# Evolutionary Computing
# الأحتساب التطوري

أعداد
الأستاذ الدكتور
زياد طارق الطائي

المرحلة الرابعة
علوم الحاسبات

# References :

1. "Genetic Algorithms in Search, Optimization, and Machine Learning", by David E. Goldberg, (1989), Addison-Wesley, Chapter 1-8, page 1- 432.

2. "An Introduction to Genetic Algorithms", by Melanie Mitchell, (1998), MIT Press, Chapter 1-6, page 1- 203,

3. "Genetic Algorithms: Concepts And Designs", by K. F. Man, K. S. and Tang, ***S.*** Kwong, (201), Springer, Chapter 1- 10, page 1- 348,

4. "Genetic algorithms and engineering design", by Mitsuo Gen, and Runwei Cheng, (1997), John Wiley & Sons Inc, chapter 1- 10, page 1-411.

5. 'Practical genetic algorithms", by Randy L. Haupt, (2004), John Wiley ***S Sons*** Inc, Chapter 1- 7, page 1- 251.

# Evolutionary Computing:

## Evolution:

is a long time scale process that changes a population of organism by generating better offsprings trough reproduction.

## Introduction to Evolutionary computing:

In computer science is a subfield of artificial intelligence (more particularly computational intelligence) that can be defined by the type of algorithms it is concerned with. These algorithms, called evolutionary algorithms, are based on adopting Darwinian principles, hence the name. Technically they belong to the family of trial and error problem solvers and can be considered global optimization methods with a metaheuristic or stochastic optimization character, distinguished by the use of a population of candidate solutions (rather than just iterating over one point in the search space). They are mostly applied for black box problems (no derivatives known), often in the context of expensive optimization.

Evolutionary computation uses iterative progress, such as growth or development in a population. This population is then selected in a guided random search using parallel processing to achieve the desired end. Such processes are often inspired by biological mechanisms of evolution.

As evolution can produce highly optimized processes and networks, it has many applications in computer science.

## Evolutionary Computing History:

The use of Darwinian principles for automated problem solving originated in the 1950s. It was not until the 1960s that three distinct interpretations of this idea started to be developed in three different places.

Evolutionary programming was introduced by Lawrence J. Fogel in the US, while John Henry Holland called his method a genetic algorithm. In Germany Ingo Rechenberg and Hans-Paul Schwefel introduced evolution strategies. These areas developed separately for about 15 years. From the early nineties on they are unified as different representatives ("dialects") of one technology, called evolutionary computing. Also in the early nineties, a fourth stream following the general ideas had emerged – genetic programming. Since the 1990s, nature-inspired algorithms are becoming an increasingly significant part of evolutionary computation.

These terminologies denote the field of evolutionary computing and consider evolutionary programming, evolution strategies, genetic algorithms, and genetic programming as sub-areas.

Simulations of evolution using evolutionary algorithms and artificial life started with the work of Nils Aall Barricelli in the 1960s, and was extended by Alex Fraser, who published a series of papers on simulation of artificial selection. Artificial evolution became a widely recognised optimization method as a result of the work of Ingo Rechenberg in the 1960s and early 1970s, who used evolution strategies to solve complex engineering problems. Genetic algorithms in particular became popular through the writing of John Holland. As academic interest grew, dramatic increases in the power of computers allowed practical applications, including the automatic evolution of computer programs.

Evolutionary algorithms are now used to solve multi-dimensional problems more efficiently than software produced by human designers, and also to optimize the design of systems.

**Evolutionary Computing Techniques:**

Evolutionary computing techniques mostly involve metaheuristic optimization algorithms. Broadly speaking, the field includes:

- Ant colony optimization
- Artificial Bee Colony Algorithm
- Artificial immune systems
- Artificial life (also see digital organism)
- Bees algorithm
- Cultural algorithms
- Differential evolution
- Dual-phase evolution
- Evolutionary algorithms
- Evolutionary programming
- Evolution strategy
- Gene expression programming
- Genetic algorithm
- Genetic programming
- Harmony search

- Learnable Evolution Model
- Learning classifier systems
- Particle swarm optimization
- Self-organization such as self-organizing maps, competitive learning
- Swarm intelligence

**Search Optimization Algorithms:**

Fig. below shows different types of Search Optimization algorithms.

