Team members

dept	level	id	name
cs	3	201900486	علي حسن علي هريدي
cs	3	201900465	عبدالله فتحي سيد عليوه
cs	3	201900533	عمر مختار محمد عبدالفتاح
cs	3	201900437	عبدالرحمن محمد فكري ثابت
cs	3	201900423	عبدالرحمن عصام الدين محمد

AI-Documentation

“Genetic Algorithms”

1. Project Idea in details:

- **Intro:** Optimization techniques are the techniques used to discover the best solution out of all the possible solutions available under the constraints present and use it to improve my score. The genetic algorithm is one such optimization algorithm built based on the natural evolutionary process of our nature. The idea of Natural Selection and Genetic Inheritance is used here. Unlike other algorithms, it uses guided random search, i.e.,

finding the optimal solution by starting with a random initial cost function.

- It is not the strongest of the species that survives, nor the most intelligent, but the one most responsive to change. You must be thinking what has this quote got to do with genetic algorithm? Actually, the entire concept of a genetic algorithm is based on the previous definition.

- Let's take an example to understand that where you are the president of a country and in order to keep your city safe from bad things you implement a policy like this.

- You choose all the good people, and you ask them to have their children.
- This is repeated for a few generations.
- You will notice that you now have a whole bunch of good people.

This example is not the best, but this example was just to help you understand the concept. And so the basic idea was that we changed the inputs (i.e. the population) so that we would get a better output (ie a better country).

Now we notice that the genetic algorithm is somewhat related to biology. So we will see to understand more. At first it was known that cells are the basic building block of all living things, so in every cell there is a set of chromosomes, Chromosome are basically the strings of DNA and it is consisting of genes, commonly referred as blocks of DNA, where each gene encodes a specific trait, for example hair color or eye color.

❖ What is a Genetic Algorithm?

From the previous talk we will know that the genetic algorithm depends on the genetic structure and behavior of the population chromosome. The following things are the basis of genetic algorithms.

Each chromosome indicates a possible solution. Thus the population is a set of chromosomes.

The function of fitness characterizes each individual in society. Therefore, better fitness is the solution Population, the best individuals are used to reproduce to generate the next generation. The resulting offspring will have the traits of both parents and be the result of the mutation. A mutation is a small change in the structure of a gene.

1- Steps Involved in Genetic Algorithm

1. Initialization of Population (Coding):

   To solve this problem using genetic algorithm, our first step would be defining our population. our population will contain individuals, each having their own set of chromosomes

2. Fitness Function:

   We have to select the best ones to reproduce offspring out of the available chromosomes, so each chromosome is given a fitness value.
   where The fitness score helps to select the individuals who will be used for reproduction. helps choosing individuals from the population.
3. Selection:

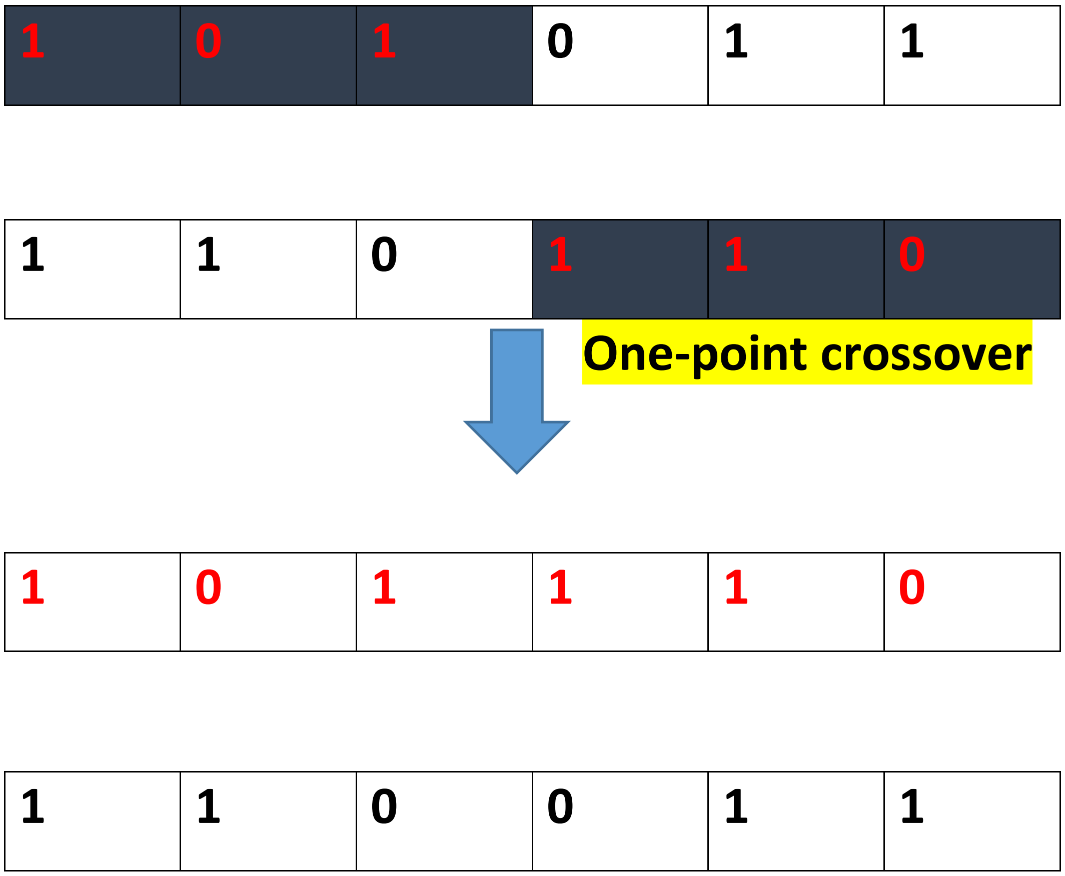
   his phase's main goal is to find the getting the best solution is more.
   So, in this method we can get both our parents in one go. This method is known as Stochastic Universal Selection method.
4. Reproduction:

