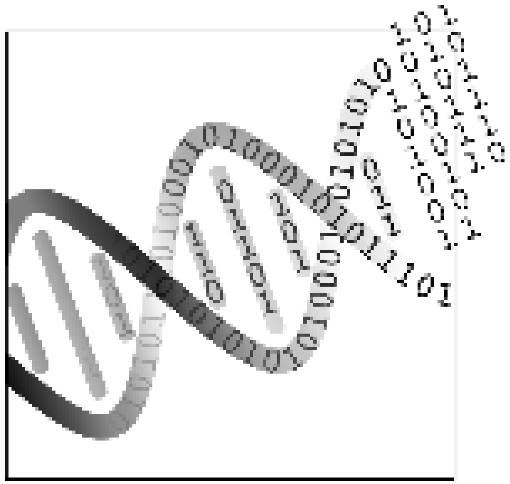
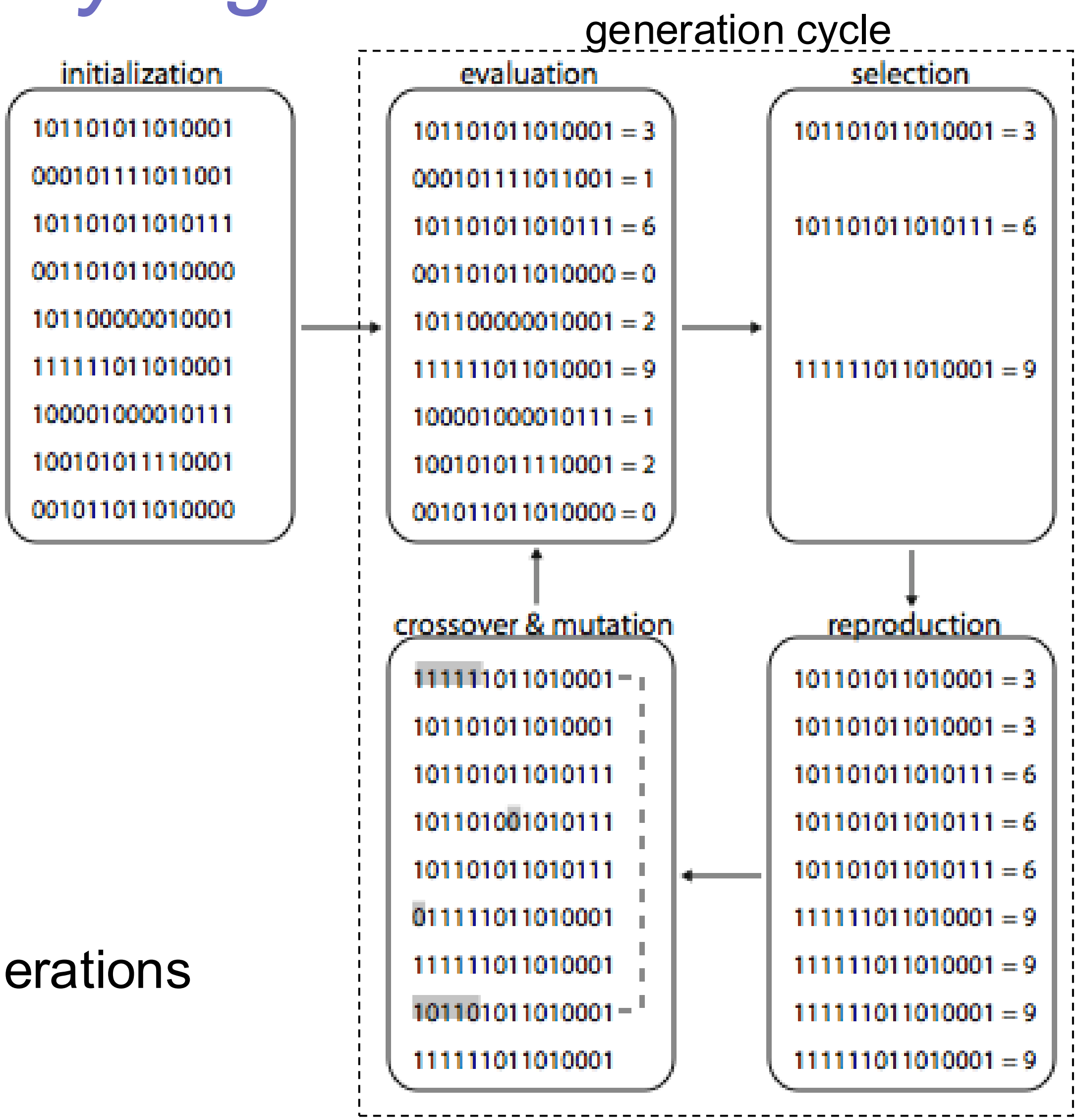
Evolutionary Computation Operators



Companion slides for the book *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies* by Dario Floreano and Claudio Mattiussi, MIT Press

Elements of an evolutionary algorithm

- Devise genetic representation
- Build a population
- Design a fitness function
- Choose selection method
- Choose crossover & mutation
- Choose data analysis method



<u>Repeat</u> generation cycle until:
- maximum fitness value is found
- solution found is good enough
- no fitness improvement for several generations

