**A PRELIMENERY REPORT ON**

**“GENETIC ALGORITHM”**

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE

IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE

OF

**BACHELOR OF ENGINEERING (COMPUTER ENGINEERING)**

##### MADE BY

Amey Gondhalekar 421040

Gibraan Jafar 421037

Usman Khan 421054

Aayush Jain 421045



## DEPARTMENT OF COMPUTER ENGINEERING

**AIM**-The aim of this project is to apply genetic algorithm for optimization on Travelling Salesman Problem (TSP).

**OBJECTIVE**-The objective of this project is to understand the working of genetic algorithm.

**THEORY-**

A **Genetic Algorithm** is a search heuristic that is inspired by Charles Darwin’s theory of natural evolution. This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce offspring of the next generation.

Genetic Algorithms (GAs) are adaptive heuristic search algorithms that belong to the larger part of evolutionary algorithms. Genetic algorithms are based on the ideas of natural selection and genetics. These are intelligent exploitation of random search provided with historical data to direct the search into the region of better performance in solution space. They are commonly used to generate high-quality solutions for optimization problems and search problems.

Genetic algorithms simulate the process of natural selection which means those species who can adapt to changes in their environment are able to survive and reproduce and go to the next generation. In simple words, they simulate “survival of the fittest” among individual of consecutive generation for solving a problem. Each generation consist of a population of individuals and each individual represents a point in search space and possible solution. Each individual is represented as a string of character/integer/float/bits. This string is analogous to the Chromosome.

**Foundation of Genetic Algorithms:**

**The 5 phases in a Genetic Algorithm are:**

1. Initial Population
2. Fitness Function
3. Selection
4. Crossover
5. Mutation

Genetic algorithms are based on an analogy with genetic structure and behavior of chromosome of the population. Following is the foundation of GAs based on this analogy:–

1. The individual in population compete for resources and mate.

2. Those individuals who are successful (fittest) then mate to create more offspring than others.

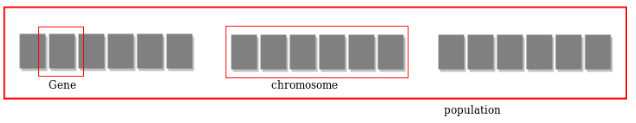
3. Genes from “fittest” parent propagate throughout the generation, that is sometimes parents create offspring which is better than either parent.

4. Thus each successive generation is more suited for their environment.

**Search space:**

The population of individuals is maintained within the search space. Each individual represents a solution in the search space for a given problem. Each individual is coded as a finite length vector (analogous to chromosome) of components. These variable components are analogous to Genes. Thus, a chromosome (individual) is composed of several genes (variable components).

**Fitness Score:**



A Fitness Score is given to each individual which ​shows the ability of an individual to “compete” ​. The individual having optimal fitness score (or near optimal) are sought. The GAs maintains the population of n individuals (chromosome/solutions) along with their fitness scores. The individuals having better fitness scores are given more chance to reproduce than others. The individuals with better fitness scores are selected who mate and produce ​better offspring by combining chromosomes of parents. The population size is static so the room has to be created for new arrivals. So, some individuals die and get replaced by new arrivals eventually creating new generation when all the mating opportunity of the old population is exhausted. Over successive generations better solutions will arrive while least fit die.

Each new generation has on average more “better genes” than the individual (solution) of previous generations. Thus, each new generation has better ​ “partial solution” than previous generations. Once the offspring produced having no significant difference than offspring produced by previous populations, the population is converged. The algorithm is said to be converted to a set of solutions for the problem.

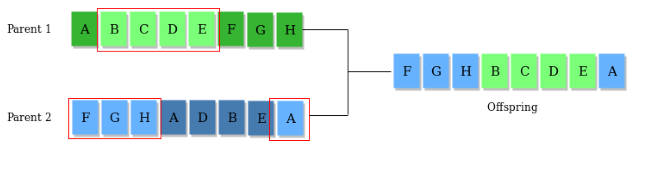
**Operators of Genetic Algorithms:**

Once the initial generation is created, the algorithm evolves the generation using the following operators –

1) Selection Operator: The idea is to give preference to the individuals with good fitness scores and allow them to pass their genes to the successive generations.

