Experiment No. 8

**Aim:** Case​ ​study​ ​of​ ​Traveling​ ​salesman​ ​problem​ ​using​ ​Genetic​ ​Algorithm.

**Theory:**

**Genetic​ ​Algorithm:**

The genetic algorithm is a method for solving both constrained and unconstrained optimization problems that is based on natural selection, the process that drives biological evolution. The genetic algorithm repeatedly modifies a population of individual solutions. At each step, the genetic algorithm selects individuals at random from the current population to be parents and uses them to produce the children for the next generation. Over successive generations, the population "evolves" toward an optimal solution. You can apply the genetic algorithm to solve a variety of optimization problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, nondifferentiable, stochastic, or highly nonlinear. The genetic algorithm can address problems​ ​of​ ​mixed​ ​integer​ ​programming,​ ​where​ ​some​ ​components​ ​are​ ​restricted​ ​to​ ​be​ ​integer-valued.

The genetic algorithm uses three main types of rules at each step to create the next generation from​ ​the​ ​current​ ​population:

* Selection rules select the individuals, called parents, that contribute to the population at the next generation.
* Crossover​ ​rules​ ​combine​ ​two​ ​parents​ ​to​ ​form​ ​children​ ​for​ ​the​ ​next​ ​generation.
* Mutation​ ​rules​ ​apply​ ​random​ ​changes​ ​to​ ​individual​ ​parents​ ​to​ ​form​ ​children.

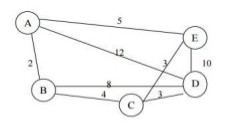
**Traveling​ ​Salesman​ ​problem:**

A traveler needs to visit all the cities from a list, where distances between all the cities are known and each city should be visited just once. What is the shortest possible route that he visits each city exactly​ ​once​ ​and​ ​returns​ ​to​ ​the​ ​origin​ ​city?

Although it may not be practical to find the best solution for a problem like ours, we do have algorithms that let us discover close to optimum solutions such as the nearest neighbor algorithm and swarm optimization. These algorithms are capable of finding a 'good-enough' solution to the travelling salesman problem surprisingly quickly. We will be using genetic algorithms as our optimization technique.

**Example:**

The aim of the following example is to minimize the total length of the trip



**Method Used:**

* We have a set of cities (points in 2D plane). Each city has a road to each city. We need to find loop-path that will be in each city only one time and path length is minimal.
* The genetic algorithm is sequence of following operations that repeated in generations loop:

1) Generate random population and finding path length.

2) find probabilities for selection.

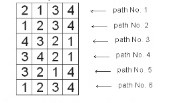
3) prepare to crossover according probabilities.

4) crossover, parents replaced with children.

5) mutations.

**Step 1** : ***Create Random Population***

* So we have a lot of paths with different lengths. A path can have crossover with another path and mutate. After crossover it will be replaced by its children. All paths recorded in matrix G.



* Here we have population with 6 paths (population size ps=6), and it is for 4 cities.
* In the code to generate initial population of random paths randperm matlab function was used. It return randomly permuted numbers, for example:

» randperm(4)

ans

3 4 2 1

**Step 2 : *Fitness evaluation***

* To have fast calculation of paths length before start iteration, it is calculated in a matrix of distances dsm.lt is a nxn matrix where n is number of cities, dsm(i1,i2) is distance between city i1 and city i2.
* The logic used is that if we want to minimize distance then we want to maximize inverse distance.
* Thus the fitness function we want to maximize is the inverse of distance

f(x)=1/d

**Step 3 *: Selection For Crossover***

* For this case roulette\_wheel\_indexes.m function was made. First calculate probabilities



* Here pi is probability of i path to be put to crossover, di is i path length.
* for example we have 6 paths with numbers

123456

* and let it have probabilities

0.05 0.5 0.05 0.05 0.05 0.3

* then by using roulette\_wheel\_indexes.m function we will put for the crossover for example

1 2 6 2 6 2

* You can see that 2 most frequent and 6 also frequent, rest ELIMINATED. But all this is random.
* So after that we have pair wise crossover:

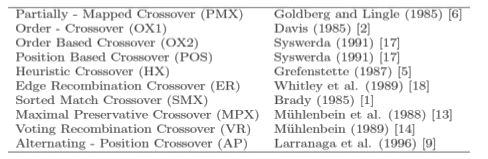
1 and 2 (gives 2 children)

6 and 2 (gives 2 children)

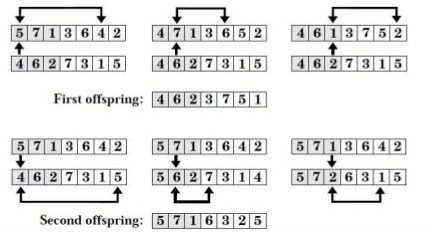
6 and 2 (gives 2 children)

**Step 4: *Crossover***

* The different crossover techniques used for solving the travelling salesman problem are listed below:



* Among these PMX, ER and POS are quoted to be the fastest operators as far as the number of necessary iterations to reach convergence is concerned . The convergence rates of these three operators are observed to be similar.

****

**Step 5: Mutation**

* There are 3 types of mutations used:

1. Choose 2 random cities



and swap them,

Result:



1. Mutation by exchange of 2 pieces of path :

First randomly chose point of spit, for example between 2 and 3 positions:



now swap two pieces:



1. Mutation of flip random piece of path:

first choose some random piece of path:



now flip it left-right:



**Conclusion:**

​Hence Traveling salesman problem using genetic algorithm studies successfully. Genetic algorithm​ ​includes​ ​the​ ​selection,​ ​crossover,​ ​mutation​ ​operators​ ​along​ ​with​ ​fitness​ ​function.