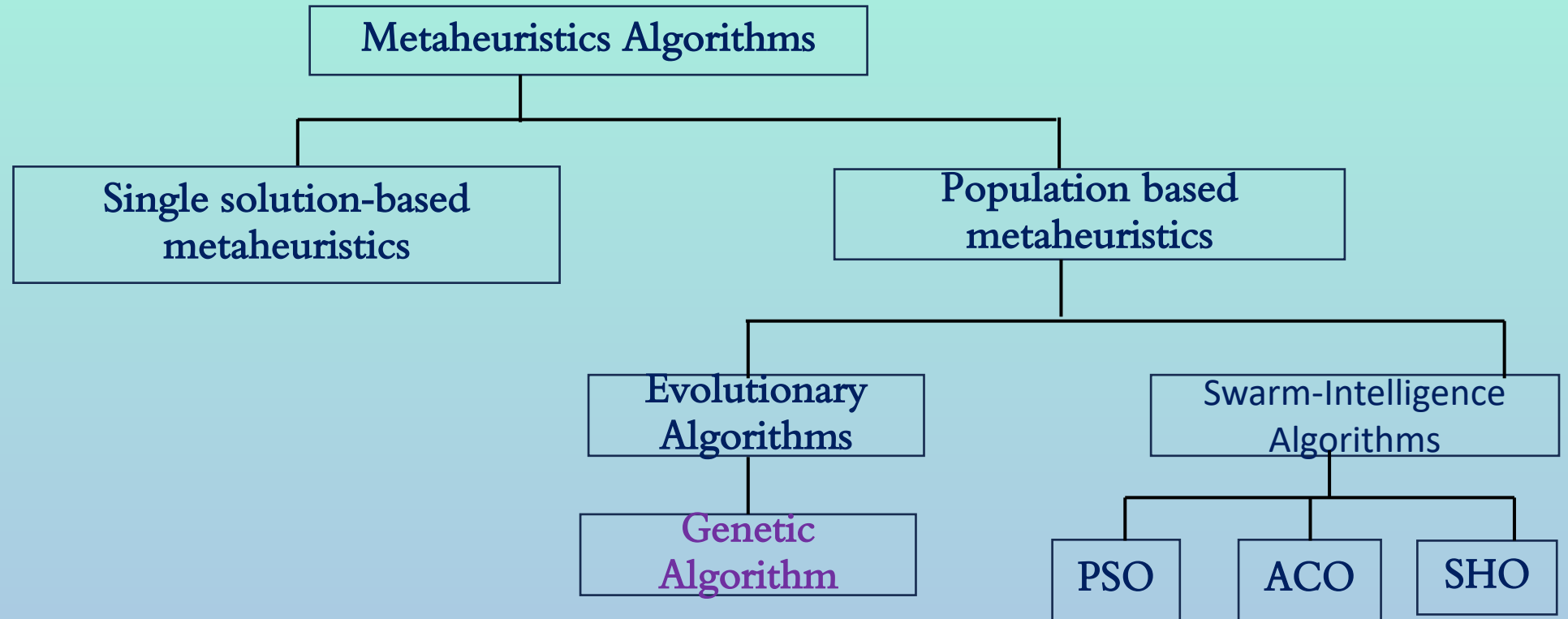


Artificial Intelligence and Machine Learning (ACCA6011)

Genetic Algorithm and Optimization

Dr. Seena Joseph

Genetic Algorithm



Particle swarm optimization; Ant colony optimization; spotted hyena optimizer;

Genetic Algorithm

Genetic Algorithm (proposed by Holland in [3]) – A well-known all metaheuristic algorithm inspired by biological evolution process.

It is a population-based search algorithm.

Can be used to generate solutions to problems that we have no way to find a solution.

It mimics the Darwinian theory – survival of fittest in nature.

Elements of GA : Chromosome representation; fitness selection; and biological inspired operators (selection, crossover, and mutation).