2) Crossover Operator: This represents mating between individuals. Two individuals are selected using selection operator and crossover sites are chosen randomly. Then the genes at these crossover sites are exchanged thus creating a completely new individual (offspring).

For example:



The whole algorithm can be summarized as:

1) Randomly initialize populations p

2) Determine the fitness of the population

3) Until convergence repeat:

a) Select parents from population

b) Crossover and generate a new population

c) Perform mutation on new population

d) Calculate fitness for the new population

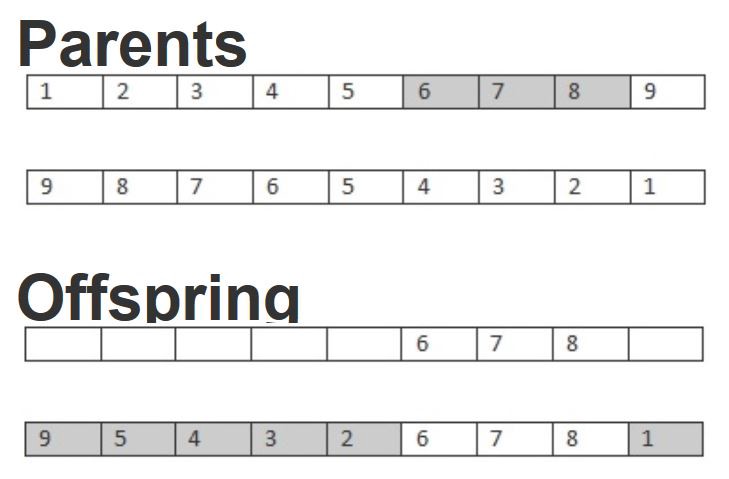
We select suitable parameters to be passed to the algorithm. The parameters include:

1. Population Size
2. Number of top chromosomes/parents to be selected
3. Mutation Rate
4. Number of Generations

The process begins with a set of individuals which is called a **Population**. Each individual is a solution to the problem you want to solve. We start with list of 25 cities with random x and y co-ordinates as our initial population. (0 < x, y < 200). We then create a pool of random routes of cities.

**Fitness proportionate selection** method is used to select parents that will be used to create the next generation. The fitness of each individual relative to the population is used to assign a probability of selection. We then filter the parents based on their fitness score. Lower the distance between the cities, higher is their fitness score.

With our mating pool created, we can create the next generation in a process called **crossover**(aka “breeding”). We will use a special breeding function called **ordered crossover**. In ordered crossover, we randomly select a subset of the first parent string and then fill the remainder of the route with the genes from the second parent in the order in which they appear, without duplicating any genes in the selected subset from the first parent.



Crossover

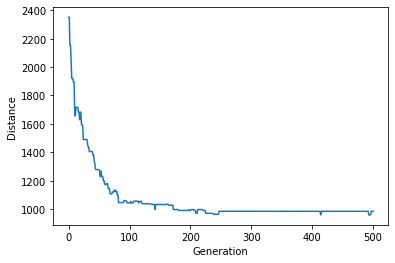
**Mutation** serves an important function in Genetic Algorithm, as it helps to avoid local convergence by introducing novel routes that will allow us to explore other parts of the solution space. Mutation would simply mean assigning a low probability of a gene changing from 0 to 1, or vice versa. However, since can’t drop cities we’ll use **swap mutation**. This means that, with specified low probability, two cities will swap places in our route.

This process gets repeated for the number of generations provided.

At the end of this algorithm, we get the optimal route covering all the cities.

**OUTPUT:**





**CONCLUSION:**

In this project we understood the working of Genetic algorithm on Travelling Salesman Problem (TSP).