Introduction to computational intelligence

Basics of evolutionary computation

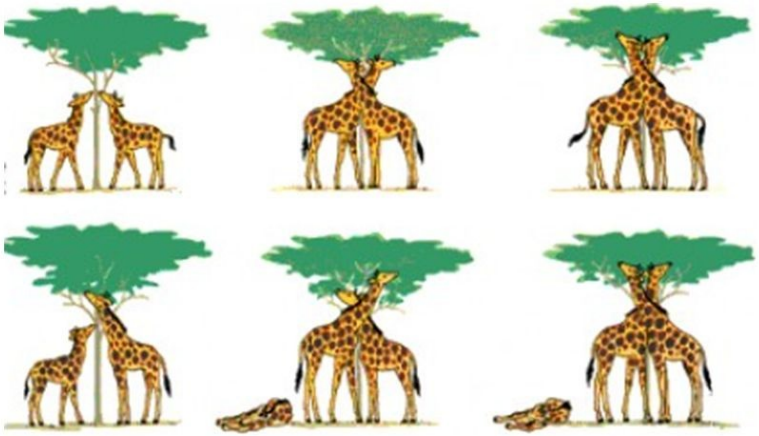
Igor Farkaš
Department of Applied Informatics
Comenius University in Bratislava

Engelbrecht (2007), John Wiley & Sons, 2nd ed.

Basic concepts

- Evolutionary computation (EC) – "core" part of CI
- Evolution = optimization process to improve the ability of an organism to survive in dynamically changing and competitive environments.
- Defined in various domains, we focus on biological evolution
- Different views:
  - Lamarckian view (1744–1829) – inheritance of acquired traits
  - Darwinian view (1809–1882) – theory of natural selection

Lamarckism vs Darwinism



Darwin's theory of natural selection

- In a world with limited resources and stable populations, each individual competes with others for survival.

  → Those individuals with the "best" characteristics (traits) are more likely to survive and to reproduce, and those characteristics will be passed on to their offspring.

- During production of a child organism, random events cause random changes to the child organism's characteristics.

  → If these new characteristics are a benefit to the organism, then the chances of survival for that organism are increased.

Generic Evolutionary Algorithm

Let $t = 0$ be the generation counter;

Create and initialize a population $C(t)$ of individuals; [encoding of solutions]

while stopping condition(s) not true do

Evaluate the fitness, $f(x_i(t))$, of each individual $x_i(t)$; [fitness function]

Perform reproduction to create offspring; [reproduction operators]

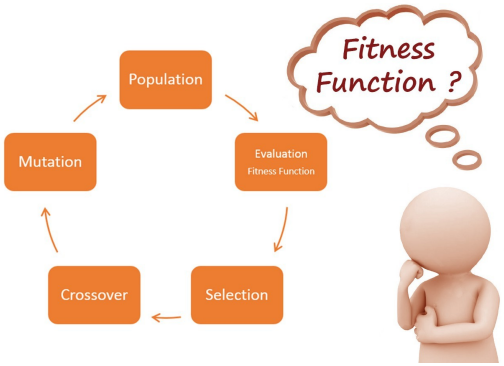
Select the new population, $C(t+1)$; [selection operators]

Advance to the new generation, i.e. $t \leftarrow t + 1$;

end

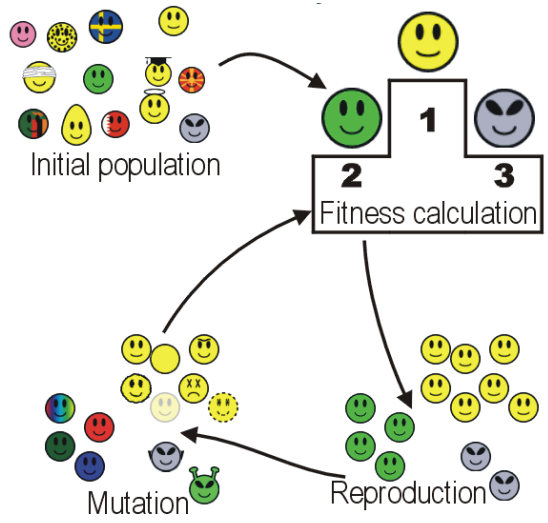
How to design a fitness function?