Genetic Algorithm

Suitable solution for the problem– **Individual**

Individual is represented as - **Chromosome**(coded as string or arrays)

Set of individuals/Chromosomes – **Population**

Each element in the chromosome – **Gene**

How well a solution solves the problem at hand/or each individual ability to solve a problem – **fitness evaluation**

Evaluate the fitness of each chromosome

Selection of individual from the solution for reproduction – **selection function**

Genetic operation that combines parts of the genetic material - **Crossover** or recombination

Crossover the parents to form a new offspring

Random changes to an organism's gene – **Mutation**

Combine information

Genetic Algorithm

Chromosome representation:

- Exampe : It takes the binary string format.
- Each position of chromosome(locus) has two possible form of genes(alleles) 0 and 1.
- Chromosomes are considered as points in the solution space.
- Processed by genetic operators.

Fitness function:

- It is used to assign a value for all chromosomes.

Biological Inspired operators:

- Selection, Mutation and Crossover.

Genetic Algorithm

Outline of the Genetic Algorithm

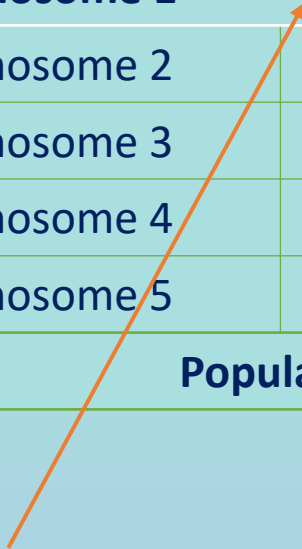
1. **[Start]** Generate random population of n chromosomes(suitable solution for the problem).
2. **[Fitness]** Evaluate the fitness, $f(x)$ of each chromosome x in the problem.
3. **[New population]** Create new population by repeating following steps until the new population is complete.
 1. **[Selection]**Select two parent chromosome from a population according to their fitness (the better fitness, the bigger chance to be selected)
 2. **[Crossover]** With a crossover probability, crossover the parents to form a new offspring (children). If no crossover was performed, offspring is an exact copy of parents.
 3. **[Mutation]** with a mutation probability mutate new offspring at each locus (position in chromosome)
 4. **[Accepting]** place new offspring in a new population
4. **[Replace]** Use new generated population for a further run of algorithm.
5. **[Test]** If the end condition is satisfied, **stop** and return the best solution in current population.
6. **[Loop]** Go to step 2

Genetic Algorithm

Population

Chromosome 1	1	1	1	1	1	1	0	1
Chromosome 2								
Chromosome 3								
Chromosome 4								
Chromosome 5								
Population of five chromosome								

Gene



Genetic Algorithm

Algorithm 1: *Classical Genetic Algorithm (GA)*

Input:

Population Size, n

Maximum number of iterations, MAX

Output:

Global best solution, Y_{bt}

begin

Generate initial population of n chromosomes Y_i ($i = 1, 2, \dots, n$)

Set iteration counter $t = 0$

Compute the fitness value of each chromosomes

while ($t < MAX$)

Select a pair of chromosomes from initial population based on fitness

Apply crossover operation on selected pair with crossover probability

Apply mutation on the offspring with mutation probability

Replace old population with newly generated population

Increment the current iteration t by 1.

