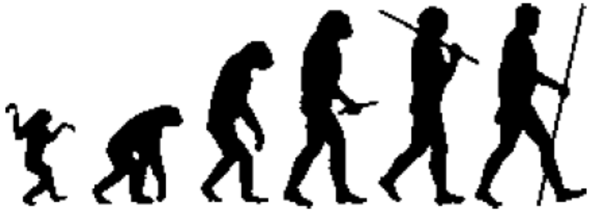
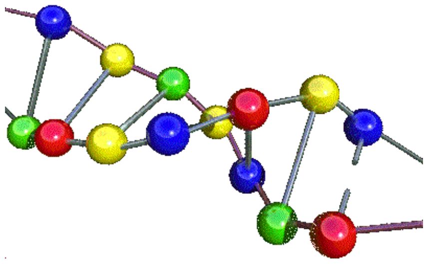
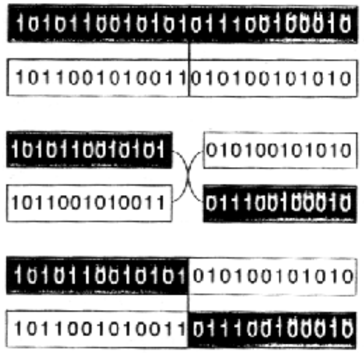
Artificial Intelligence

Genetic algorithms







Slides by Carlos Grilo

Genetic algorithms

- Genetic algorithms are parallel and stochastic search algorithms

- They were created in the 70s of the 20th century by John Holland

- They are based on the principles of natural selection and genetics

Natural selection

- According to Darwin, nature promotes the most well adapted individuals along the years leading them to reproduce more often than others
- But, one species subject only to the natural selection mechanism would tend to converge to a homogeneous population composed by the most well adapted individual of the initial population

Genetics

- This doesn't happen, however, due to the existence of operators that act at the genetic material level:
  - The recombination operator, which exchanges genetic material between two chromossomes
  - The mutation operator, which introduces new genetic material leading to population diversity

Concepts and terminology – part I

- Chromossomes consist in DNA (deoxyribonucleic acid) chains constituted by genes that codify the characteristics of the individuals
- The different values that genes may have are named aleles
- The genome corresponds to all genetic material of the individual
- Genotype is the set of genes contained in the genome

Concepts and terminology – part I

- Chromossomes are organic entities that, with the development that occurs during the life of an individual, code for his/her/its fenotype

- The fenotype corresponds to the observable characteristics of an individual

  - For example, the color of the eyes and the size of the nose are part of the fenotype

Concepts and terminology – part II

- Genetic algorithms act upon a population of individuals
- Each individual represents a potencial solution to the problem we want to solve
- Each generation (iteration), a new set of individuals is created through the application of recombination, mutation or other operators
- The probability that an individual is selected depends on its quality as a solution to the problem, computed with a fitness function

In order to use a GA…

- It is necessary to represent the problem according to the concepts already mentioned:
  - Population (namely, its size)
  - Individual
  - Chromossome
  - Gene and alele

Genet(r)ic Algorithm

```
t = 0
create_initial_population(P(t))
evaluate(P(t))
while stop condition not met
        P'(t) = select(P(t))
        P''(t) = apply_genetic_operators(P'(t))
        P(t + 1) = create_next_population(P(t), P''(t)))
        evaluate(P(t + 1))
        t = t + 1
return best individual found
```

Genet(r)ic Algorithm

```
t = 0
create_initial_population(P(t))
evaluate(P(t))
while stop condition not met
    P'(t) = select(P(t))
    P''(t) = apply_genetic_operators(P'(t))
    P(t + 1) = create_next_population(P(t), P''(t)))
    evaluate(P(t + 1))
    t = t + 1
return best individual found
```

Initial population generation

- The first step of the algorithm consists in creating the initial population, that is, the first set of candidate solutions (individuals)

- In general, this is done randomly

- However, we can use knowledge about the domain to create better initial individuals

Genet(r)ic Algorithm

```
t = 0
create_initial_population(P(t))
evaluate(P(t))
while stop condition not met
    P'(t) = select(P(t))
    P''(t) = apply_genetic_operators(P'(t))
    P(t + 1) = create_next_population(P(t), P''(t)))
    evaluate(P(t + 1))
    t = t + 1
return best individual found
```

