**Acknowledgements**

This project is a golden opportunity for learning and self development. We consider ourselves lucky to be a part in the completion of this project.

We take this opportunity to express our profound gratitude and regards to our guide Ms.AninditaDas for her exemplary guidance, monitoring and constant encouragement throughout the course of this project. Our grateful thanks to our guide who in spite of being extraordinarily busy with her duties, took time out to hear, guide and keep us on the correct path.

We would also like to thank our college **SWAMI VIVEKANANDA** **INSTITUTE OF SCIENCE AND TECHNOLOGY** and Head of Department , Mr.Pranab Gayen for the help provided to us to get us through such an excellent opportunity.

**Group Members:-**

1. **Priyanka Chakraborty**
2. **Suvra Mitra**
3. **Saswati Pradhan**
4. **Malyashree Bhaduri**
5. **Sweta Ghosh**
6. **Debasmita Basu**
7. **Madhurima Chowdhury**
8. **Sampurna Chatterjee**

**ABSTRACT**

Genetic algorithms are an evolutionary technique that is used to solve different optimization problems including the Travelling Salesman problem. In the travelling salesman problem, a salesman must visit every city in his territory exactly once and then return to the starting point. Given the costs of travelling between all pairs of cities, his objective is to minimize the total cost for the entire tour. In this project, we have a 5-city graph and the costs for travelling between each pair of cities. Our objective is to find the optimum cost for this travelling salesman problem using genetic algorithm.

**Table of Content**

1. Introduction

2. Single Objective Genetic Algorithm

2.1 Selection

2.2 Crossover

2.3 Mutation

3. Results

4. Scope of the Project

5. Bibliography

**1. INTRODUCTION**

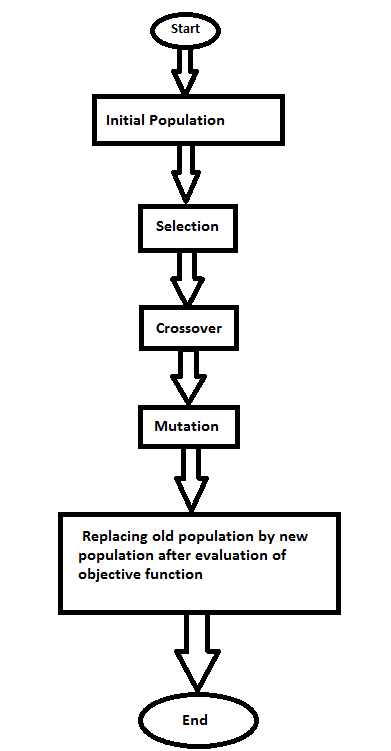
The idea of the Travelling Salesman problem is to find a tour of a given number of cities, visiting each city exactly once and returning to the starting city where the length of the tour is minimized. The standard TSP can be stated mathematically as follows: Given a weighted graph G=(V,E) where the weight Cij on the edge between nodes i and j is a non-negative value, we find the tour of all nodes that has the minimum total cost. The standard or symmetric TSP is a Hamiltonian path problem since the starting and the ending city have to be the same. The cost of travelling from city i to city j is taken to be same as the cost for travelling from j to i. The TSP is NP-hard, so there is no known algorithm that will solve it in polynomial time. One approach to solve it is the exhaustive enumeration, but it becomes practically useless when the number of cities is large. Other approaches like Dynamic Programming, Greedy algorithms etc. do not always give feasible solutions. In these types of optimization problems where the search space is large and complex, mathematical analysis is not available and traditional search methods fail, Genetic Algorithms can be applied. GAs can be used to obtain a near perfect solution in very little time.

**2. SINGLE OBJECTIVE GA**

The Genetic Algorithm is a randomized search and optimization technique based on the principle of natural evolution. These algorithms encode a potential solution to a specific problem on a simple chromosome-like data structure and apply recombination operators to these structures so as to preserve critical information.