end while

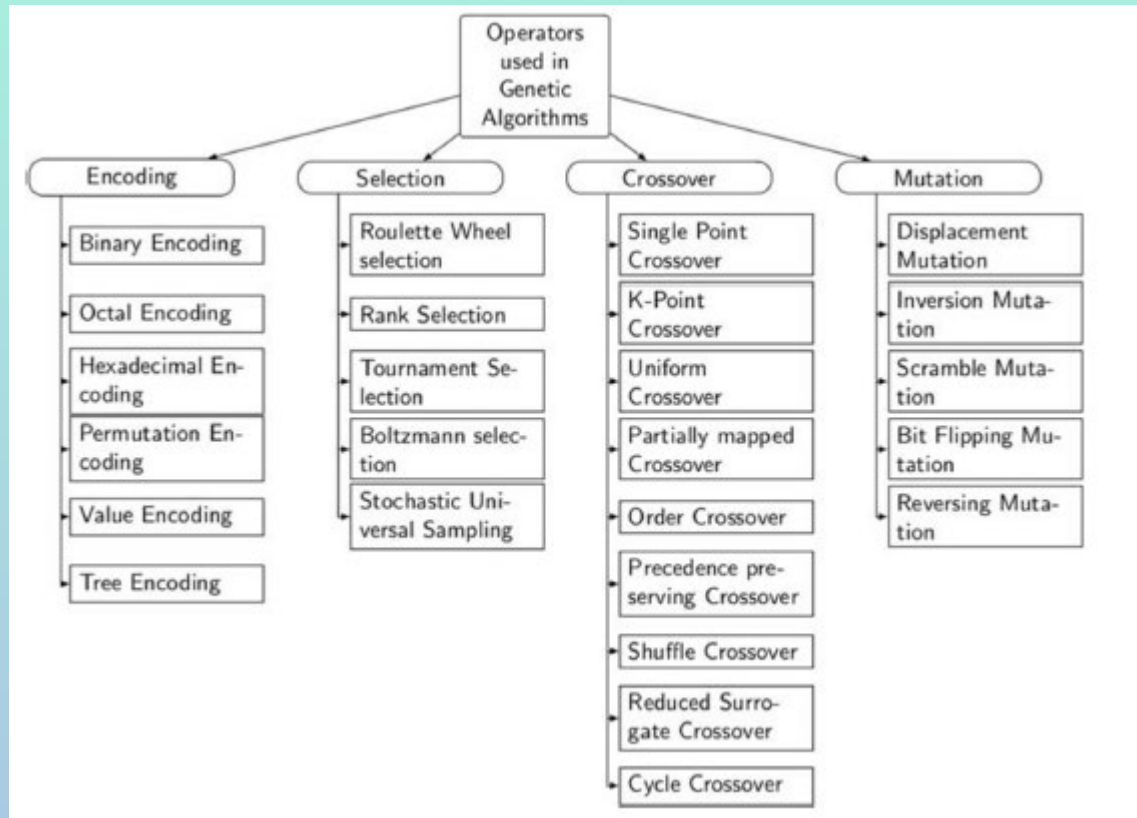
return the best solution, Y_{bt}

end

Source : [1]

Genetic Algorithm

- Operators used in GA



Source : [1]

Genetic Algorithm

Encoding Scheme	Process	Pros	Cons	Application
Binary	Each chromosome is represented as a string of 0's and 1's	Easy to implement	No support for inversion operator	For problems that support binary encoding
Octal	In the form of octal number [0-7]	Easy to implement	No support for inversion operator	Limited use
Hexadecimal	In the form of hexa decimal number [0-9], [A-F]	Easy to implement	No support for inversion operator	Limited use
Permutation	In the form of string of numbers that represents the position in a sequence	Support inversion operator	No support for binary operator	Task ordering problems
Value	In the form of string of values(real, integer, character)	No need of value conversion	Require specific cross over and mutation	Neural Network Problems
Tree	In form of a tree of functions or commands	Operator can easily apply	Difficult to design tree for some problems	Evolving Problems

Source : [1]

Genetic Algorithm

GA Process

1. Initialization
2. Fitness Evaluation
3. Selection
4. Crossover
5. Mutation
6. Replacement
7. Repeat/Termination

Genetic Algorithm

Initialization

Example 1: solution of a feature selection problem is recorded as a binary string (each string indicate whether a feature is selected or not)

Example 2: $f(x_1, x_2) = x_1^2 + x_2^2$

if x_1 and x_2 are discrete then solution can be represented as an array of integers;

If x_1, x_2 continuous, then solution can be represented as an array real values.

Other examples ?

Genetic Algorithm

Fitness Calculation/Evaluation

	X1	X2	Fitness
Chromosome 1	2.5	3.12	15.9844
Chromosome 1	1.11	0.12	1.2465
Chromosome 1	1.123	1.41	3.2492
Chromosome 1	0.01	4.71	22.1842
Chromosome 1	4.4	1.22	20.8484

Genetic Algorithm

Selection: Selection(reproduction operator) : to determine whether the particular string will participate in the reproduction process or not