   Generation of offspring happen in 2 ways:

a) Crossover:

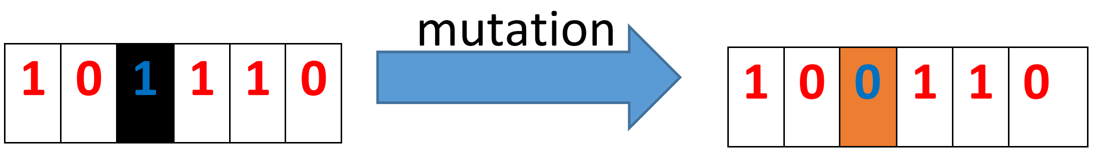
So in this previous step, we have selected parent chromosomes that will produce off-springs. So in biological terms, crossover is nothing but reproduction. defines the offspring generated

Now I have 2 chromosomes that I got from the previous step as follows:



b) Mutation:

Now if you think in a biological sense, do the children that are produced have the same traits as their parents? The answer is no. During their development, some change occurs in the genes of the children that makes them different from their parents, and this process is known as **mutation**, which can be defined as a random modification in the chromosome, which also reinforces the idea of diversity in the population. Let's see an example of that mutation



5. Convergence (when to stop):

Some of the rules that determine when GA stops are as follows:

- When an acceptable and appropriate solution is obtained.
- When there is no improvement in the quality of the generation after a certain completion
- When a hard and fast combination of generations and time is reached.

2-<u>differential evolution algorithm :</u>

is a global optimization algorithm,
It is a type of evolutionary algorithm and is related to other evolutionary algorithms such as the genetic algorithm.
Unlike the genetic algorithm, it was specifically designed to operate upon vectors of real-valued numbers instead of bitstrings. Also unlike the genetic algorithm it uses vector operations like vector subtraction and addition to navigate the search space instead of genetics-inspired transforms

The algorithm works by maintaining a population of candidate solutions represented as real-valued vectors. New candidate solutions are created by making modified versions of existing solutions that then replace a large portion of the population each iteration of the algorithm.
New candidate solutions are created using a "*strategy*" that involves selecting a base solution to which a mutation is added, and other candidate solutions from the population from which the amount and type of mutation is calculated, called a difference vector. For example, a strategy may select a best

candidate solution as the base and random solutions for the difference vector in the mutation

Mutation is calculated as the difference between pairs of candidate solutions that results in a difference vector that is then added to the base solution, weighted by a mutation factor hyperparameter set in the range [0,2].
Not all elements of a base solution are mutated. This is controlled via a recombination hyperparameter and is often set to a large value such as 80 percent, meaning most but not all variables in a base solution are replaced. The decision to keep or replace a value in a base solution is determined for each position separately by sampling a probability distribution such as a binomial or exponential.

Frist step:The function takes the name of the objective function and the bounds of each input variable as minimum arguments for the search.
It creates new candidate solutions by selecting random solutions from the population, subtracting one from the

other, and adding a scaled version of the difference to the best candidate solution in the population.

new = best + (mutation * (rand1 – rand2))

The “*popsize*” argument controls the size or number of candidate solutions that are maintained in the population. It is a factor of the number of dimensions in candidate solutions and by default, it is set to 15. That means for a 2D objective function that a population size of (2 * 15) or 30 candidate solutions will be maintained.

The total number of iterations of the algorithm is maintained by the “*maxiter*” argument and defaults to 1,000.

The “*mutation*” argument controls the number of changes made to candidate solutions each iteration. By default, this is set to 0.5. The amount of recombination is controlled via the “*recombination*” argument, which is set to 0.7 (70 percent of a given candidate solution) by default.

Finally, a local search is applied to the best candidate solution found at the end of the search. This is controlled via the "*polish*" argument, which by default is set to True. The result of the search is an OptimizeResult object where properties can be accessed like a dictionary. The success (or not) of the search can be accessed via the '*success*' or '*message*' key.

2. Main functionalities:

- Initial population
- Fitness function
- Selection
- Crossover & mutation
- End in fitting solution

3. Similar applications in the market:

- ***Robotics:***

The use of genetic algorithm in the field of robotics is quite big. Actually, genetic algorithm is being used to create learning robots which will behave as a human and will do tasks like cooking our meal, do our laundry etc.

- **Feature Selection:**

Every time you enter a data science competition, do you want to know how to choose which features are important in predicting a target variable? First you always look at the feature importance of some model, then manually decide the bottom line, and decide which features have a higher significance than that.

But is there a better way to deal with this kind of situation? Indeed, there is another way which is the genetic algorithm because it is one of the most advanced algorithms for selecting features.

We will start again with chromosome counts, since each chromosome will be a diploid chain. It will refer to the next paragraph, the model paragraph, the model paragraph.

Another difference is the change of trapping function. The fitness function here will be the measure of competition accuracy. The higher the value of our collection of flowers.

4. Initial literature review of Academic publications:

Abstract

Genetic algorithms, computer programs that simulate natural evolution, are increasingly applied across many disciplines. They have been used to solve various optimization problems from neural network architecture search to strategic games, and to model phenomena of adaptation and learning. Expertise on the qualities and drawbacks of this technique is largely scattered across the literature or former, motivating a compilation of this knowledge at the light of the most recent developments of the field. In this review, we present genetic algorithms, their qualities, limitations and challenges, as well as some future development perspectives. Genetic algorithms are capable of exploring large and complex spaces of possible solutions, to quickly locate promising elements, and provide an adequate modelling tool to describe evolutionary systems, from games to economies. They however suffer from high computation costs, difficult parameter configuration, and crucial representation of the solutions. Recent developments such as GPU computing, parallel and quantum computing, conception of powerful parameter control methods, and novel approaches in representation strategies, may be keys to overcome those limitations. This compiling review aims at informing practitioners and newcomers in the field alike in their genetic algorithm research, and at outlining promising avenues for future research. It highlights the potential for interdisciplinary research associating genetic algorithms to pulse original discoveries in social sciences, open ended evolution, artificial life and AI.