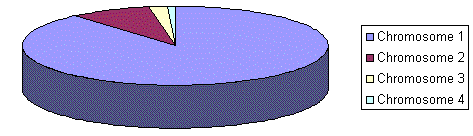
An implementation of GA begins with a population of random chromosomes. These structures are then evaluated and allocated reproductive opportunities in such a way that these chromosomes which represent a better solution to the target problem are given more chances to reproduce than those chromosomes which are poorer solutions. The GA process consists of the following steps:

* Selection which equates to survival of the fittest
* Crossover which represents mating between chromosomes
* Mutation which introduces random modification

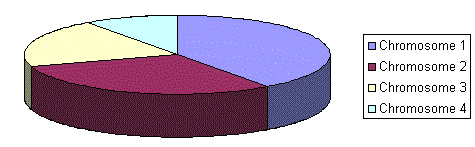
**Flow Chart for the GA** 

**2.1 Selection** - A common selection approach is Roulette Wheel Selection which assigns a probability of selection Pj to each individual j based on its cost function. A series of N random numbers is generated and compared against the cumulative probability of the population. The appropriate individual is selected and copied into the new population. The probability Pj for each individual is given by: where Fj equals the cost of individual j. The process repeats until the desired number of individuals are obtained. There are two types of Roulette Wheel Selection: - a) Proportional Roulette Wheel Selection, b) Rank based Roulette Wheel Selection.

In proportional roulette wheel, individuals are selected with a probability that is directly proportional to their fitness values i.e. an individual selection corresponds to a portion of a roulette wheel. The probabilities of selecting a parent can be seen as spinning a roulette wheel with the size of the segment for each parent being proportional to its fitness. The basic advantage of proportional roulette wheel selection is that it discards none of the individuals in the population and gives a chance to all of them to be selected.



Rank-based roulette wheel selection is the selection strategy where the probability of a chromosome being selected is based on its fitness rank relative to the entire population. Rank-based selection schemes first sort individuals in the population according to their fitness and then computes selection probabilities according to their ranks rather than fitness values. Rank-based selection scheme helps prevent premature convergence due to “super” individuals, since the best individual is always assigned the same selection probability, regardless of its objective value.



In our project, we have taken a 4-city travelling salesman problem. The cities and the cost incurred while travelling the path between any two cities can be shown in the following undirected graph.

A

15

13

18

C

B

17

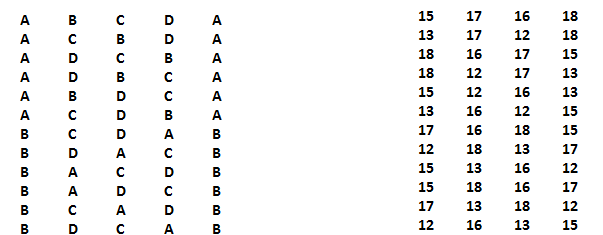
12

16

D

With these 4 cities A, B, C and D, an initial population of 24 chromosomes can be created with their respective input costs.

Input Data Set (Initial Pool) Input Costs









Here fitness function of each chromosome i.e. route is the total cost incurred while travelling that route. The method used in our project for selection is given below:

s = total sum of population fitness function

r = random number in the range 0-23 since 24 is the population size

k = s / r

Index of parent = k % population size

Thus any two parents are selected from these random indices. For example:

s = 1450

r = 3

k = 1450 / 3 = 484

So index of 1st parent = k % 24 = 484 % 24 = 4.

Again if r = 16

k = 1450 / 19 = 77

Index of 2nd parent = k % 24 = 77 % 24 = 5.

Thus from our initial population the selected parent chromosomes and their costs will be:

**A B D C A 15 12 16 13**

**A C D B A 13 16 12 15**

**2.2 Crossover** – It is the process of combining two parent members to form two new different members with similar characteristics as parents. Not every chromosome is used in crossover. The objective cost function is used to decide the probability of a chromosome participating in crossover. The chromosomes are chosen randomly to get the combining pairs and crossover is performed. The new chromosomes are moved into the new population. This will hopefully mean that the new population will be better than the last. Crossovers continue until the new population is full. The different types of crossovers that can be performed are Single point crossover, Double point crossover, Multipoint crossover, Uniform crossover, Subtour Exchange crossover, Arithmetic crossover, Directional crossover etc.

In Single Point Crossover, a single crossover point is randomly selected. The part from beginning to the crossover point is copied from one parent chromosome and the rest is copied from second parent. In this operation, the crossover site is not known, since it is randomly selected. If an appropriate site is chosen, good child strings are produced.

+

F

G

C

E

B

D

B

G

C

A

D

F

A

E

Parent 1 Parent 2

After crossover,

F

G

C

E

D

E

A

D

B

G

C

B

A

F

Child 1 Child 2

In our problem, we have used Single Point Crossover between the two parent chromosomes that are selected in the previous step by Roulette Wheel selection. The site of crossover is selected randomly in the range 1-5 as the size of each chromosome is 5. For example, the two selected parents were

**A**

**C**

D

**B**

**A**

**A**

**B**

**D**

**C**

**A**

and if the randomly selected index is 3, then after single point crossover the resulting child chromosomes will be:

**A**

**B**

**D**

**C**

**A**

**A**

**C**

**D**

**B**

**A**

It is possible to check each new chromosome to make sure it is valid and does not already exist in the population. It is also possible to move the best solution from previous population directly into the next population. This is called Elitism where the elite solutions are preserved for next stage. This means that the best solution can never get any worse since even if on average the population is worse, it will still include the best solution so far.