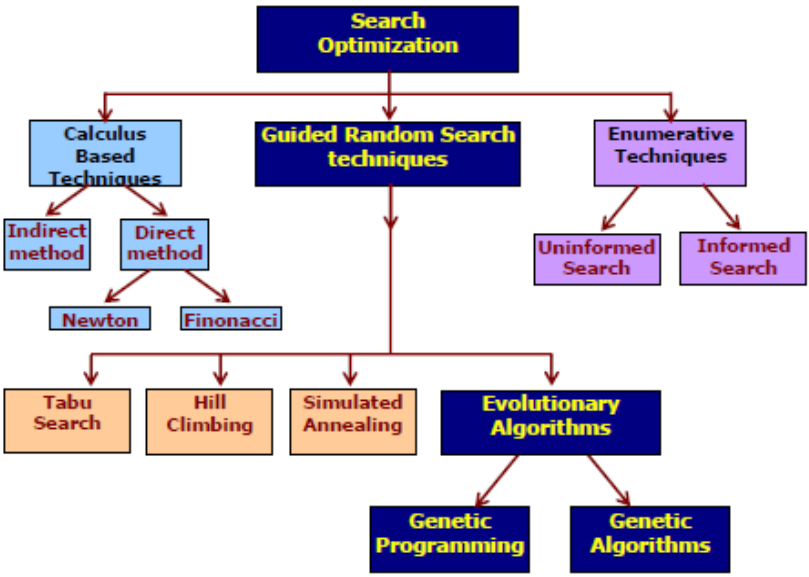


Fig. Taxonomy of Search Optimization techniques

- We are interested in evolutionary search algorithms. The Evolutionary Algorithms include
- Genetic Algorithms and
- Genetic Programming

**Evolutionary algorithms:**

Evolutionary algorithms form a subset of evolutionary computation in that they generally only involve techniques implementing mechanisms inspired by biological evolution such as reproduction, mutation, recombination, natural selection and survival of the fittest. Candidate solutions to the optimization problem play the role of individuals in a population, and the cost function determines the environment within which the solutions "live" (see also fitness function). Evolution of the population then takes place after the repeated application of the above operators.

In this process, there are two main forces that form the basis of evolutionary systems: **Recombination** and **mutation** create the necessary diversity and thereby facilitate novelty, while **selection** acts as a force increasing quality.

Many aspects of such an evolutionary process are stochastic. Changed pieces of information due to recombination and mutation are randomly chosen. On the other hand, selection operators can be either deterministic, or stochastic. In the latter case, individuals with a higher fitness have a higher chance to be selected than individuals with a lower fitness, but typically even the weak individuals have a chance to become a parent or to survive.

Evolutionary Computation (EC) is a general term for several computational techniques. Evolutionary Computation represents

powerful search and optimization paradigm influenced by biological mechanisms of evolution : that of natural selection and genetic.

Evolutionary Algorithms (EAs) refers to Evolutionary Computational
models using randomness and genetic inspired operations.

EAs involve selection, recombination, random variation and competition of the individuals in a population of adequately represented potential solutions. The candidate solutions are referred as chromosomes or individuals.

Genetic Algorithms (GAs) represent the main paradigm of Evolutionary Computation, because of:
- GAs simulate natural evolution, mimicking processes the nature uses : Selection, Crosses over, Mutation and Accepting.
- GAs simulate the survival of the fittest among individuals over consecutive generation for solving a problem.

## Fundamentals of Genetic Algorithms

### What are GAs ?

- Genetic Algorithms (GAs) are adaptive heuristic search algorithm based on the evolutionary ideas of natural selection and genetics.

- Genetic algorithms (GAs) are a part of Evolutionary computing, a rapidly growing area of artificial intelligence. GAs are inspired by Darwin's theory about evolution - "survival of the fittest".

- GAs represent an intelligent exploitation of a random search used to solve optimization problems.

## Introduction to Genetic Algorithms:

Solving problems mean looking for solutions, which is best among others.

Finding the solution to a problem is often thought :

- In computer science and AI, as a process of search through the space of possible solutions. The set of possible solutions defines the search space (also called state space) for a given problem. Solutions or partial solutions are viewed as points in the search space.
- In engineering and mathematics, as a process of optimization. The problems are first formulated as mathematical models expressed in terms of functions and then to find a solution, discover the parameters that optimize the model or the function components that provide optimal system performance.

## Why Genetic Algorithms ?

- It is better than conventional AI ; It is more robust.
- Unlike older AI systems, the GA's do not break easily even if the inputs changed slightly, or in the presence of reasonable noise.
- While performing search in large state-space, or multi-modal state-space, or n-dimensional surface, a genetic algorithms offer significant benefits over many other typical search optimization techniques like - linear programming, heuristic, depth-first, breath-first.

"Genetic Algorithms are good at taking large, potentially huge search spaces and navigating them, looking for optimal combinations of things, the solutions one might not otherwise find in a lifetime.

## Optimization:

Optimization is a process that finds a best, or optimal, solution for a problem. The Optimization problems are centered around three factors :

1.*An objective function* which is to be minimized or maximized;
Examples

* In manufacturing, we want to maximize the profit or minimize the cost.
* In designing an automobile panel, we want to maximize the strength.

