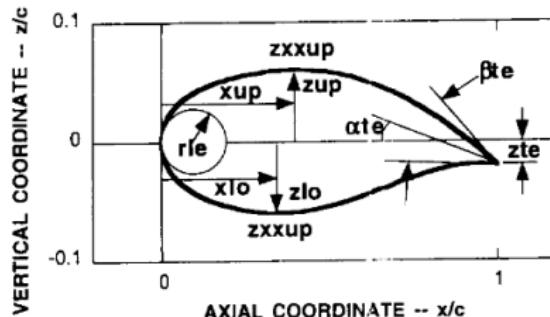
- (Car design)
- (Genetic Algorithm Walkers)
- (Smart rockets)
- (Learn to walk)
- (Flexible Muscle-Based Locomotion for Bipedal Creatures)
- (MarI/O - Machine Learning for Video Games)
- (A genetic algorithm learns how to fight!)
- (Evolved Electrophysiological Soft Robots)

# Study case

## Case study I: Transonic wing shape optimization

Problem: Design a wing shape for transonic flight

- Maximize lift



Holst T.L., Pulliam T.H. (2003) Transonic Wing Shape Optimization Using a Genetic Algorithm. In: IUTAM Symposium Transsonicum IV. Fluid Mechanics and its Applications, vol 73. Springer.

# Study case

## Case study II: Mars orbital insertion

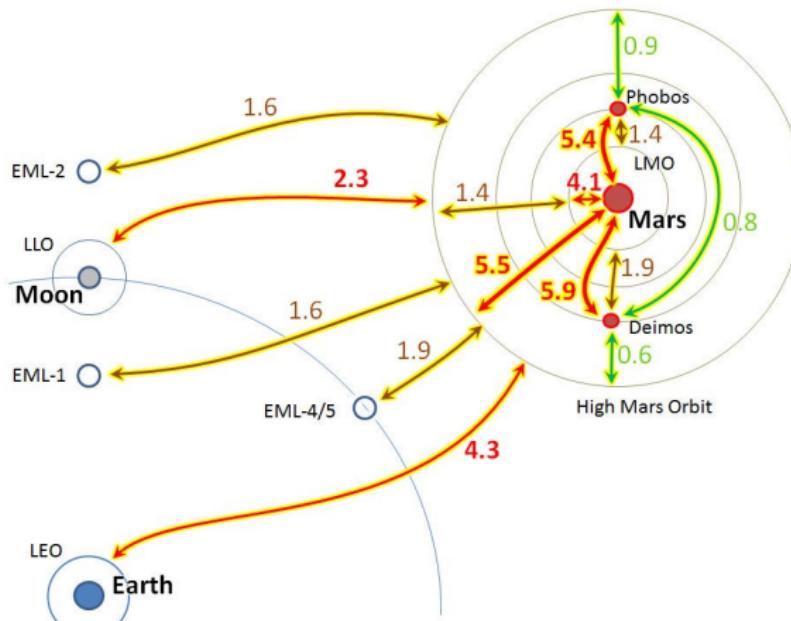


Chart by Richard Penn CC-BY, data from David Hollister [hopsblog-hop.blogspot.co.uk](http://hopsblog-hop.blogspot.co.uk)

# Study case

## Case study III: 9<sup>th</sup> Global Trajectory Optimization Competition

### GTOC: Global Trajectory Optimization Competition

- Proposed by ESA Advanced Concepts Team
- Difficult trajectory optimization problems

#### GTOC 9: The Kesser Run

- 123 orbiting debris
- Remove debris
- Design multiple missions

(Video) (Solution)