What you will learn in this lecture

Choosing a genetic representations

Building an initial population

Selection and reproduction methods

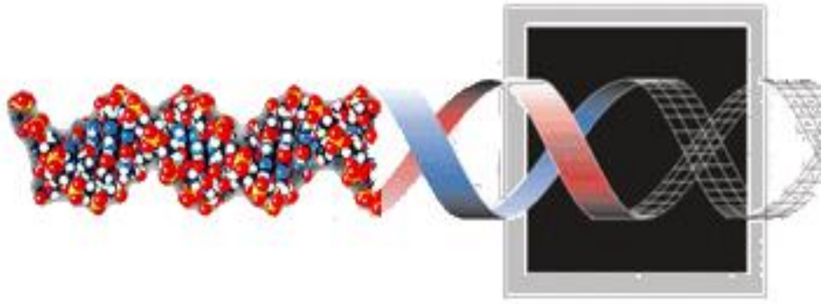
Mutations and crossover

Types of evolutionary algorithms

A simple example: evolving an antenna design

Genetic Representation

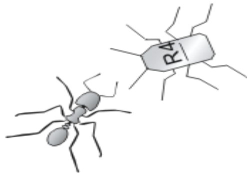
Coding of the phenotype (function variables, network weights, body parts, etc.) into a string



1001101010001

Simplification of biology:
- Single stranded sequence of elements
- Fixed length along generations, only genic
- Haploid structure and one chromosome
- Often one-to-one direct correspondence between genotype and phenotype

Types of representations:
- Discrete
- Real-valued
- Sequence
- Tree-based



Discrete Representations

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

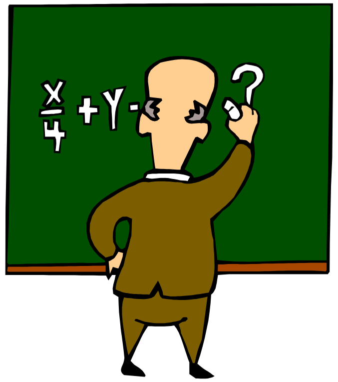
- E.g., binary string of 8 positions (l=8, k=2): 01010100
- Can be mapped into several phenotypes:

to configuration string of
FPGA electronic circuits

01010100

to integer *i* using
binary code

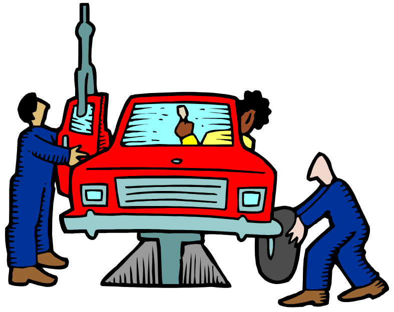
84



0.328125

to real value *r* in range [min, max]:
$r = min + (i/255)(max\text{-}min)$

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

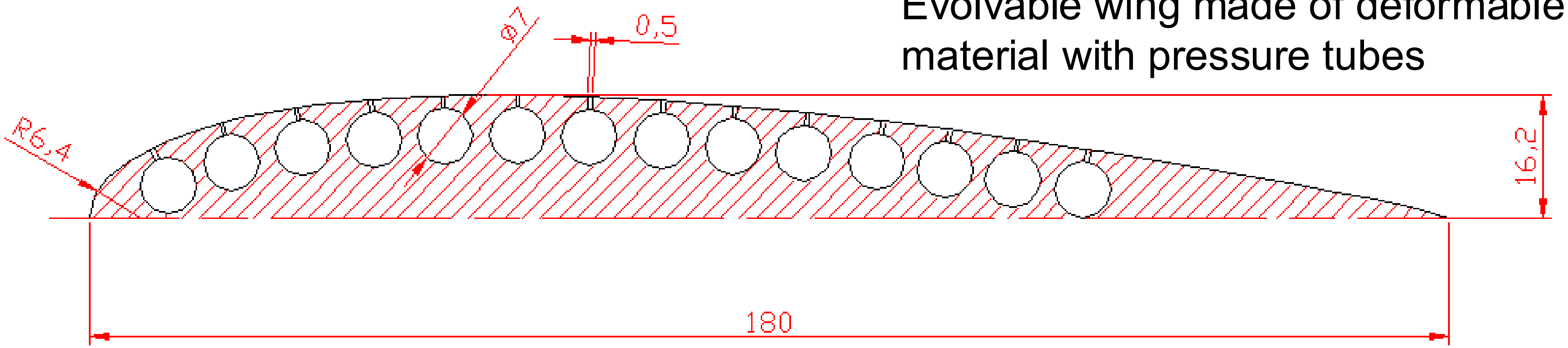
Real-Valued Representation

A vector of real values that represent parameters, e.g.: [0.6, -0.01, 0.87, …]

- This representation is used when high-precision parameter optimization is required, e.g.:
  - Variables of multi-dimensional function
  - Connection weights of neural networks
  - Parameters defining a shape