2.*A set of unknowns or variables* that affect the objective function,
Examples

* In manufacturing, the variables are amount of resources used or the time spent.
* In panel design problem, the variables are shape and dimensions of the panel.

3.*A set of constraints* that allow the unknowns to take on certain
values but exclude others;
Examples

* In manufacturing, one constrain is, that all "time" variables to be non-negative.
* In the panel design, we want to limit the weight and put constrain on its shape.

An optimization problem is defined as : Finding values of the variables that minimize or maximize the objective function while satisfying the constraints.

**Genetic Algorithms (GAs) – Basic Concepts:**

Genetic algorithms (GAs) are the main paradigm of evolutionary computing. GAs are inspired by Darwin's theory about evolution - the 'surviva of the fittest". In nature, competition among individuals for scanty resources results in the fittest individuals dominating over the weaker ones.

- GAs are the ways of solving problems by mimicking processes nature uses; ie., Selection, Crosses over. Mutation and Accepting, to evolve a solution to a problem.
- GAs are adaptive heuristic search based on the evolutionary ideas of natural selection and genetics.
- GAs are intelligent exploitation of random search used in optimization problems.
- GAs, although randomized, exploit historical information to direct the search into the region of better performance within the search space.

**Biological Background - Basic Genetics:**

* Every organism has a set of rules, describing how that organism is built. All living organisms consist of cells.

* In each cell there is same set of chromosomes. Chromosomes are strings of DNA and serve as a model for the whole organism.

* A chromosome consists of genes, blocks of DNA.

* Each gene encodes a particular protein that represents a trait (feature), e.g., color of eyes.

* Possible settings for a trait (e.g. blue, brown) are called alleles.

* Each gene has its own position in the chromosome called its locus.

* Complete set of genetic material (all chromosomes) is called a genome.

* Particular set of genes in a genome is called genotype.

* The physical expression of the genotype (the organism itself after birth) is called the phenotype, its physical and mental characteristics, such as eye color, intelligence etc.

* When two organisms mate they share their genes; the resultant offspring may end up having half the genes from one parent and half from the other. This process is called recombination (cross over) .

* The new created offspring can then be mutated. Mutation means, that the elements of DNA are a bit changed. This changes are mainly caused by errors in copying genes from parents.

* The fitness of an organism is measured by success of the organism in its life (survival).

Below shown, the general scheme of evolutionary process in genetic along with pseudo-code.

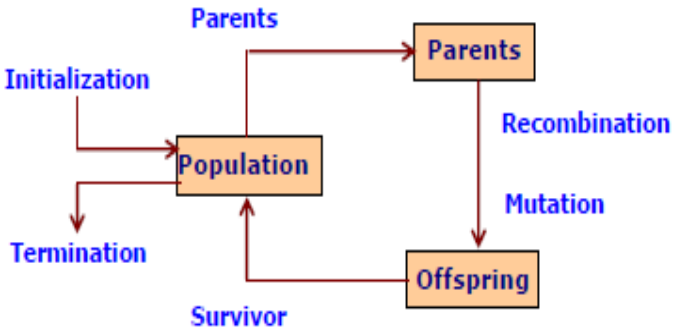


Fig. General Scheme of Evolutionary process

## Search Space:

In solving problems, some solution will be the best among others. The space of all feasible solutions (among which the desired solution resides) is called **search space** (also called state space).

- Each point in the search space represents one **possible solution.**
- Each possible solution can be "marked" by its value (or **fitness**) for the problem.
- The GA looks for the **best solution** among a number of possible solutions represented by one point in the search space.
- Looking for a solution is then equal to looking for some extreme value (minimum or maximum) in the search space.
- At times the search space may be well defined, but usually only a few points in the search space are known.

In using GA, the process of finding solutions generates other points (possible solutions) as evolution proceeds.

Genetic Algorithms Disadvantages:

1. No guarantee for optimal solution within a finite time
2. Weak theoretical basis
3. Interdependency of genes
4. Parameter tuning is an issue
5. Often computationally expensive, i.e. slow

Genetic Algorithms Advantages:

1. A robust search technique
2. No (little) knowledge (assumption) the problem space
3. Fairly simple to develop: low development costs
4. Easy to incorporate with other methods
5. Solutions are interpretable
6. Can be run interactively, i.e. accommodate user preference
7. Provide many alternative solutions
8. Acceptable performance at acceptable costs on a wide range of problems
9. Intrinsic parallelism (robustness, fault tolerance)

**Working Principles:**

Before getting into GAs, it is necessary to explain few terms.

- Chromosome : a set of genes; a chromosome contains the solution in form of genes.
- Gene : a part of chromosome; a gene contains a part of solution. It determines the solution, e.g. 16743 is a chromosome and 1, 6, 7, 4 and 3 are its genes.
- Individual : same as chromosome.
- Population: number of individuals present with same length of chromosome.