Genetics algorithm are Inspired from Darwin’s theory of evolution, the Genetic Algorithm (GA) is an adaptive search algorithm that simulates some of the evolution processes: selection, fitness, reproduction, crossover (recombination), mutation. A population of individuals evolves under selection in environment. The fittest individuals reproduce new genes, while mutations explore new individual.

A genetic algorithm is a member of the family of evolutionary algorithms, that are computational search methods inspired from natural selection. They simulate Darwinian evolution on individual entities, gathered in a population. These individual entities can be anything: digital organisms, bit strings, computer programs, financial strategies.

Entities or generations evolve to maximize their fitness to survive in an environment.

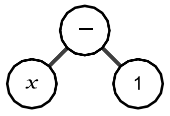
In GA's, these entities are represented with a genotype and a phenotype concept from genetics.

The representation using a genotype can be seen as encoding genetic information, a popular representation technique in GAs is a bit-sequence with binary encoding, composed of 0s and 1s, can also be stored with an alphabet encoding, These strings of bits or alphabetic.

called chromosomes and every element in this string called genes,

The different values each gene can take are called alleles. For example, in the binary encoding, each gene can have either allele 0 or 1.

(0 1 0 1 1 ...) *(A T A G C ...)*



Binary genotype Binary genotype Tree genotype

Each element of the chromosome is gene and number of genes are the length of chromosome
The phenotype corresponds to the entity characteristics: shape, size, color that are encoded in the genetic information(genotype).

For instance, genes **OCA2** and **HERC2** located on chromosome 15 in humans are considered as the main genes responsible for the eye color so the genetic information contained in genes OCA2 and HERC2 is the genotype. The resulting eye color is the phenotype.

- Fitness Function

In natural systems, evolution favors individuals that are more capable of reproducing. In GAs, fitness measures the performance of the individuals in the population which is called **Fitness Function** in GAs. Evaluating the fitness of all individuals in the current generation consists in applying our fitness criterion to them which allows to discriminate between individuals with low and high fitness score which becomes important in selection of parents to build the next generations of individuals.

-Selection

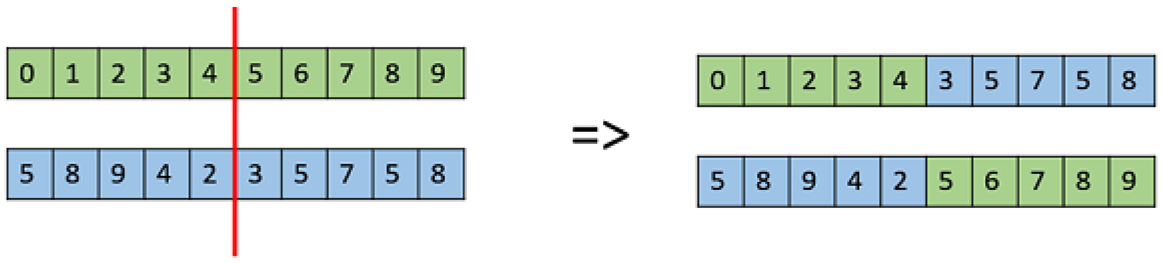
Two parents will mate, exchange genetic information, and generate two offspring that share some genetic information from their parents. The process of selection of parents continues until we have enough offspring to production of the next generation.
There are a large variety of selection methods. one of them are, The fitness" roulette-wheel" selection method. Each individual has a

probability to be selected as parent, and this probability depends on the ratio of its fitness function, so a highly fit individual could reproduce several times. Elitist selection saves the current best individual in population without modification in the next generation, in order to save the current best individual during search.

-Recombination (Crossover)

After selection best parents in population based on their fitness function, recombination step combines their genes to create two new individuals in the population. These offspring share a combination of the genes of their two parents, recombination allows to mix their genotypes together to evolve a potentially more fittest child. The **Single-point crossover** mechanism is the most frequent implementation of the recombination step, and which we used in our VRP problem. Its works by choosing a random place for trunk in two parts the parents chromosomes, then generate two different child's by exchanging those parts by this manner:

-

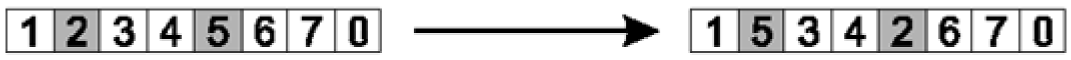


-Mutation

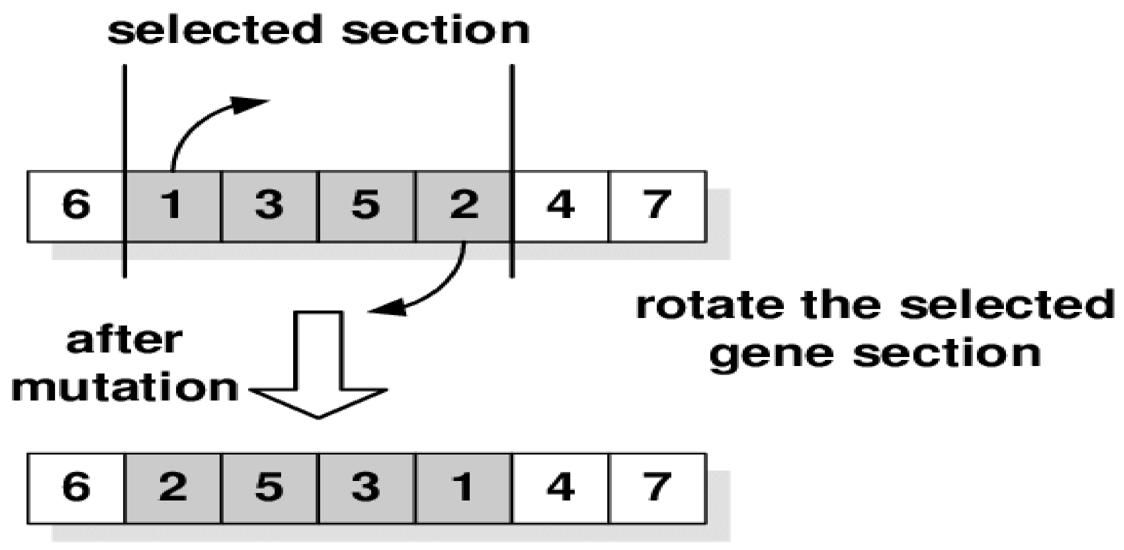
In simple terms, mutation may be defined as a small random tweak in the chromosome, to get a new solution. It is used to maintain and introduce diversity in the genetic population, we replace a given percentage of the population with random entries,

and is usually applied with a low probability.

Mutation is the part of the GA which is related to the "exploration" of the search space. There are many different options for choosing our mutation method. We have to try two:
In our first case, it has been the random swapping method. This method exchanges the position of two random genes on each given chromosome as we show now:



In the second case, we have to try the mutation by inversion. We need to select a random part of our chromosome, and invert the order of genes of this part. By this method we have found better solutions that with swapping mutation.



the repetition of these four steps: evaluation, selection, recombination and mutation, allow the population of entities to explore the search space. while maintaining diversity and continuous exploration the question of the terminal condition of the GAs is often discarded. Most of the GAs are run for a predefined number of iterations (repetitions), after which the outcome and the new population should be satisfying for our problem.

5.Dataset employed:

Random population.

6. Details of the Algorithm and Approaches:

- Used Algorithm

  -Genetic Algorithm is the algorithm we used in our VRP project.

  -It is not the strongest of the species that survives, nor the most intelligent, but the one most responsive to change, the genetic algorithm is one such optimization algorithm

  -Genetic algorithms are commonly used to **generate high-quality solutions to optimization and search problems**

  - The genetic algorithm one of the techniques that repeatedly modifies a population of individual solutions.

  -Random Forest Algorithm can be summarized into the following 4 steps:

  - Initialization of Population to generation solution for that.
  - Then select the best solution.
  - Then evaluation the result using fitness function.
  - Then do crossover (Recombination) and mutation

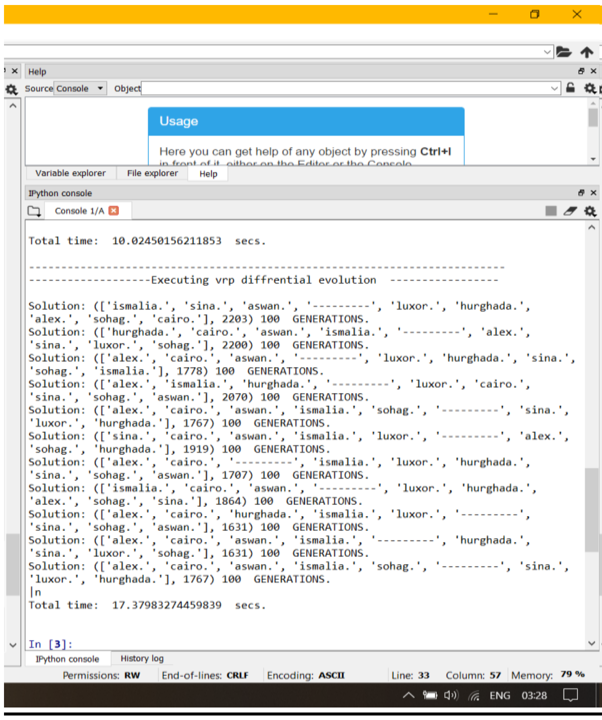
  -it is still work widely **in engineering optimization problems**. and the computer models model on genetic which attempt to solve complex problem.

Pseudocode for this Algorithm:

```
1-start
2-initilalize population randomly
3-define fitness function of the problem
4-determine the fitness of the population
5-while! converging or optimum not achieved do
6-parent selection from population
7-crossover operation for new population
8-perform mutation on the new population
9-calculate fitness of new population
10- end
11-if optimum achieved display the final result
12- stop
```

• **Result:**

-After taking a sample from dataset and apply main functions of the genetic Algorithm.