Evaluation – part I

- The second step consists in evaluating the individuals quality using a predefined quality criterium
- In general, this criterium takes the form of a function designated as fitness function, that computes, for each individual, a numeric value reflecting its quality as a solution to the problem

Genet(r)ic Algorithm

```
t = 0
create_initial_population(P(t))
evaluate(P(t))
while stop condition not met
    P'(t) = select(P(t))
    P''(t) = apply_genetic_operators(P'(t))
    P(t + 1) = create_next_population(P(t), P''(t)))
    evaluate(P(t + 1))
    t = t + 1
return best individual found
```

Stop condition

- Common stop criteria:
  - A predefined number of generations (iterations) has been reached → the most common criterium
  - No significative changes occur in the population during some generations
  - The existence of an individual in the population that is a solution to the problem

Genet(r)ic Algorithm

```
t = 0
create_initial_population(P(t))
evaluate(P(t))
while stop condition not met
    P'(t) = select(P(t))
    P''(t) = apply_genetic_operators(P'(t))
    P(t + 1) = create_next_population(P(t), P''(t)))
    evaluate(P(t + 1))
    t = t + 1
return best individual found
```

Selection

- In each iteration, the first step consists in stochasticaly selecting the best individuals of the population

- From the application of this step, a temporary population P'(*t*) is created

- There are several selection methods that may be applied, all of them obeying to the following principles:
  - The best indivuals have more chances of being selected
  - The selection is done with reposition: it is possible to choose the same individual more than once, so that the best individuals can be chosen more often

Genet(r)ic Algorithm

```
t = 0
create_initial_population(P(t))
evaluate(P(t))
while stop condition not met
    P'(t) = select(P(t))
    P''(t) = apply_genetic_operators(P'(t))
    P(t + 1) = create_next_population(P(t), P''(t)))
    evaluate(P(t + 1))
    t = t + 1
return best individual found
```

Genetic operators application

- Genetic operators are applied on the individuals of population P'(t)
- These operators manipulate the individuals' genes so that different individuals are produced, allowing other areas of the search space to be explored
- Genetic operators are applied hoping that better individuals are produced

Genetic operators application

- A given occurrence probability is associated to each operator
- It may happen that some individuals are not subject to any changes, thus passing to the next population with no modifications
- A temporary population P''(t) results from the application of these operators

Genet(r)ic Algorithm

```
t = 0
create_initial_population(P(t))
evaluate(P(t))
while stop condition not met
    P'(t) = select(P(t))
    P''(t) = apply_genetic_operators(P'(t))
    P(t + 1) = create_next_population(P(t), P''(t))
    evaluate(P(t + 1))
    t = t + 1
return best individual found
```

Next population generation

- There are basically two strategies to create the next population:
  - Generational strategy, which consists in replacing the current population by P’’(t)
  - Stable state strategy, in which the new population is created replacing only a small set of individuals form the current population, usually the worst ones, by individuals of P’’(t)
- In any case, the new population should have the same size N as the previous one

Genet(r)ic Algorithm

```
t = 0
create_initial_population(P(t))
evaluate(P(t))
while stop condition not met
    P'(t) = select(P(t))
    P''(t) = apply_genetic_operators(P'(t))
    P(t + 1) = create_next_population(P(t), P''(t)))
    evaluate(P(t + 1))
    t = t + 1
return best individual found
```

Evaluation part II

- The new individuals are evaluated in the last step of each iteration
- We say that the algorithm converged when almost all individuals are composed by the same genetic material
- It is hoped that, during the evolutionary process, an optimal or near optimal individual is generated

Individuals representation

- The original algorithm used binary sequences of 1’s and 0’s of fixed size
- We can use integer, real numbers or other representations
- The sequences should be represented so that each symbol has a precise meaning, thus establishing a mapping between each individual and the search space

Selection methods

- As we have seen, after the evaluation process, it is necessary to decide which individuals will be allowed to produce descendants for the next generation and in what proportion

- The selection method should allow the best individuals to be chosen more often, in hope that their descendants are even better, allowing the population to evolve until a (good) solution is found