- Fitness : the value assigned to an individual based on how far or close a individual is from the solution; greater the fitness value better the solution it contains.
- Fitness function : a function that assigns fitness value to the individual. It is problem specific.
- Breeding : taking two fit individuals and then intermingling there chromosome to create new two individuals.
- Mutation : changing a random gene in an individual.
- Selection : selecting individuals for creating the next generation.

Genetic algorithm begins with a set of solutions (represented by chromosomes) called the population.

- Solutions from one population are taken and used to form a new population. This is motivated by the possibility that the new population will be better than the old one.
- Solutions are selected according to their fitness to form new solutions (offspring); more suitable they are, more chances they have to reproduce.
- This is repeated until some condition (e.g. number of populations or improvement of the best solution) is satisfied.

**Outline of the Basic Genetic Algorithm:**

1. [Start] Generate random population of n chromosomes (i.e. suitable solutions for the problem).
2. [Fitness] Evaluate the fitness f(x) of each chromosome x in the population.
3. [New population] Create a new population by repeating following steps until the new population is complete.

(a)[Selection] Select two parent chromosomes from a population according to their fitness (better the fitness, bigger the chance to be selected)

(b)[Crossover] With a crossover probability, cross over the parents to form new offspring (children). If no crossover was performed, offspring is the exact copy of parents.

(c)[Mutation] With a mutation probability, mutate new offspring at each locus (position in chromosome).

(d)[Accepting] Place new offspring in the new population

4. [Replace] Use new generated population for a further run of the algorithm

5. [Test] If the end condition is satisfied, stop, and return the best solution in current population

6. [Loop] Go to step 2.

Note : The genetic algorithm's performance is largely influenced by two operators called crossover and mutation. These two operators are the most important parts of GA.

## Flow chart for Genetic Algorithm:

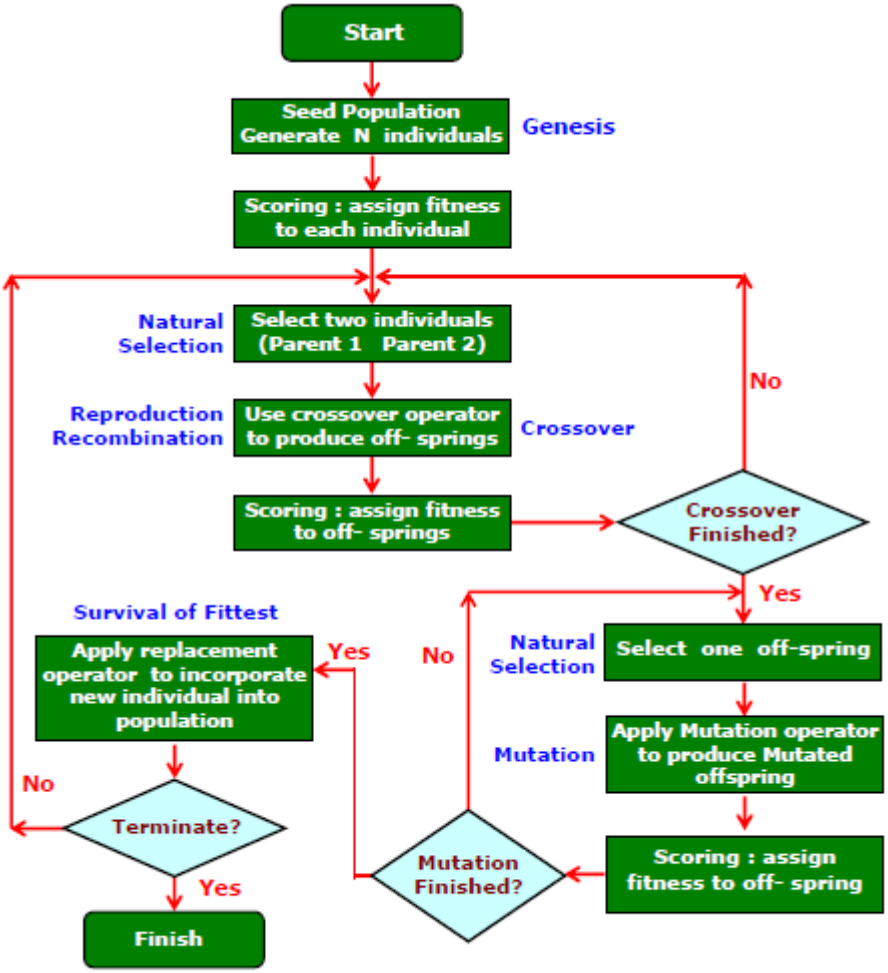


Fig. Genetic algorithm – program flow chart