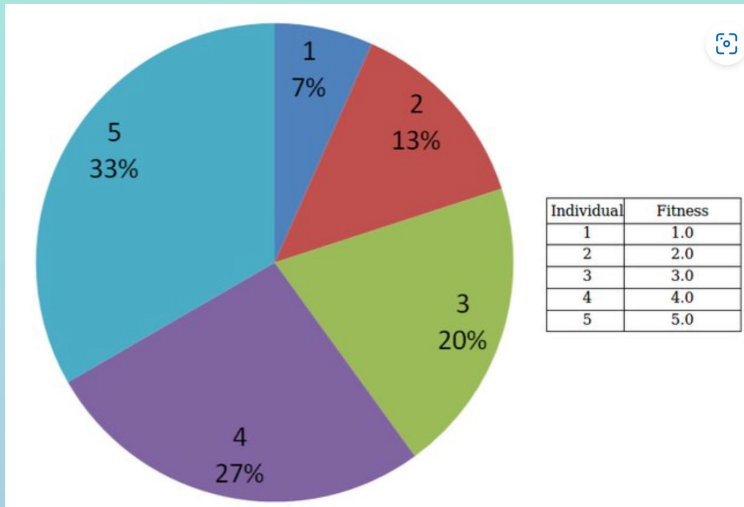
	X1	X2	Fitness	Probability	cumulative
Chromosome 1	2.5	3.12	15.9844	0.251	0.251
Chromosome 2	1.11	0.12	1.2465	0.0196	0.2706
Chromosome 3	1.123	1.41	3.2492	0.0511	0.3217
Chromosome 4	0.01	4.71	22.1842	0.350	0.6717
Chromosome 5	4.4	1.22	20.8484	0.328	1.0
		Sum	=63.5127		

A random number in the range 0 to 1 will be generated and if it lies in 0 – 0.251 then chromosome 1 will be selected for crossover, if it lies in 0.252 – 0.2761 then chromosome 2 will be selected, and so on.

Fitness proportionate selection..

Genetic Algorithm

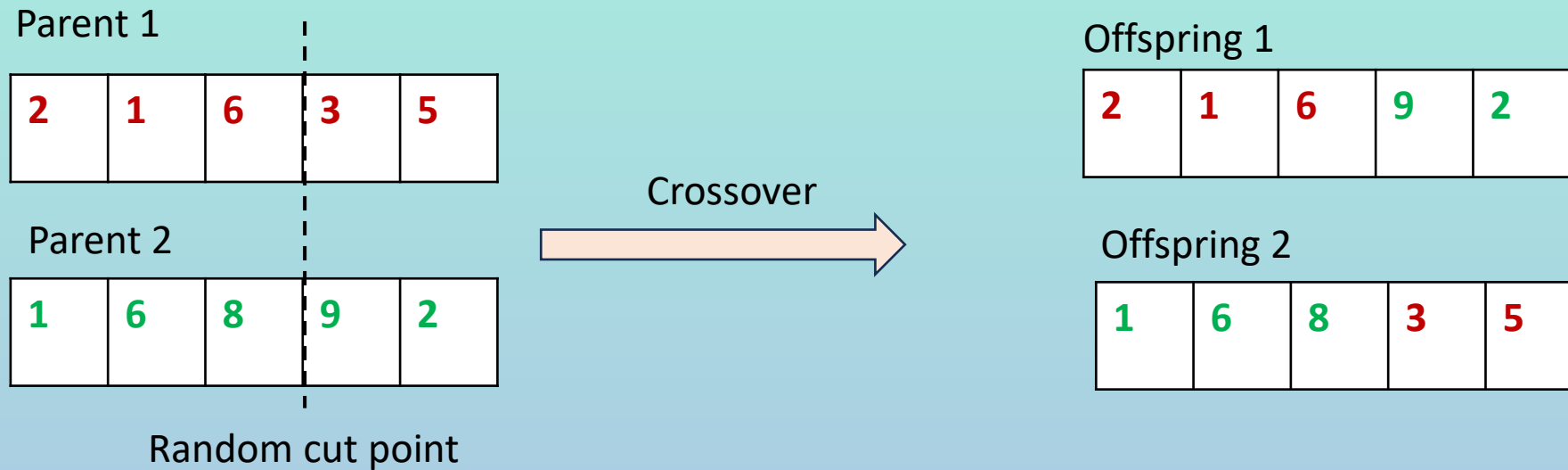
Selection techniques are: **roulette wheel**, rank, tournament, boltzmann, and stochastic universal sampling.