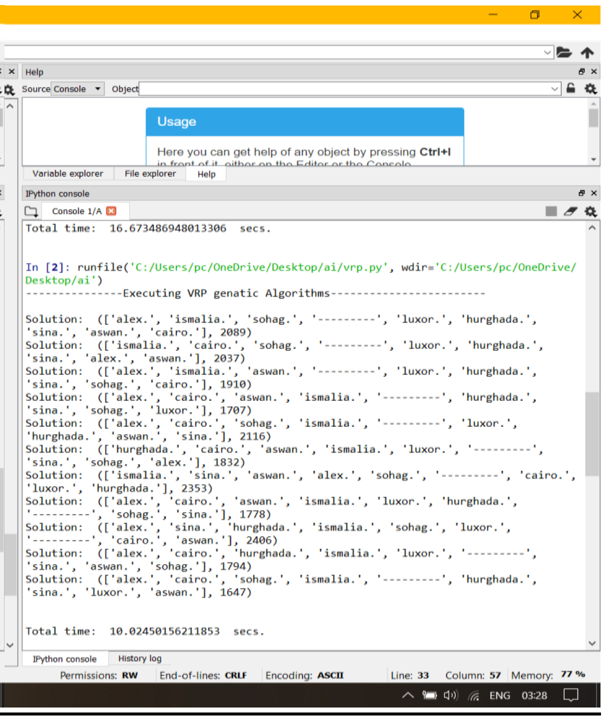
```
Total time:  10.02450156211853  secs.

--------------------------------------------------------------------------
------------------Executing vrp diffrential evolution  -----------------

Solution: (['ismalia.', 'sina.', 'aswan.', '---------', 'luxor.', 'hurghada.',
'alex.', 'sohag.', 'cairo.'], 2203) 100  GENERATIONS.
Solution: (['hurghada.', 'cairo.', 'aswan.', 'ismalia.', '---------', 'alex.',
'sina.', 'luxor.', 'sohag.'], 2200) 100  GENERATIONS.
Solution: (['alex.', 'cairo.', 'aswan.', '---------', 'luxor.', 'hurghada.', 'sina.',
'sohag.', 'ismalia.'], 1778) 100  GENERATIONS.
Solution: (['alex.', 'ismalia.', 'hurghada.', '---------', 'luxor.', 'cairo.',
'sina.', 'sohag.', 'aswan.'], 2070) 100  GENERATIONS.
Solution: (['alex.', 'cairo.', 'aswan.', 'ismalia.', 'sohag.', '---------', 'sina.',
'luxor.', 'hurghada.'], 1767) 100  GENERATIONS.
Solution: (['sina.', 'cairo.', 'aswan.', 'ismalia.', 'luxor.', '---------', 'alex.',
'sohag.', 'hurghada.'], 1919) 100  GENERATIONS.
Solution: (['alex.', 'cairo.', '---------', 'ismalia.', 'luxor.', 'hurghada.',
'sina.', 'sohag.', 'aswan.'], 1707) 100  GENERATIONS.
Solution: (['ismalia.', 'cairo.', 'aswan.', '---------', 'luxor.', 'hurghada.',
'alex.', 'sohag.', 'sina.'], 1864) 100  GENERATIONS.
Solution: (['alex.', 'cairo.', 'hurghada.', 'ismalia.', 'luxor.', '---------',
'sina.', 'sohag.', 'aswan.'], 1631) 100  GENERATIONS.
Solution: (['alex.', 'cairo.', 'aswan.', 'ismalia.', '---------', 'hurghada.',
'sina.', 'luxor.', 'sohag.'], 1631) 100  GENERATIONS.
Solution: (['alex.', 'cairo.', 'aswan.', 'ismalia.', 'sohag.', '---------', 'sina.',
'luxor.', 'hurghada.'], 1767) 100  GENERATIONS.
|n
Total time:  17.37983274459839  secs.


In [3]:
```

```
Permissions: RW    End-of-lines: CRLF    Encoding: ASCII    Line: 33   Column: 57   Memory: 79 %
```



```
Total time:  16.673486948013306  secs.


In [2]: runfile('C:/Users/pc/OneDrive/Desktop/ai/vrp.py', wdir='C:/Users/pc/OneDrive/
Desktop/ai')
--------------Executing VRP genatic Algorithms-----------------------

Solution:  (['alex.', 'ismalia.', 'sohag.', '---------', 'luxor.', 'hurghada.',
'sina.', 'aswan.', 'cairo.'], 2089)
Solution:  (['ismalia.', 'cairo.', 'sohag.', '---------', 'luxor.', 'hurghada.',
'sina.', 'alex.', 'aswan.'], 2037)
Solution:  (['alex.', 'ismalia.', 'aswan.', '---------', 'luxor.', 'hurghada.',
'sina.', 'sohag.', 'cairo.'], 1910)
Solution:  (['alex.', 'cairo.', 'aswan.', 'ismalia.', '---------', 'hurghada.',
'sina.', 'sohag.', 'luxor.'], 1707)
Solution:  (['alex.', 'cairo.', 'sohag.', 'ismalia.', '---------', 'luxor.',
'hurghada.', 'aswan.', 'sina.'], 2116)
Solution:  (['hurghada.', 'cairo.', 'aswan.', 'ismalia.', 'luxor.', '---------',
'sina.', 'sohag.', 'alex.'], 1832)
Solution:  (['ismalia.', 'sina.', 'aswan.', 'alex.', 'sohag.', '---------', 'cairo.',
'luxor.', 'hurghada.'], 2353)
Solution:  (['alex.', 'cairo.', 'aswan.', 'ismalia.', 'luxor.', 'hurghada.',
'---------', 'sohag.', 'sina.'], 1778)
Solution:  (['alex.', 'sina.', 'hurghada.', 'ismalia.', 'sohag.', 'luxor.',
'---------', 'cairo.', 'aswan.'], 2406)
Solution:  (['alex.', 'cairo.', 'hurghada.', 'ismalia.', 'luxor.', '---------',
'sina.', 'aswan.', 'sohag.'], 1794)
Solution:  (['alex.', 'cairo.', 'sohag.', 'ismalia.', '---------', 'hurghada.',
'sina.', 'luxor.', 'aswan.'], 1647)


Total time:  10.02450156211853  secs.
```

```
Permissions: RW    End-of-lines: CRLF    Encoding: ASCII    Line: 33   Column: 57   Memory: 77 %
```

7. Development Platform:

-PyCharm is one of the most common and popular integrated development environments (IDE) that is used by many programmers in computer programming, especially Python language. providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for **productive Python, web, and data science development** . ou can install and run PyCharm on as many machines as you have, and use the same environment and functionality across all your machines.