Selection methods

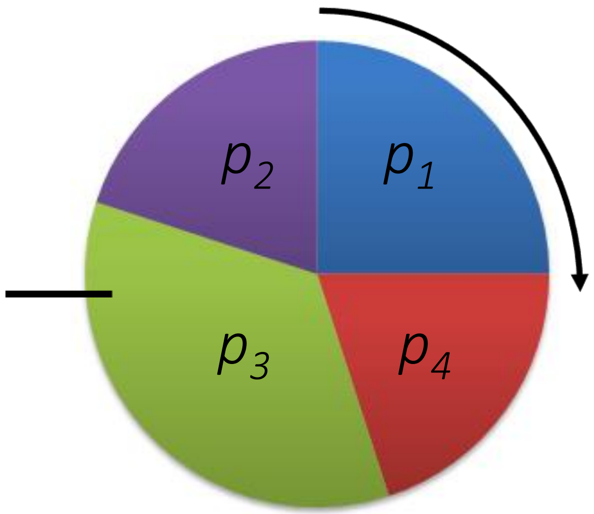
- Fitness proportional selection
  - Roulette wheel
  - Universal stochastic sampling
- Rank selection
- Truncation selection
- Tournament selection

Roulette wheel

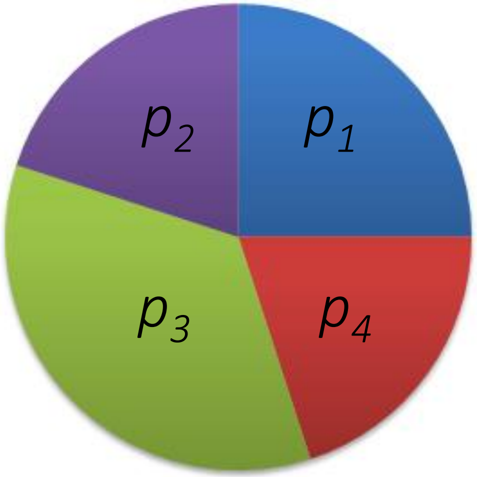
- The probability that an individual is chosen is

$$p_i = \frac{f_i}{\sum_{j}^{N} f_j}$$

where N is the size of the population and f_i is the quality of individual i computed with the fitness function



Roulette wheel

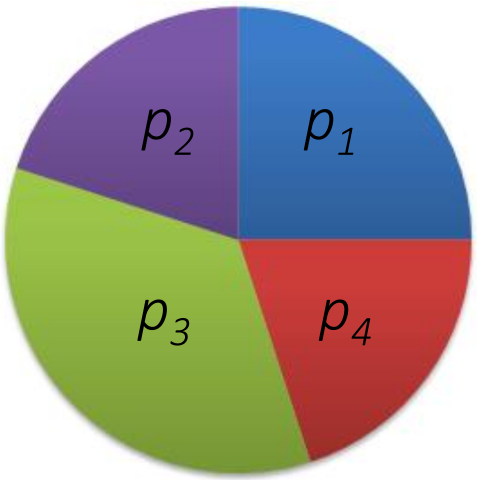


Individual	Fitness	p_i	Accumulated probabilities
1	5	0.25	0.25
2	4	0.2	0.45
3	7	0.35	0.80
4	4	0.2	1
Sum	20	1	

- In order to select the *N* individuals of P'(t), first, the accumulated probabilities are computed
- Then, the following procedure is performed *N* times:
  - Generate a random number in the [0, 1[interval
  - Select the individual whose accumulated value is right above the generated number

Roulette wheel

Individual	Fitness	p_i	Accumulated probabilities
1	5	0.25	0.25
2	4	0.2	0.45
3	7	0.35	0.80
4	4	0.2	1
Sum	20	1	



- Example:
  - Generate 0.94 -> select individual 4
  - Generate 0.62 -> select individual 3
  - Generate 0.12 -> select individual 1
  - Generate 0.46 -> select individual 3