Genetic Algorithm

Crossover or Reproduction phase : Selects Gene from parent chromosome and create a new offspring.

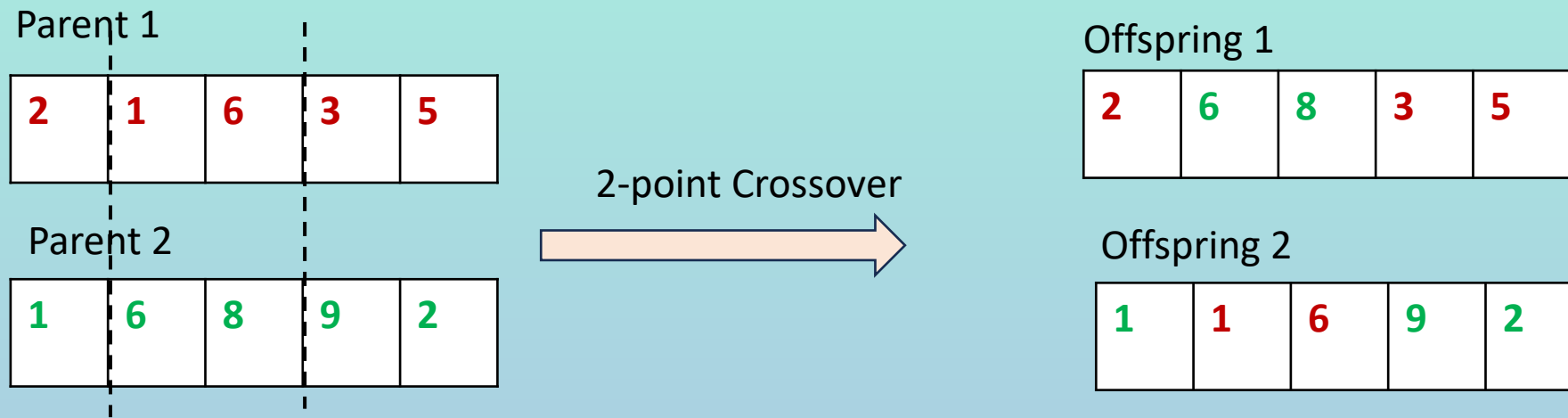
One-point cross over



Genetic Algorithm

Crossover or Reproduction phase

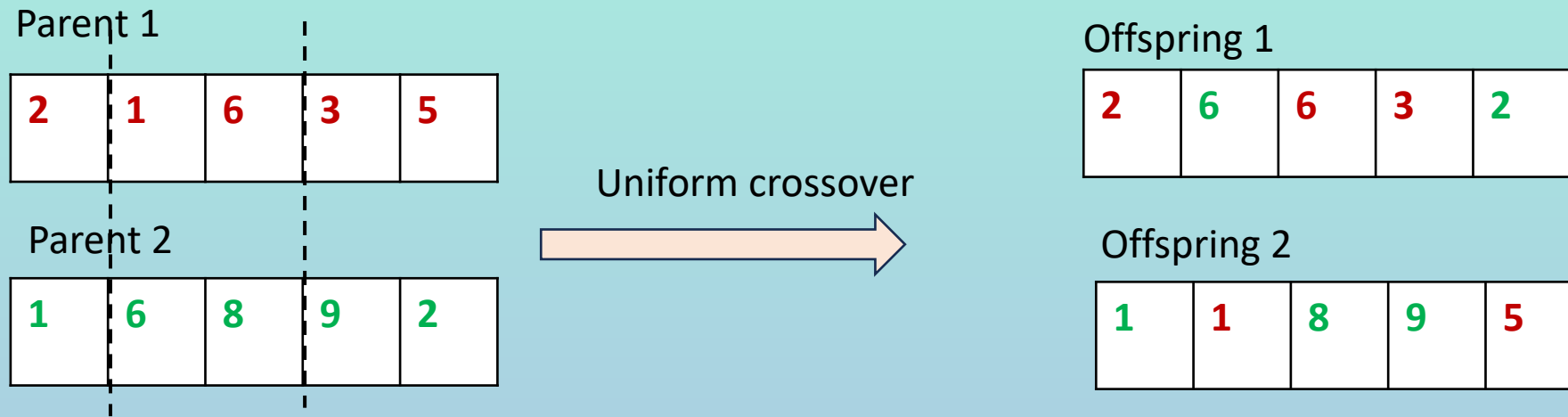
Multipoint crossover



Genetic Algorithm

Crossover or Reproduction phase

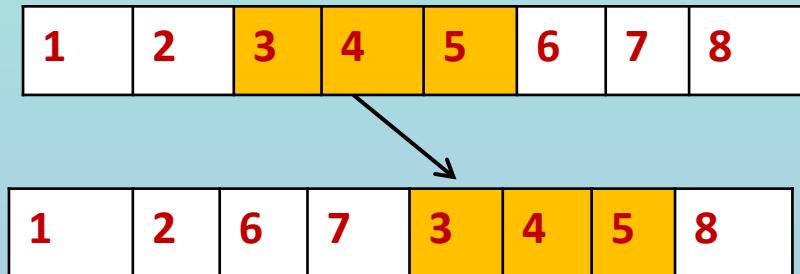
Uniform crossover : Each gene for an offspring is selected with 0.5 probability from parent 1 and 0.5 probability from parent 2



Genetic Algorithm

Mutation : Randomly change parts of some individuals to introduce genetic diversity and avoid local optima.