- Example: representation of wing profile for shape optimization

Evolvable wing made of deformable material with pressure tubes

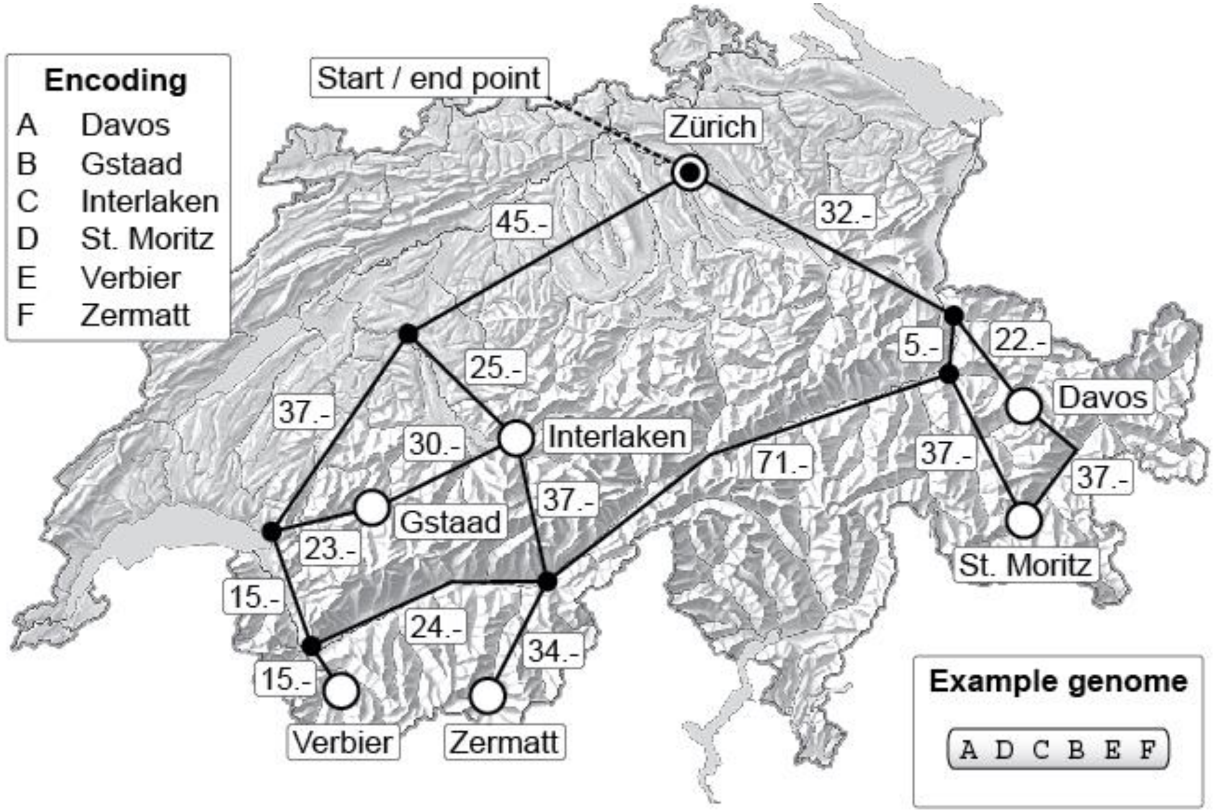


<u>Genotype</u>= 14 pressure values of the 14 tubes

Sequence Representation

A discrete representations where *l* = *k* and all *k* elements are included in each string.

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



Cardinality $k = 6$
String length $l = 6$

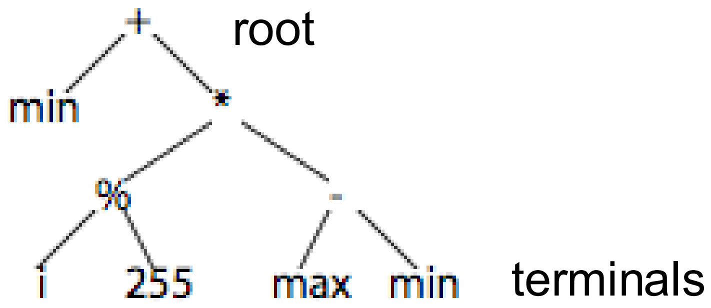
Tree-based Representation

A nested list of operators and operands that describes an expression or a computer program
It can be visualized as a tree with branching points and terminals

A computer program is an expression made of elements from:
- a Function set: `multiplication, sum, If-Then, Log`, etc.
- a Terminal set: constants, variables, sensor readings, etc.

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

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



A tree-based representation must satisfy two principles (conditions):
- Closure: all functions must accept all terminals of the Terminal set and all outputs of the Function set (e.g., instead of `/`, use protected division `%`)
- Sufficiency: elements of Function and Terminal sets must be sufficient to generate a program that solves the problem (e.g., `If` must be included in the Function set if a condition is required to solve the problem)

Build Initial Population

Sufficiently large to cover problem space, but sufficiently small for computational costs (typical size in the literature: between 10s and 1000s individuals). See also slides on Diversity

Create genetic strings by <u>random sampling</u> of genotype space. For example:

- Binary: for each location, set 0 or 1 with probability 0.5
- Real-valued: for each location, sample uniform or normal distribution centered at 0.0
- Sequence: distribute all elements k at random locations of each individual string
- Trees are built recursively starting from root:
  1. set maximum tree depth (e.g., max 5 levels)
  2. Root node is randomly picked from function set
  3. for every branch node, randomly pick from all elements of function set and of terminal set
     – if a terminal is picked, node becomes a leaf (end of the tree)
     – If node is level 5, randomly pick only from terminal set

Create genetic strings by cloning and mutating all locations of *best-guess string* (e.g., [0.0, 0.5])
Potential problems:

- Small genetic diversity
- Unrecoverable bias

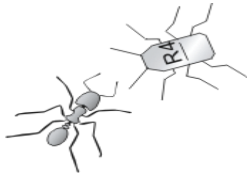
Selection

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

Popular methods:

- Proportionate selection
- Rank-based selection
- Truncated rank-based selection
- Tournament selection