-PyCharm is cross-platform with Windows, macOS and Linux versions.

- It is developed by the Czech company JetBrains.

- The beta version was released in July 2010, with the 1.0 arriving 3 months later. Version 2.0 was released on 13 December 2011, version 3.0 on 24 September 2013, and version 4.0 on 19 November 2014.

-PyCharm provides many features for its users such as:

*Code Analysis.

*Graphical Debugger.

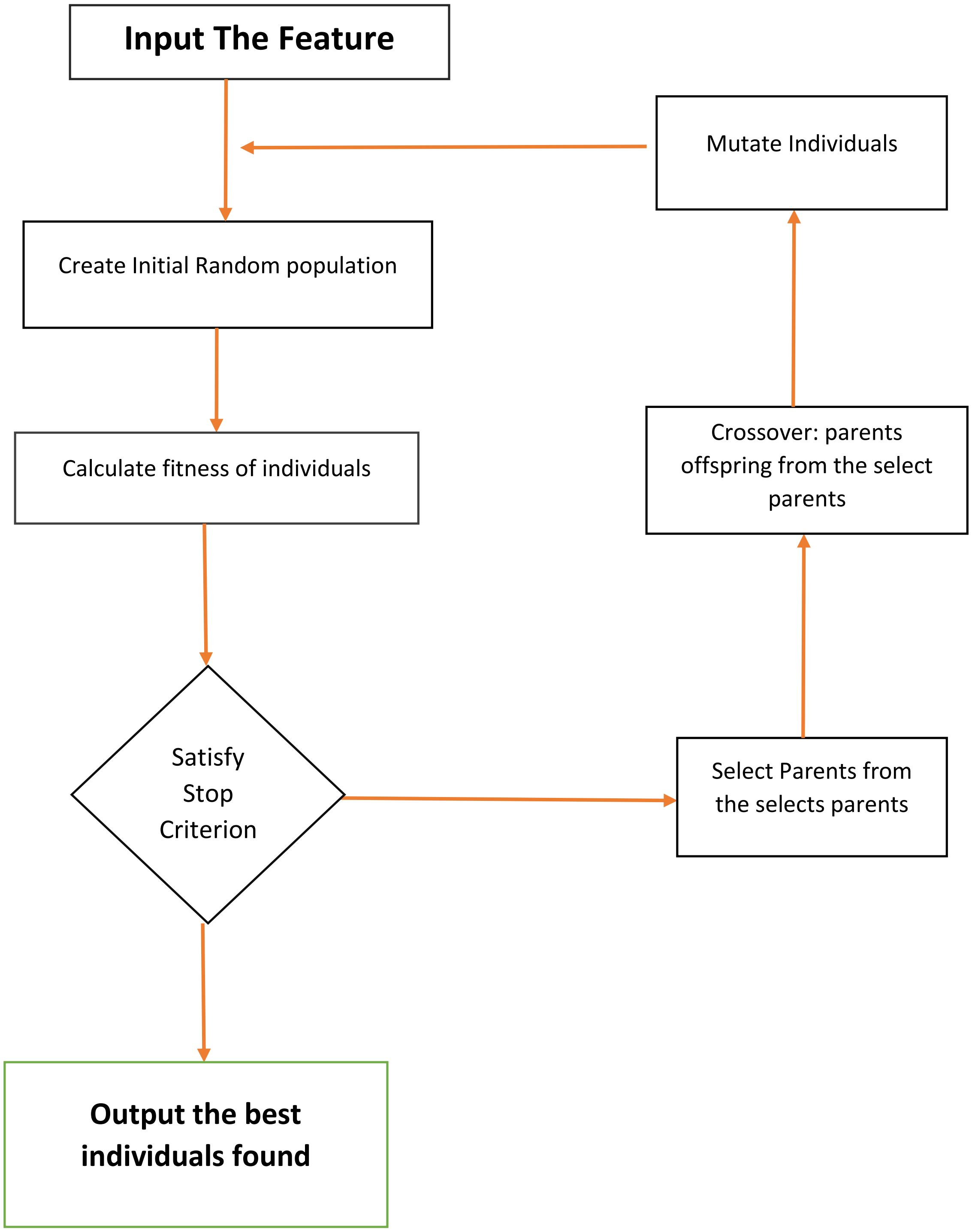
*Integrated unit tester.

*Support web frameworks such as Djano and as well as Data science with Anaconda.