Stochastic universal sampling

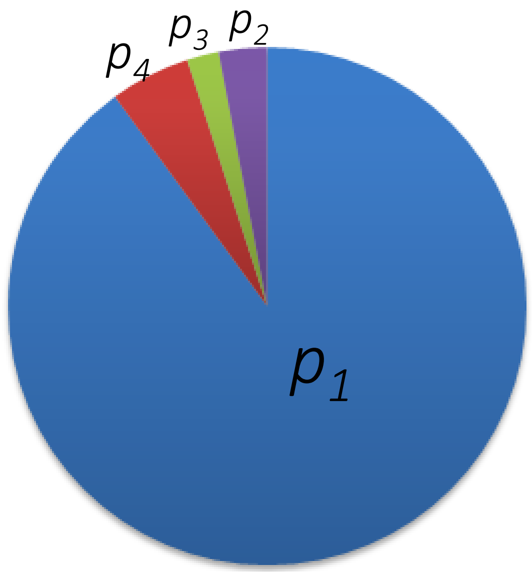
- *N* equally spaced pointers are placed at once in the roulette
- Thus, each individual is selected the number of times corresponding to the number of pointers pointing to its slice

Roulette wheel and SUS: problems

- It may happen that, in the the first generations, a small set of (bad) individuals takes over the population because they are much better than the rest
- On the other side, if the standard deviation of the individuals' fitness is small, the selective pressure can be insufficient
- Selective pressure: tendency to select the best individuals

Roulette wheel – problem 1

Individual	Fitness	p_i
1	90	0.9
2	5	0.05
3	2	0.02
4	3	0.03
Sum	100	1

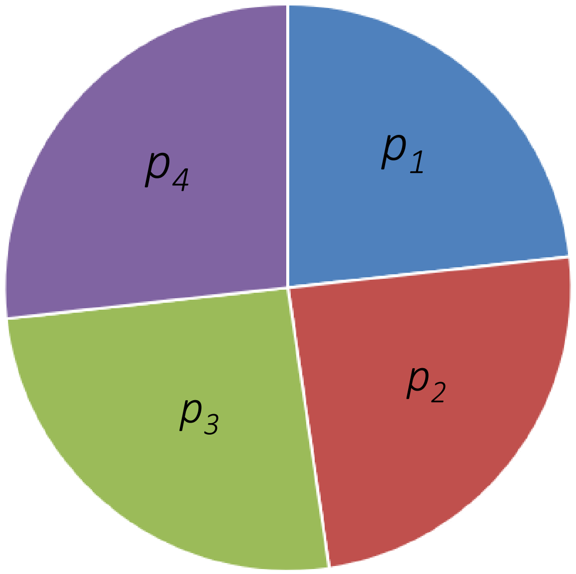


The probability that individuals 2, 3 or 4 are chosen is very low

There is a high probability that the next generation is composed only by descendants of individual 1

Roulette wheel – problem 2

Individual	Fitness	p_i
1	20	0,233
2	21	0,244
3	22	0,256
4	23	0,267
Sum	86	1



The individuals are very similar regarding fitness

However, with the roulette wheel method, we don't have a way of amplifying their differences

Premature convergence

- These problems, mainly the first one, can lead to premature convergence of the population
- Premature convergence: (quick) estabilization of the population in a set of individuals far from the optimal solution

Rank based selection

1. Rank individuals according to their fitness, the best individual in position *N* and the worst in position 1

2. Forget fitness values!…

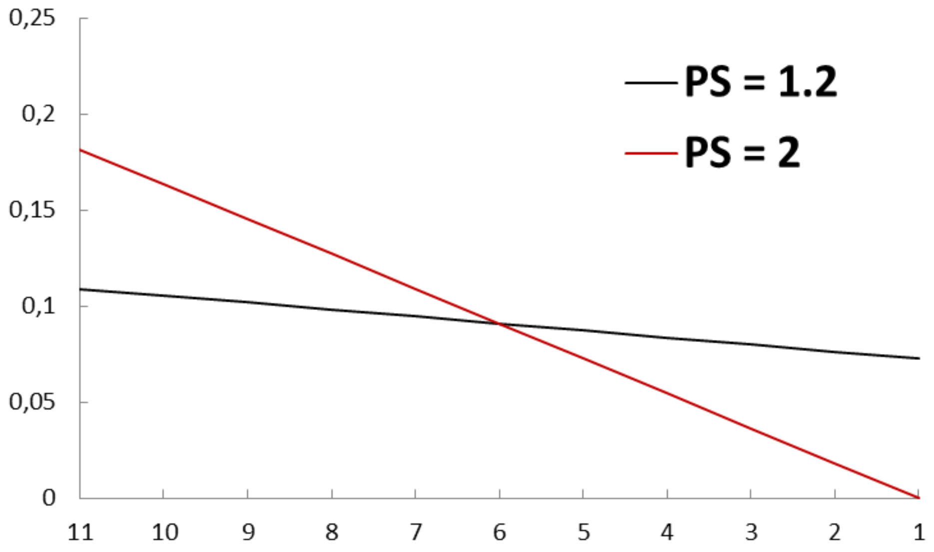
3. Set selection probability using a linear, exponential or other distribution according to the position of each individual in the rank so that all probabilities sum to 1