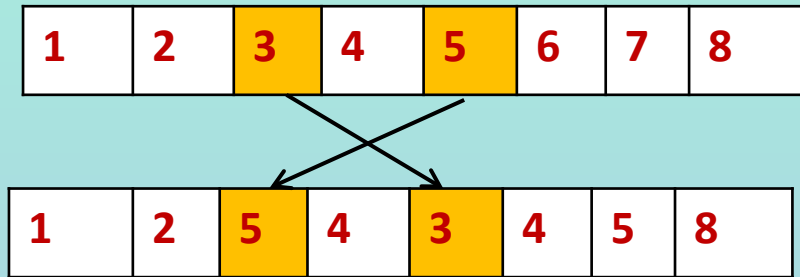
Displacement mutation



Genetic Algorithm

Mutation

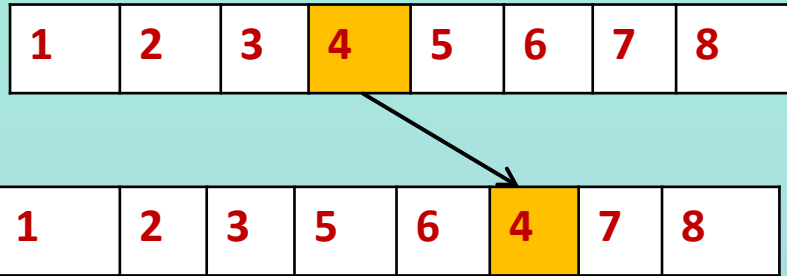
Exchange mutation



Genetic Algorithm

Mutation

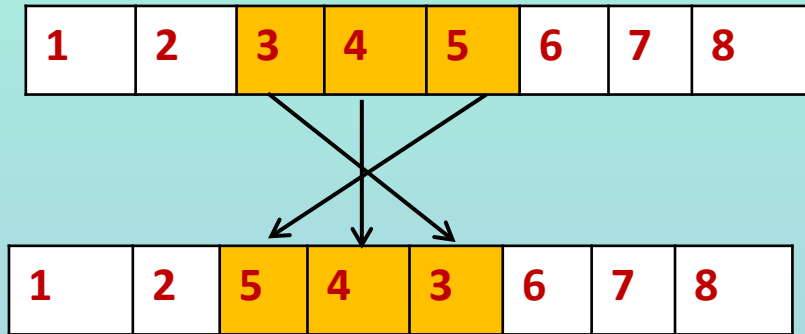
Insertion mutation



Genetic Algorithm

Mutation

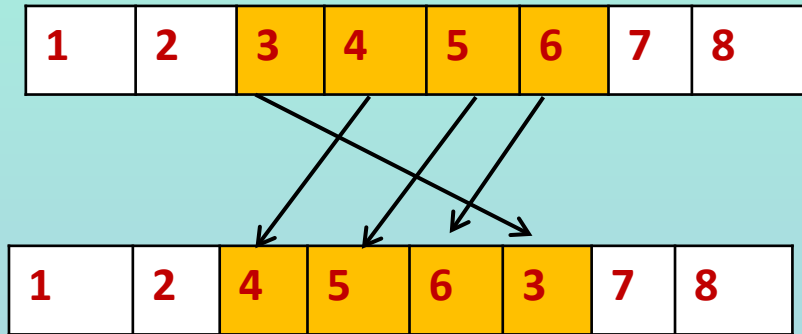
Simple inversion mutation



Genetic Algorithm

Mutation

Scramble mutation



Genetic Algorithm

Replacement : Replace some or all of the old population with the new offspring, by determining which individuals move on to the next generation.

Repeat/ Termination: The previous steps 2-6 are looped over for a set number of generations or until a termination condition is met.

Genetic Algorithm

GA Python Demo

Optimization

Optimization – to minimize or maximize the objective function to improve the model performance.

Machine learning optimization – Iteratively improving the accuracy of a model, lowering the degree of errors.

Optimization is measured using a lost or cross function – basically defining the difference between actual and predicted value of data.

Optimization

Standard form of a general optimization problem

$$\text{Minimize : } f(x)$$