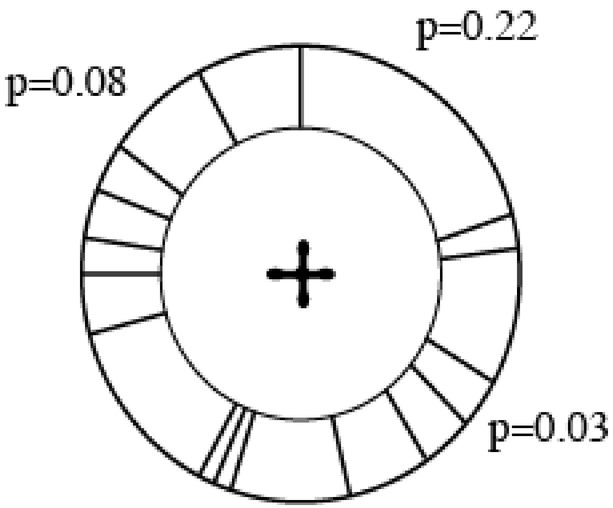
- Selection pressure is inversely proportional to percentage of selected individuals
- High selection pressure = rapid loss of diversity and premature convergence
- Make sure that also less performing individuals have a chance to reproduce

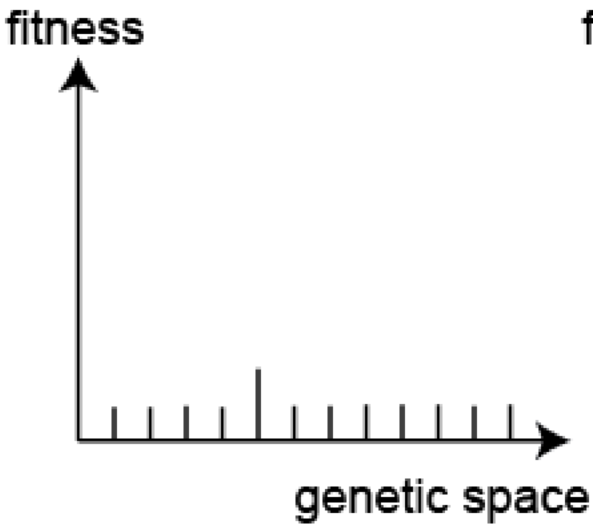


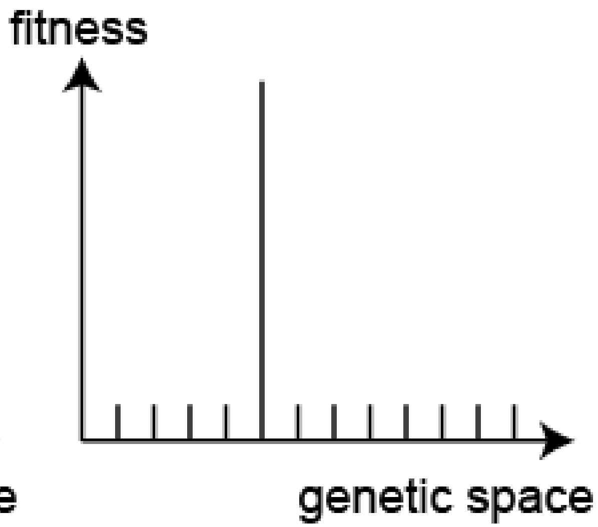
Proportionate Selection

The probability that an individual makes an offspring is proportional to how good its fitness is with respect to the population fitness: **p(i) = f(i)/Σf(i)**

Also known as *Roulette Wheel selection*



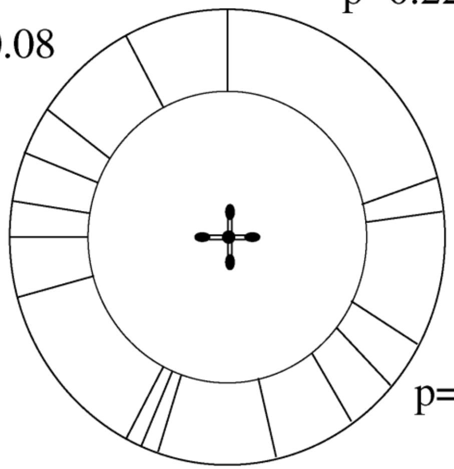




Potential problems:
Uniform fitness values = random search
Few high-fitness individuals = too high selection pressure

Rank-based Selection

- Individuals are sorted on their fitness value from best to worse. The place in this sorted list is called the rank r.
- Instead of using the fitness value of an individual, the rank is used to select individuals: **p(i) = 1 - r(i)/Σr(i)**
- Use roulette wheel



individual	fitness	rank
A	5	5
B	7	3
C	8	2
D	2	8
E	3	7
F	9	1
G	7	4
H	4	6

Addresses problem of proportional selection, but may require more generations

Truncated Rank-based Selection

Only the best **x** individuals are allowed to make offspring

Each selected individual makes the same number of offspring **N/x**, where **N** is the population size

E.g., in population of 100 individuals, make 5 copies of each of the 20 best individuals