Example: linear distribution

The probability that individual *i* is selected is given by

$$p_i = \frac{1}{N}\left[2 - PS + 2(PS - 1)\frac{i-1}{N-1}\right]$$

where *PS* ∈ [1, 2] represents the selective pressure



Rank based selection

- It tries to reduce the quick convergence to a local maximum

- Since selection is based on the rank and not on the relative fitness values of the individuals, it is possible to avoid that a small number of individuals, much better than the rest, takes over in subsequent generations

- On the other side, if the fitnesses' standard deviation is low, an adequate selection pressure can be kept

Truncation selection

- Description:
  - Individuals are ordered according to their fitnesses
  - Only a fraction *F* of the best individuals can be selected; these individuals have the same probability of being selected
- Seletive pressure decreases as the value of *F* grows
- This method limits the destructive power of genetic operators
- However, it can also lead to premature convergence

Tournament selection

- The following procedure is repeated N times:
  - Randomly choose T (the tournament size) individuals and select the best of them
- Often, $T = 2$, but other values may be used
- The higher T, the higher the selective pressure

Tournament selection

- Advantage: it is computationaly efficient since it does not need a centralized comparison of all the individuals in order to order them by fitness value as in the rank selection method
- This allows to considerably speed the evolutionary process and, besides, it allows to easily parallelize the algorithm

Elitism

- Selection methods don’t guarantee that the best individual is chosen
- Often, GAs users grant that the best individual is chosen at least once (it is deterministically copied once to P’(t))
- This is called elitism
- It doesn’t guarantee better results…