$$\text{Subject to : } g_j(x) \leq 0 \quad j = 1 \text{ to } m$$

$$\quad : \quad h_k(x) = 0 \quad k = 1 \text{ to } p$$

$$\quad : \quad x_i(L) \leq x_i \leq x_i(U) \quad i = 1 \text{ to } n$$

$f(x)$ - Objective function

$g_j(x)$ - Inequality constraint

$h_k(x)$ = Equality constraint

x - represents the n design variable, that are modified to obtain the optimum.

$x_i(L)$ and $x_i(U)$ are the lower and upperbound of the design variables.

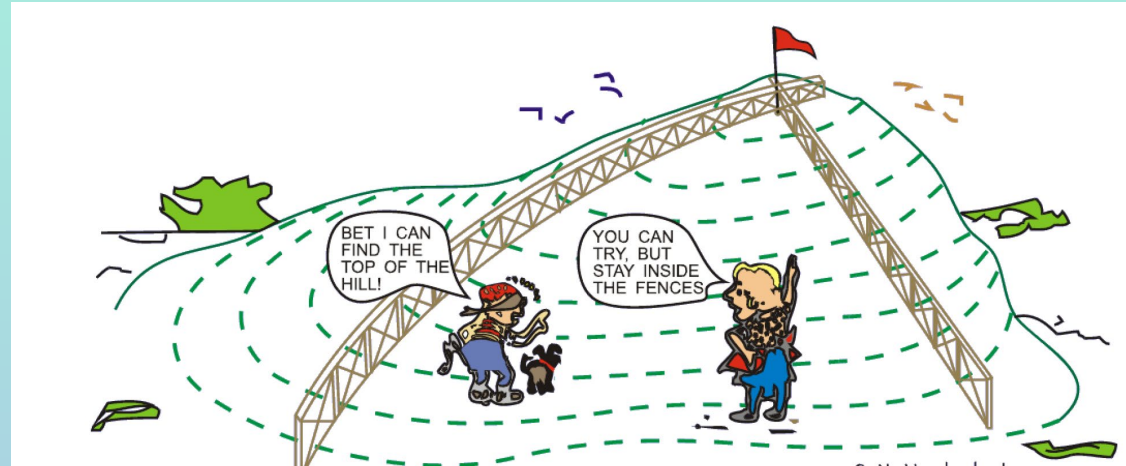
Optimization

Local and Global Optimization algorithms

- Local Optimization Algorithms – Generally **gradient based**.
- Use gradient information to find the optimum solution.
 - Two step process
 - 1. finding the gradient information for finding a search direction.
 - 2. Move in this direction until no more progress can be made

Optimization

Gradient optimization



Source 2

Optimization

Gradient optimization

- Good for solving problems
 - with large number of design variables.
 - where numerical noise is not a severe problem.
 - Where gradient information is available
 - When local minima is not an issues
- They are complex, difficult to implement, susceptible to numerical noises.