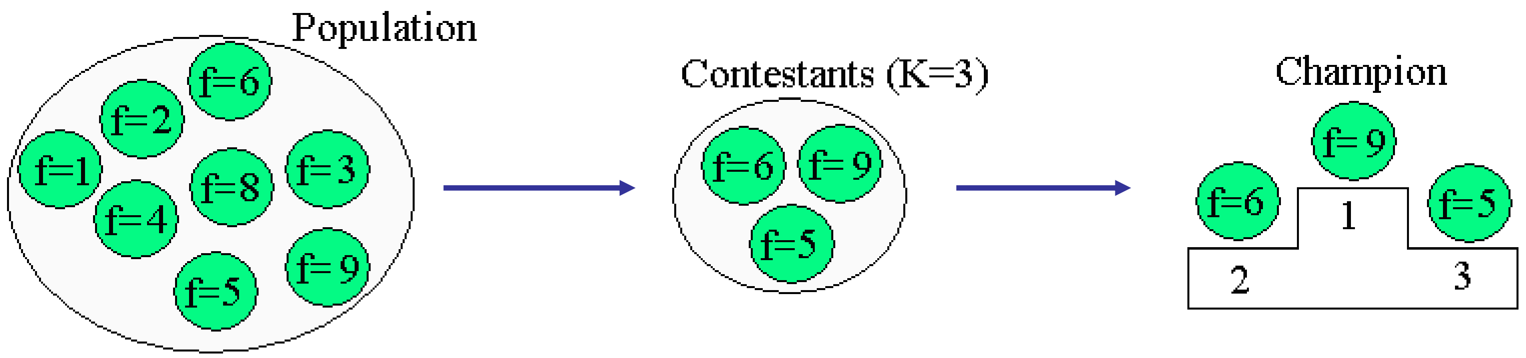
individual	fitness	rank	list
A	5	5	F
B	7	3	C
C	8	2	B
D	2	8	G
E	3	7	A
F	9	1	H
G	7	4	E
H	4	6	D

Addresses problem of Ranked-based selection, but may converge on local optima

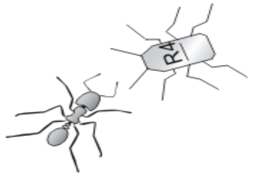
Tournament Selection

For every offspring to be generated:

- Randomly pick ***k*** individuals from the population
- Choose the individual with the highest fitness and make a copy
- Put all ***k*** individuals back in the population



k is the tournament size (larger size = larger selection pressure)



Generational Replacement

At each generation, all individuals are replaced by their offspring; population size is constant

Potential problem: mutations or poor fitness assessment may lead to loss of good individuals

Initial generation

Best individual

Next generation

Elitism: insert ***n*** best individuals from previous generation and randomly remove **n** offspring

Generational Rollover

Generate and insert one offspring at a time in the population and let it compete with older individuals

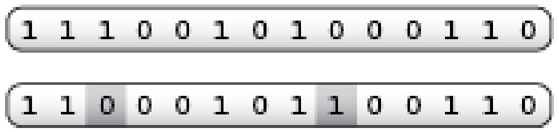
Methods to maintain constant population size

- Remove random individual
- Remove oldest individual
- Remove worst individual

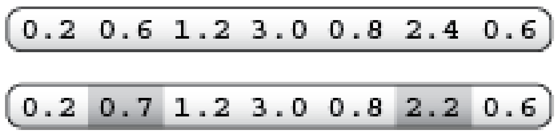
Mutation

Change each string location with probability p_m

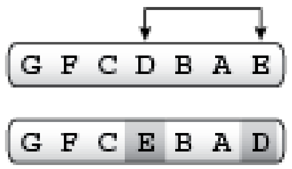
Binary genotypes



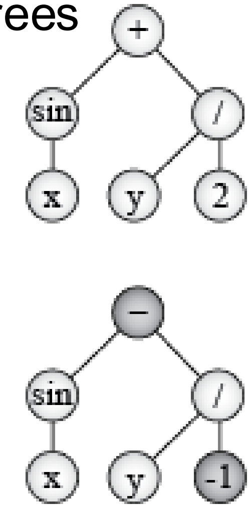
Real-valued genotypes
(uniform mutation)