First we check whether the new chromosome is Hamiltonian or not i.e. the starting and ending cities will be same. If it is Hamiltonian, then we check whether any city is repeated inside the chromosome. If repetition occurs then the chromosome is invalid. E.g.:-

Invalid chromosome

**A**

**B**

**C**

**B**

**A**

Here city B is repeated, so the chromosome is invalid. If there is no such repetition then those valid chromosomes will be checked for elitism. For that we compare the total cost of travelling for each of the two parent chromosomes and the two child chromosomes. The best two of these i.e. the chromosomes with least costs will be written in the new generation pool. If it is seen that any of the parents in the previous generation was better, then that will be carried forward to the next generation. For example:

Parent 1:- 56

Parent 2:- 60

Total Costs of the routes

Child 1:- 66

Child 2:- 66

It is seen that the parents here have least total costs, so they are taken forward to the next generation pool.

**2.3 Mutation** - It is performed after crossover by randomly choosing a chromosome in the new population to mutate. A point is then randomly chosen and altered to avoid getting similar kind of children. Mutation is used so that we do not get trapped in a local optimum. So it is a completely random way of getting to possible solutions that would otherwise not be found. Mutation is generally of three types – a) Uniform, b) Boundary, c) Non-uniform.

In our problem we have used Boundary mutation. The chromosomes which were not Hamiltonian or had repetition or both are used in this step. We select the two boundary genes and change them i.e. mutate to any one random gene. E.g. if there is a non-Hamiltonian chromosome

**A**

**B**

**C**

**D**

D

we select the boundary genes A and D, mutate them and replace both by A. The resulting chromosome will be as below:

**A**

**D**

**C**

**B**

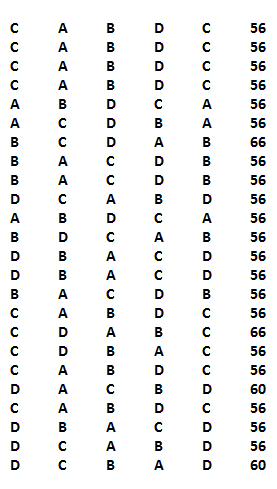
**A**

If even after mutation there remains any chromosome with invalid path, then that chromosome will be discarded. As soon as we obtain a valid chromosome after mutation, its fitness function will be compared with those of its parents. The chromosome with better fitness function will be taken into the next generation pool.

Thus after executing all the steps of this genetic algorithm for many new generation pools, we will get the optimal solution.

**3. RESULTS**

The number of generation after mutation is 100 where we see that the optimality has converged. The optimal solution is 56. The rate of crossover is 8.33%. The rate of mutation is 40%. The output for the 100th generation is given below:



**4.** **SCOPE OF THE PROJECT**

We have used the Genetic Algorithm in a symmetric Travelling Salesman problem. But in future it can be applied on asymmetric Travelling Salesman problems also.

This algorithm can also be applied on files taken from the OR library which are available on the internet. Then the same algorithm can be modified slightly to use dynamic array instead of static ones. But even then we cannot apply it on problems with more than 30 cities. Genetic Algorithms for 100 cities travelling salesman problem is still under research.

Genetic algorithms can also be applied by dynamic memory allocation for the size of populations which we will not know from beforehand if we use the OR library.

**CERTIFICATE**

Certified that this project report **“Single Objective Genetic Algorithm On Travelling Salesman Problem” is** the genuine work of Priyanka Chakraborty, Suvra Mitra, Saswati Pradhan, Malyashree Bhaduri, Sweta Ghosh, Debasmita Basu, Madhurima Chowdhury and Sampurna Chatterjee, students of CSE department, who carried out the project work under my supervision.

**Anindita Das**

**(Supervisor)**

**(Others)**

**Pranab Gayen**

**(Head of Department)**

**( External Supervisor)**

**5. Bibliography**

1. “Artificial Intelligence and Intelligent Systems”, N.P. Padhy.

2. “Genetic Algorithms and the Travelling Salesman Problem”, Kylie Bryant.

3. “Genetic Algorithm”, Tom V. Matthew.

4. “Modelling of a Roulette Wheel Selection in Genetic Algorithms”, T. Pencheva,

K. Atanassov, A. Shannon.