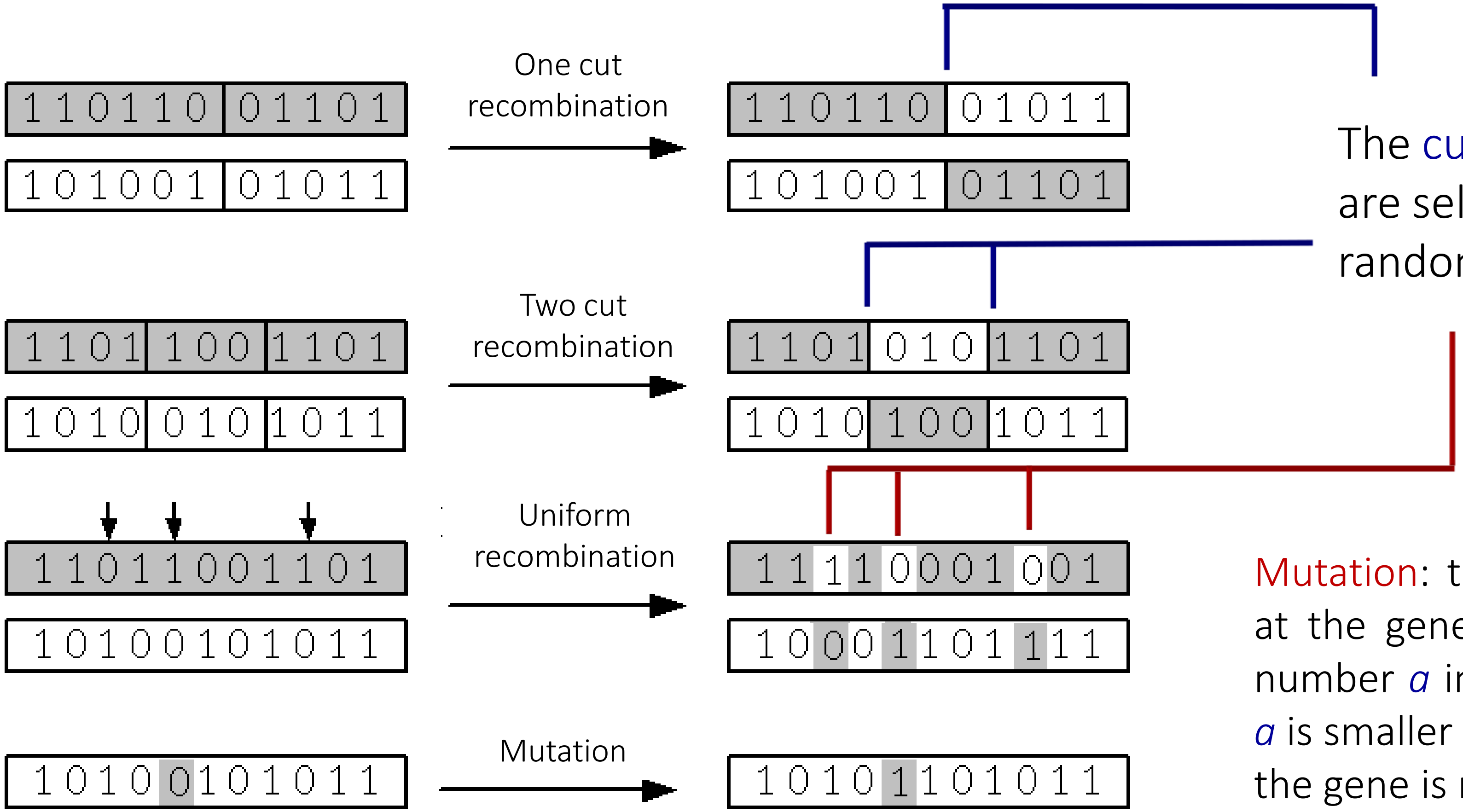
Genetic operators

- The most common operators are:
  - The recombination operator, which allows to exchange parts of two individuals, creating two others
  - The mutation operator, which modifies just one individual
- Tipically, the mutation operator is applied after the recombination operator

Genetic operators

- The probability of applying the recombination operator is usually large
- The mutation operator is usually applied with low probability because, if not, the next population can present few characteristics in common with the current population, which turns the search process close to a random one

Genetic operators



The cut points are selected randomly

Mutation: the mutation probability is defined at the genes level: for each gene, a random number *a* in the interval [0, 1[is generated; if *a* is smaller than the mutation probability, then the gene is mutated

Mutation for real valued individuals

- Let us consider individuals represented as vectors of real numbers of size *n*

$$[x_1, \dots, x_n]$$

- In these algorithms, and specially in the Evolution Strategies algorithm, it is common to use the following mutation operator

$$x_i = x_i + \delta \times N(0,1)$$

- δ is the standard deviation of the Gussian function $N(0,1)$

Mutation for real valued individuals

- There can be a δ for each gene x_i; in this case individuals are represented by two vectors

$$[x_1, \dots, x_n] \text{ and } [\delta_1, \dots, \delta_n]$$

- The vector $[\delta_1, \dots, \delta_n]$ can also be subject to mutation, for example:

$$\delta_i = \delta_i \times \exp(\tau' \times N(0,1) + \tau \times N_j(0,1))$$

where, usually $\tau = (\sqrt{2\sqrt{n}})^{-1}$ and $\tau' = (\sqrt{2n})^{-1}$

- $N_j(0,1)$ indicates that the random number is generated anew for each value of j (j refers to the individual)

Evolutionary algorithms applications

- **Optimization**
  - Numeric optimization
  - Circuit design
  - Scheduling
- **Automatic programming**
  - Programs for specific tasks
  - Cellular automata

Evolutionary algorithms applications

- Social systems
  - Study of the evolution of social systems (insects)
  - Evolution of cooperation and communication in multi-agent systems
- Biology
  - Relation between individual learning and the evolution of species

Bibliography

- Artificial Intelligence: A Modern Approach

  Stuart Russell & PeterNorvig, Prentice Hall

- Introduction to Genetic Algorithms

  Melanie Mitchell, The MIT Press

- Genetic Algorithms + Data Structures = Evolution Programs

  Zbigniew Miclalewicz, Springer