Sequence genotypes

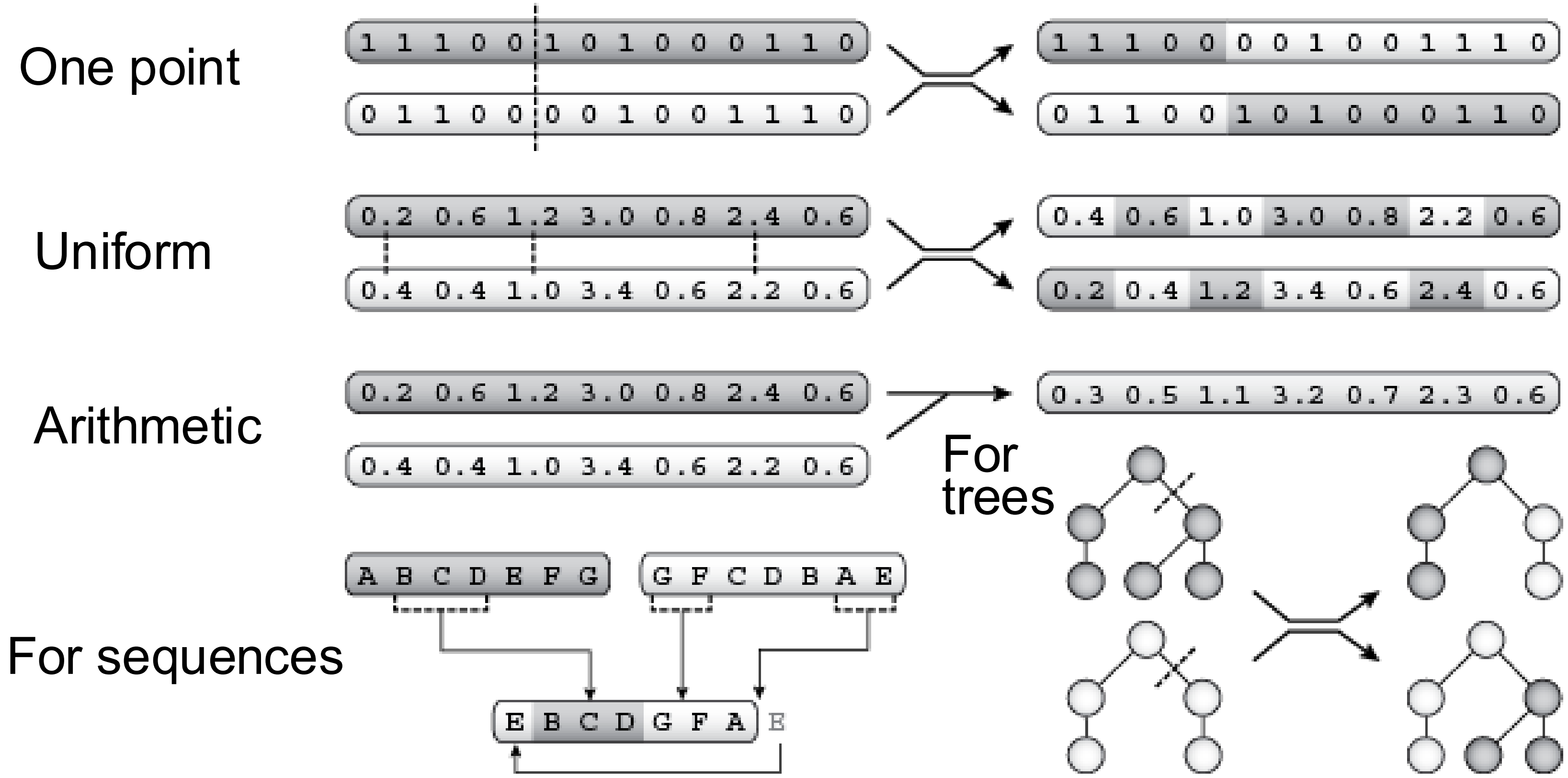


For trees



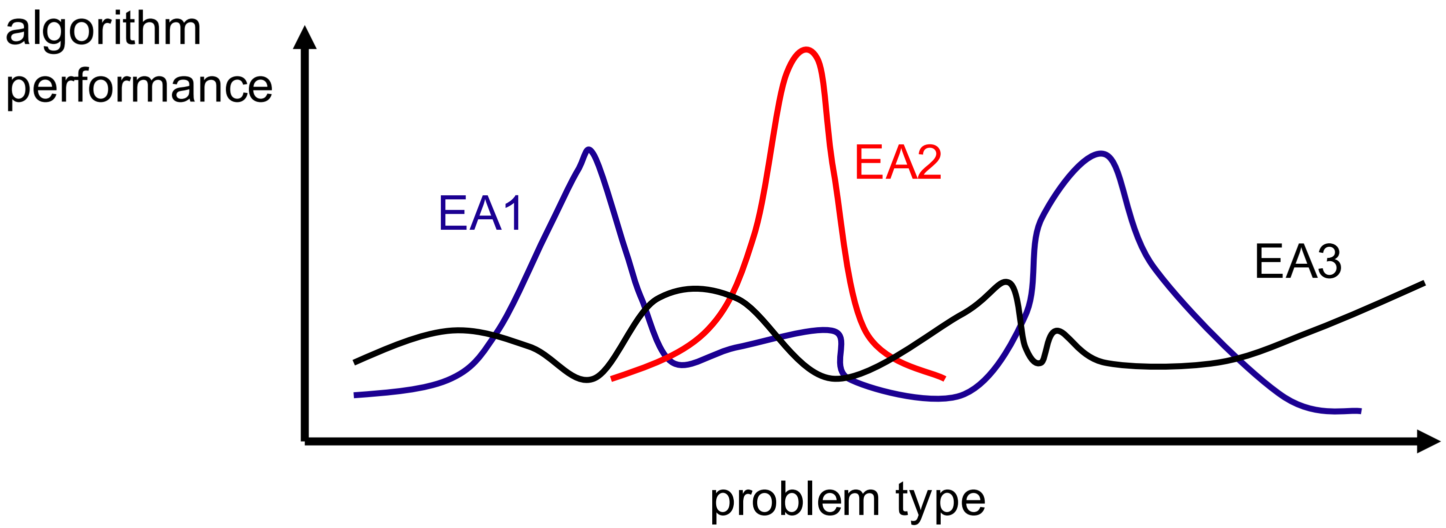
Crossover

Create random pairs of offprings
Recombine string parts of each pair probability p_c(pair)



A variety of evolutionary algorithms

- Evolutionary Algorithms differ in the choice of operators
- A specific evolutionary algorithm may perform best on a specific class of problems
- Knowledge of problem domain can help choose the best combination of operators



Examples of Evolutionary Algorithms

Genetic Algorithms (GA) - Holland, 1975
Binary genotypes, crossover and mutation

Genetic Programming (GP) - Koza, 1992
Tree-based genotypes, crossover and mutations

Steady-State GA (SSGA) – Whitley et al., 1988
Gradual replacement: Best individuals replace replace worst individuals