- FF should be clearly defined. The reader should be able to clearly understand how the fitness score is calculated.
- FF should be implemented efficiently. If it becomes the bottleneck of the algorithm, then the overall efficiency of the GA will be reduced.
- FF should quantitatively measure how well the given solution fits the problem.
- FF should generate intuitive results. The best/worst candidates should have best/worst score values.



https://towardsdatascience.com/how-to-define-a-fitness-function-in-a-genetic-algorithm-be572b9ea3b4

Evolutionary algorithm



Various EC paradigms

- Genetic algorithms – which model genetic evolution.
- Genetic programming – is based on genetic algorithms, but individuals are programs (represented as trees).
- Evolutionary programming – is derived from the simulation of adaptive behavior in evolution (i.e. phenotypic evolution).
- Cultural evolution – models the evolution of culture of a population and how the culture influences the genetic and phenotypic evolution of individuals.
- Neuroevolution – genomes represent artificial neural networks by describing structure and connection weights.
- Evolutionary learning – e.g. Learning classifier system

Representation – Chromosome

- Chromosome – represents characteristics of an individual, these are of two types (with complex mutual relationship):
- Genotype – describes the genetic composition of an individual
- Phenotype – the expressed behavioral traits of an individual in a specific environment.
- Most EAs represent solutions as vectors of a specific data type.
- Continuous search space problem is mapped into a discrete programming problem.

Operators

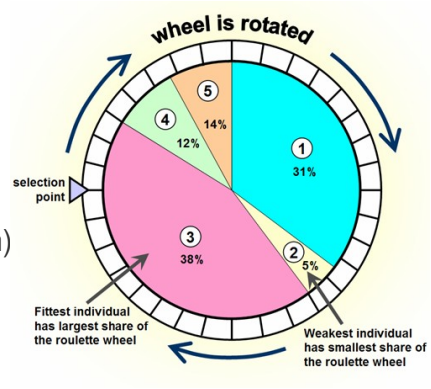
- Selection
  - Of the new population
  - reproduction
- Selective pressure (if high, decreases diversity, and vice versa)
  - random, proportional, elitism
  - ...
- Reproduction: crossover + mutation
- Fitness function: $f: G \rightarrow \mathbb{R}$ (G – chromosome repr.)

Selection operators

- Random: $P(x_i) = 1/N$
- Proportional (to fitness function): $P(x_i) = \frac{f(x_i(t))}{\sum_j f(x_j(t))}$
- Boltzmann selection: (T = temperature)

$$P(x_i) = \frac{\exp(f(x_i(t))/T(t))}{\sum_j \exp(f(x_j(t))/T(t))}$$

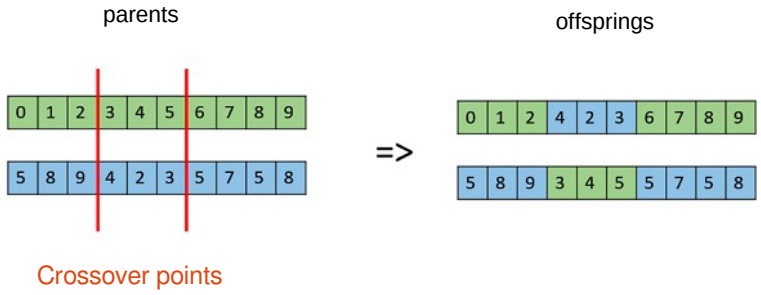
- Elitism – best candidates are copied to next generation (without mutation)



https://stackoverflow.com/questions/23183862/genetic-programming-difference-between-roulette-rank-and-tournament-selection