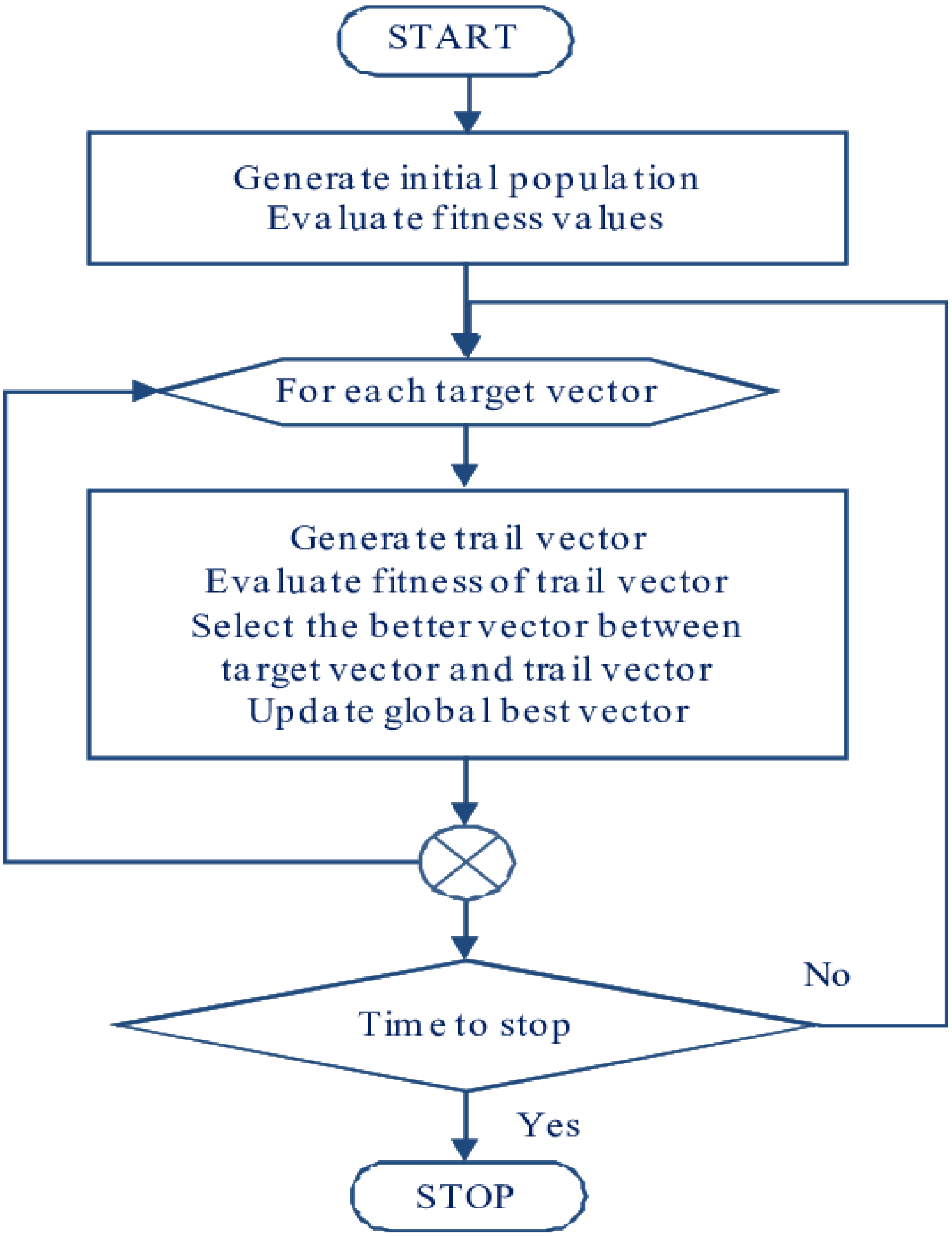
* Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages.

* Support for scientific tools like matplotlib, numpy and scipy.

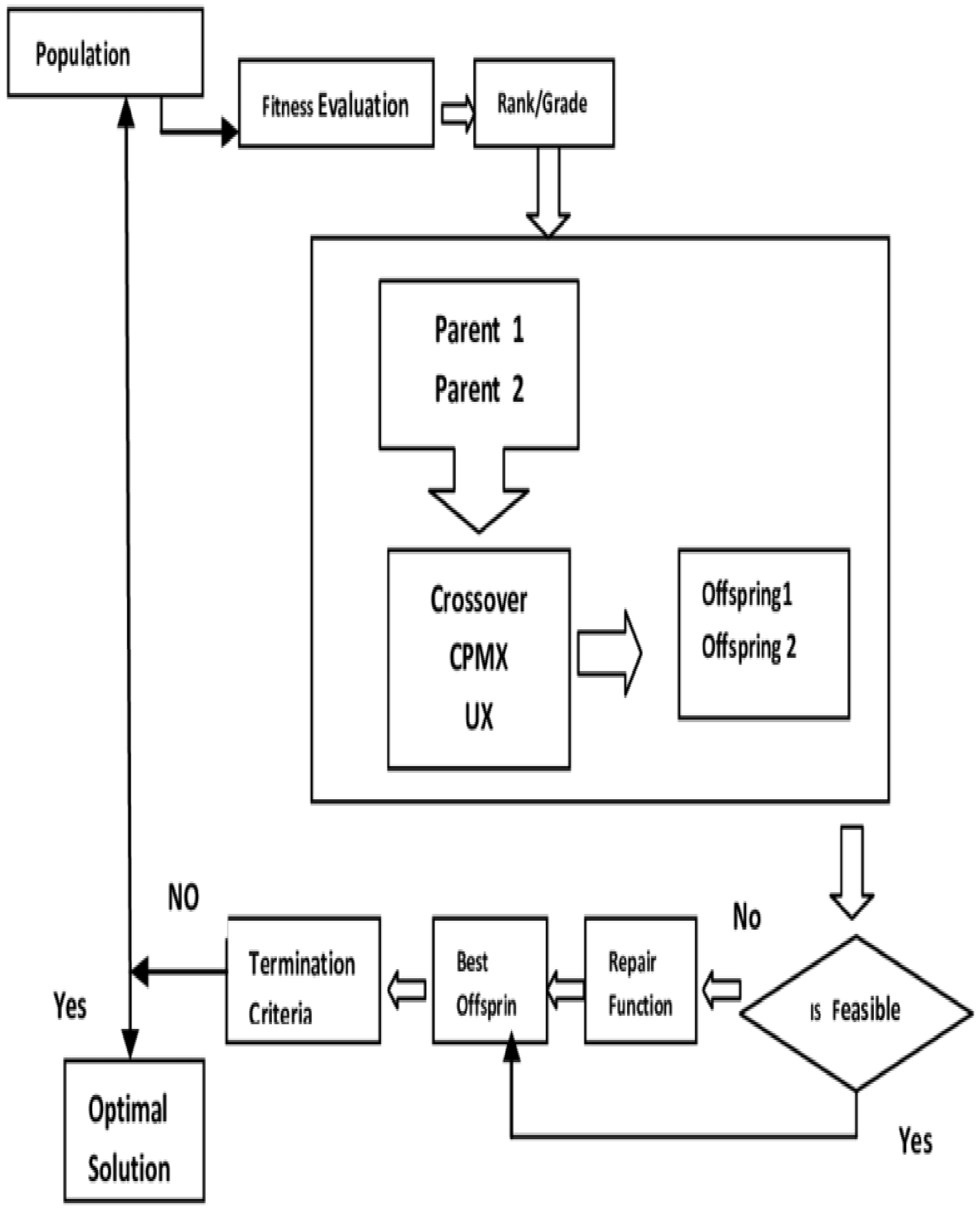
genetic algorithms:


Input The Feature
Create Initial Random population
Calculate fitness of individuals
Satisfy Stop Criterion
Output the best individuals found
Select Parents from the selects parents
Crossover: parents offspring from the select parents
Mutate Individuals

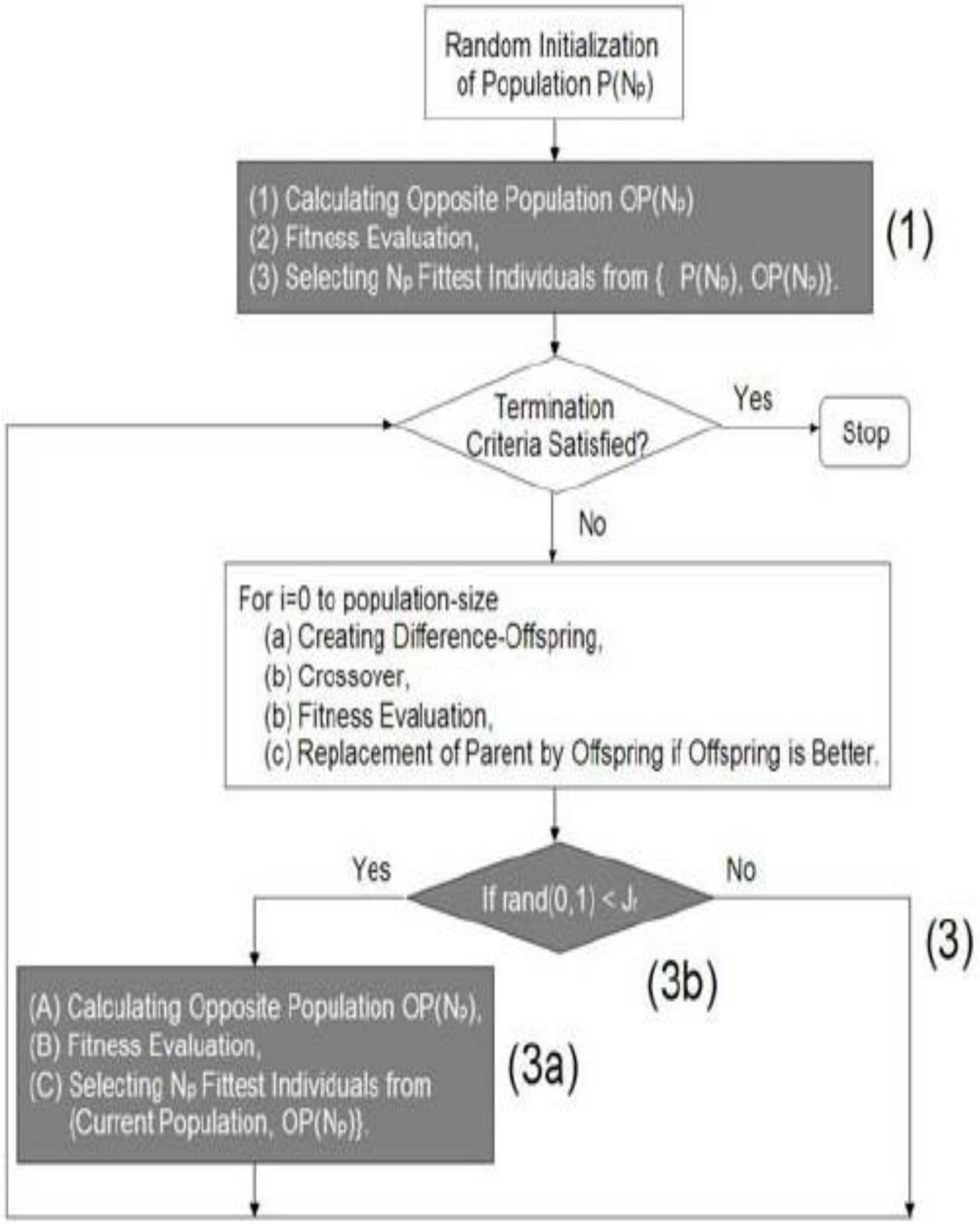
differential evolution:


START
Generate initial population
Evaluate fitness values
For each target vector
Generate trail vector
Evaluate fitness of trail vector
Select the better vector between
target vector and trail vector
Update global best vector
Time to stop
No
Yes
STOP

genetic algorithms:



differential evolution:


Random Initialization of Population P(Np)
(1) Calculating Opposite Population OP(Np)
(2) Fitness Evaluation,
(3) Selecting Np Fittest Individuals from { P(Np), OP(Np)}.
(1)
Termination Criteria Satisfied?
Yes
Stop
No
For i=0 to population-size
(a) Creating Difference-Offspring,
(b) Crossover,
(b) Fitness Evaluation,
(c) Replacement of Parent by Offspring if Offspring is Better.
Yes
If rand(0,1) < Jr
No
(3)
(3b)
(A) Calculating Opposite Population OP(Np),
(B) Fitness Evaluation,
(C) Selecting Np Fittest Individuals from {Current Population, OP(Np)}.
(3a)

Drive link:

https://tinyurl.com/ms4vfsmx