Differential Evolution (DE) – Storn & Prince, 1996
As SSGA, but with differential factor

Evolutionary Strategies (ES) - Rechenberg, 1973
Real-valued genotypes, mutation step(s) encoded in genotype

Covariance Matrix Adaptation ES (CMA-ES) – Hansen & Ostermeier, 2001
Evolutionary Strategies with correlated and adaptive mutations

Non-dominated Sorting GA (NSGA)– Srinivas, Deb, 1998
Multi-objective evolutionary optimization

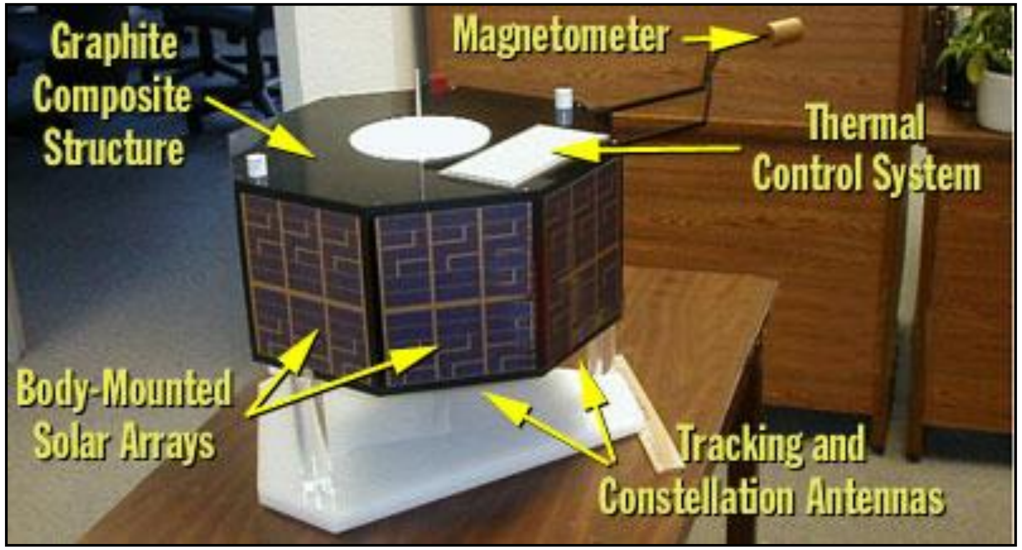
Viability Evolution (ViE)– Maesani, Mattiussi, Floreano, 2014
Evolution without fitness ranking and diversity preservation

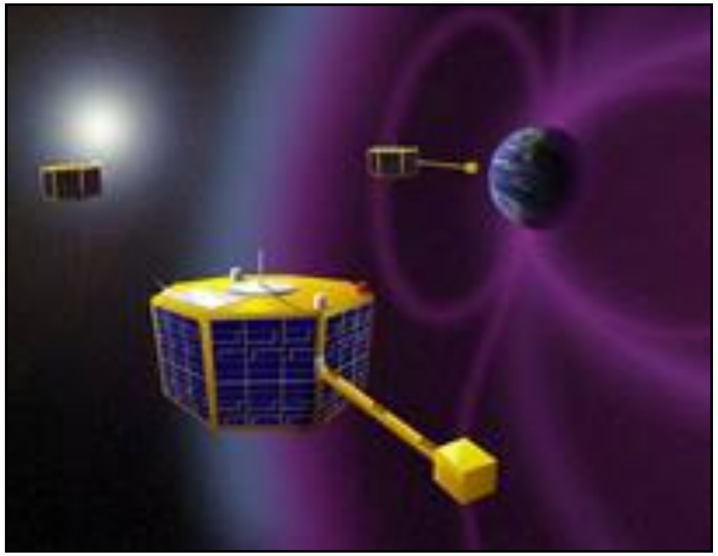
MAP Elites – Mouret and Clune, 2015
Preserve diversity by making similar solutions compete with each other

Evolutionary Design of Nanosatellite Antenna

ST5 mission: Measure effect of solar activity on the Earth's magnetosphere
3 nanosatellites (50 cm)
Problem: Design an antenna for sending data to ground station

[Lohn, Hornby, Linden, 2004]