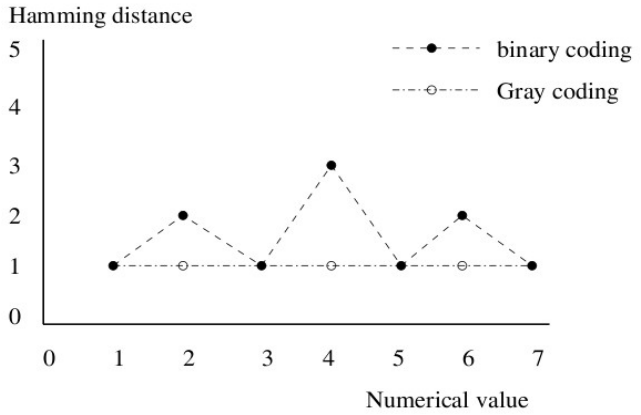
Crossover in a genetic algorithm

- can help find better configurations
- Offsprings are added to the population, two individuals with lowest fitness function are removed



Binary versus Gray encoding

	Binary	*Gray*
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100



Gray code removes undesirable 'discontinuity' of a binary code

Stopping conditions

- when no improvement is observed over a number of consecutive generations
- when there is no change in the population
- when an acceptable solution has been found
- when the objective function slope is approximately zero

EC versus classical optimization (CO)

- CO algorithms – very successful (and more efficient than EAs) in linear, quadratic, strongly convex, unimodal and other specialized problems
- EAs – more efficient for discontinuous, non-differentiable, multimodal and noisy problems.
- Differences between the two:
  - in search process
  - in information about search space used to guide the search process

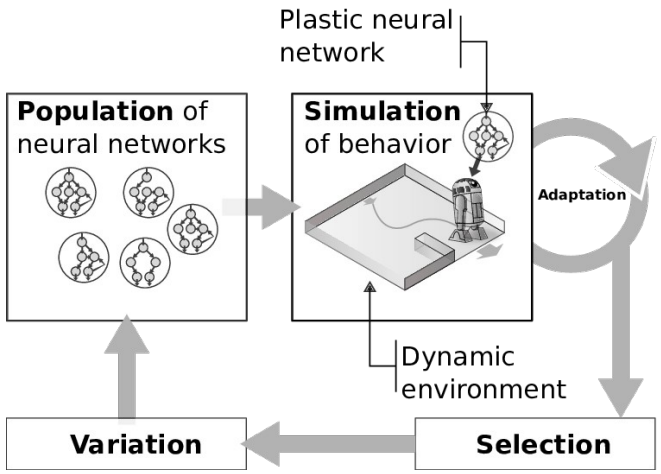
EC versus CO (ctd)

- The search process:
  - CO uses deterministic rules to move from one point in the search space to the next point.
  - CO applies a sequential search – starting from one point
  - EC uses probabilistic transition rules
  - EC applies parallel search - starts from a diverse set of initial points
- Search surface information:
  - CO uses derivative information (1st or 2nd order, of the search space)
  - EC uses no derivative information but fitness values of individuals

EC for learning

- Evolutionary learning can be used in supervised, unsupervised and reinforcement learning.
- Learning classifier systems (rule-based systems)
- Evolutionary artificial neural networks
- Evolutionary fuzzy logic systems
- Co-evolutionary learning
- Automatic modularisation of machine learning systems by specialization and niching.

Neuroevolution

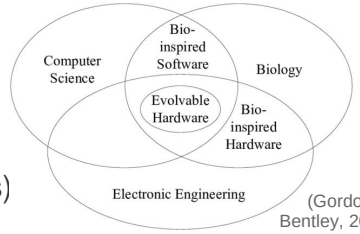


Mouret J.-P. et al. Artificial Evolution of Plastic Neural Networks: A Few Key Concepts. Studies in Computational Intelligence, 557, 2015

EC for design

EC techniques are particularly good at exploring unconventional designs which are very difficult to obtain by hand.

- Evolutionary design of artificial NNs
- Evol. design of electronic circuits
- Evolvable hardware
- Evol. design of buildings (architectures)



(Gordon & Bentley, 2002)

https://bioinspiredarchitecture.wordpress.com/

Summary

- Evolutionary computation = computer-based problem solving systems that use computational models of evolutionary processes, such as natural selection, survival of the fittest and reproduction, as the fundamental components of such systems.
- Problems are treated as optimization tasks.
- Basically, anything can be evolved (subject to formalization).
- Important components: problem encoding, fitness function, and operator used.