Optimization

Global optimization

Global optimization algorithms provide a much better chance of finding the global or near global optimum than the local algorithms discussed so far.

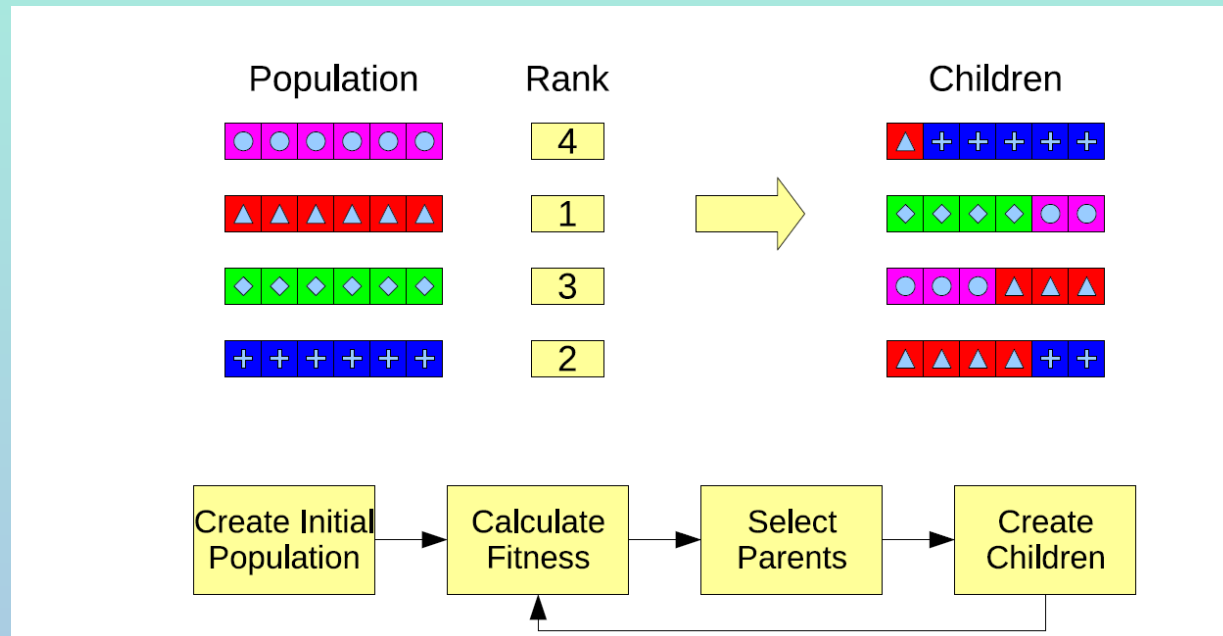
Most popular Global optimization algorithm – Evolutionary algorithms

GA, PSO

Optimization

Global optimization

Overview of a basic genetic optimization algorithm



Optimization

Global optimization Algorithm

Well suited for optimization problem

- with fewer design variables
- where the analysis is computationally inexpensive
- where the global optimization is required
- when numerical noise is severe

Extra resources

[Genetic Algorithm: Complete Guide With Python Implementation | DataCamp](#)

[Genetic Algorithms Explained By Example – Youtube video](#)

[AlgoDaily - Introduction to Genetic Algorithms in Python](#)

[How to implement genetic algorithms in Python: A Comprehensive Tutorial](#)

[Traveling Salesman Problem \(TSP\) using Genetic Algorithm \(Python\) | by Ramez Shendy | **AI monks.io** | Medium](#)

[Machine Learning Optimization - Why is it so Important? - Seldon](#)

Reference

- [1] S. Katoch, S. S. Chauhan and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimedia Tools and Applications*, vol. 80, no. 5, pp.8091, 2020.DOI: 10.1007/s11042-020-10139-6.
- [2] G. Venter, "Review of optimization techniques," 2010.
- [3] J. H. Holland, "Genetic Algorithms," *Scientific American*, vol. 267, no. 1, pp.66–73, 1992.