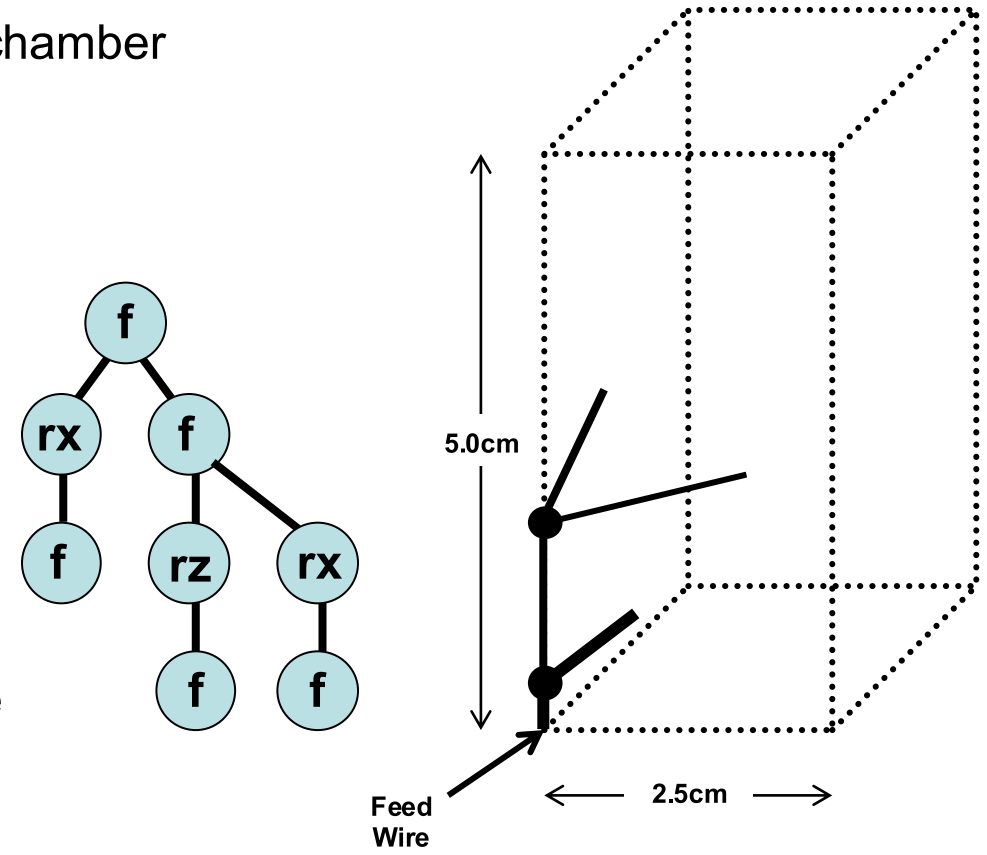


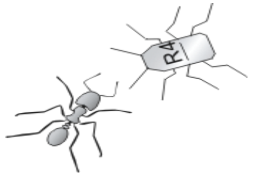


Genetic Programming of Antenna Fabrication

Tree-based Encoding with Genetic Programming
Evaluate fitness in simulation
Build best and test in anechoic chamber

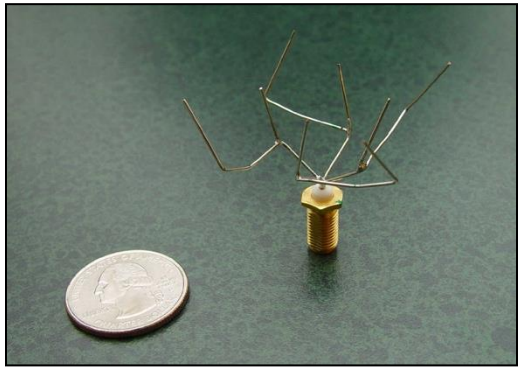
Function Set
f = forward (length, radius)
r x/y/z = rotate x/y/z
Terminal Set
Length, radius, x, y, z
Constraint:
max. 3 branches for each f node





Comparison human/evolved

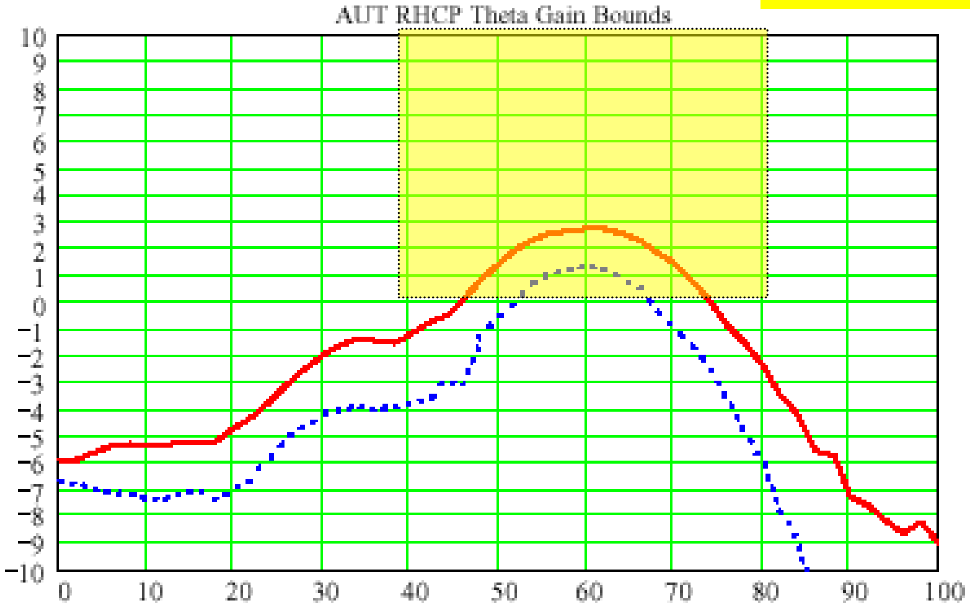
Human designed



Evolved

usable band

Signal Gain min and max



Angular difference between axis of satellite and of Earth station

Checkpoints

- What is a discrete representation?
- Into what phenotypes a binary genotype can be mapped?
- Describe how to represent a real value with a binary representation
- Describe how to represent a job schedule with a binary representation
- Describe how to represent a sequence with a discrete representation
- What are real-valued representations and when may be used?
- Describe tree-based representations
- Describe the methods to create the initial population
- Describe proportionate selection
- When does proportionate selection fail and why?
- Describe rank-based selection
- Describe truncated rank-based selection
- Describe tournament selection
- What is elitism?
- Describe types of artificial crossover for different representations
- Describe types of artificial mutations for different representations
- Is there an evolutionary algorithm that is best on any type of problem?
- How could we genetically encode an antenna design?