Traveling Salesman Problem Using Genetic Algorithm

Mohammad Asim
Assistant Professor, Department of Computer Science
MGM College of Engineering and Technology, Noida, Uttar Pradesh, India

Ritika Gopalia
Student (CSE)
MGM College of Engineering and Technology, Noida, Uttar Pradesh, India

Shivalika Swar
Student (CSE)
MGM College of Engineering and Technology, Noida, Uttar Pradesh, India

Abstract - Traveling salesman problem is quite known in the field of combinatorial optimization. Through this paper we describe how the traveling salesman problem is solved by the heuristic method of genetic algorithms. The purpose is to find the most approximate solution that gives us the least distance, which is the shortest route for traversing the cities given in the data set such that each city is passed through just once and the traveling salesman comes back to the initial city from where he started. We accomplish this by carrying out the algorithm through generating a fitness formula and with the help of genetic operators like selection, crossover and mutation.

Index Terms — genetic algorithm, genetic operators, traveling salesman problem.

I. INTRODUCTION

Given a collection of cities and the distance of travel between each pair of them, the traveling salesman problem is to find the shortest way of visiting all of the cities and returning to the starting point.[9]Though the statement is simple to state but it is more difficult to solve. Traveling Salesman Problem is an optimization problem and has a vast search space and is said to be NP-hard, which means it cannot be solved in polynomial time. It is one of the most fundamental problems in the field computer science in today's time. The Traveling salesman problem is being applied in many fields nowadays. Some of its applications are vehicle routing, manufacturing of microchips, packet routing in GSM, drilling in printed circuit boards etc.

In simpler words, say we have a set of n number of cities, and then we can obtain (n - 1)! alternative routes for covering all the n cities. Traveling salesman problem is to procure the route which has the least distance. Let us consider an example describing the Traveling salesman problem. We have a set of four cities A, B, C, and D. The distances between the cities are also given to us. Fig. 1 illustrates the collection of the cities and their distances among each other.

Here (4-1)!, that is 3! route can be generated. The tour with A→B→C→D→A will be the optimal route for given problem.

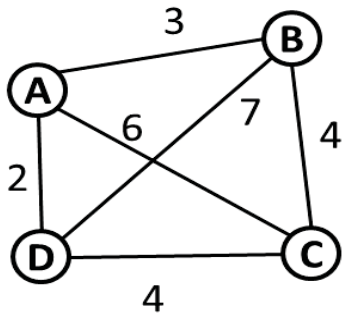


Fig. 1: Traveling salesman problem

Many heuristic techniques have been used to find the efficient solution to the problem like greedy method, ant algorithms, simulated annealing, tabu search and genetic algorithms. But as the number of cities increases the computation to find the solution becomes difficult. Despite the computational difficulty, we can use methods like genetic algoithms and tabu search which can give near to optimal solution for thousands of cities. Recently the shortest tour for a world TSP was found on 24 May, 2013. The tour of length 7,515,772,212 spans 1,904,711 locations all over the world.

II. GENETIC ALGORITHM

Genetic algorithm is a heuristic search algorithm which has been used to solve search and optimization problems. It is based on the principle of "the survival of the fittest", given by Charles Darwin. It is said to simulate the natural evolution carried out in living beings. Genetic Algorithms have been applied in fields like physics, chemistry, bioinformatics, mathematics, economics, computational science and crypto-analysis pharmacometrics as well as phylogenetics. They are being used widely for optimization of NP-Complete problems like the Traveling salesman problem, Hamiltonian path problem, knapsack problem etc. Other such optimization algorithms that can be used are simulated annealing, tabu search and artificial neural networks.

In genetic algorithms, the search space consists of individuals called chromosomes. Each individual (chromosome) comprises of genes. In order for the genetic algorithm to function efficiently, there must be some criteria to be achieved. The goodness of a chromosome based on some specific criteria gives us the fitness of the chromosome. Generally the fitter chromosomes are selected for the further process. The basic operators that are used in genetic algorithms are given as follows:

1. Selection Operator
2. Crossover Operator
3. Mutation Operator

Selection Operator

This operator is used to select the fitter chromosomes from the population for later breeding. The selected chromosomes are called parent chromosomes. The selection of the chromosomes is done on the basis of some fitness criteria. This criterion is usually formed depending on the type of problem. In case of Traveling salesman problem the fitter chromosome would be the one whose value of the genes would be minimised.

Crossover Operator

Another basic operator is the crossover operator. The crossover operator is used to combine two parent chromosomes in order to get a new pair of child chromosomes. The point at which the chromosomes are changed is called the crossover site. The different types of crossover are:

Single Point Crossover

In this a single crossover point is chosen for crossover and the parts after the crossover site of the parent chromosomes are exchanged. Fig. 2 shows how the single point crossover is carried out.

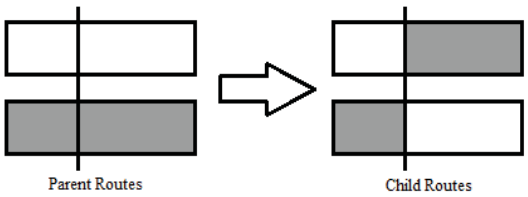


Fig. 2: Single Point Crossover

Multi Point Crossover

In this a multiple points are chosen for performing crossover and the segments of the parent chromosomes are exchanged. Fig. 3 depicts the multi point crossover.

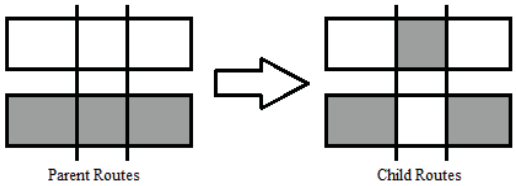


Fig. 3: Multi Point Crossover

Mutation Operator

Mutation operator is performed after crossover. It is used to preserve the genetic diversity of chromosomes at each generation of population. It is carried out by altering the genes of the chromosomes. Here we randomly flip some genes in order to maintain diversity. Or we can also exchange some genes present in a chromosome. It can be seen in fig. 4 how the mutation is achieved.

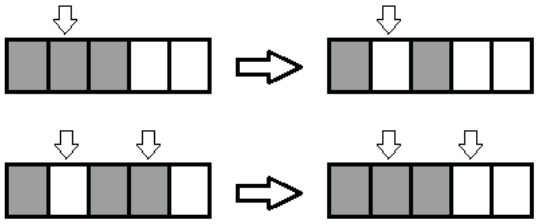


Fig. 4: Mutation in chromosome

The basic structure of the genetic algorithm can be given as follows:

1. [Start] A random set of chromosomes is selected out of all the feasible solutions, called as population.
2. [Fitness] Fitness of each chromosome present in the population is calculated depending on the fitness criteria.
3. [Selection] Two parent chromosomes are selected from a population on the basis of their fitness.
4. [Crossover] Crossover operation is used on the selected parent chromosomes to form new set of offsprings.
5. [Mutation] Mutation operation can also be performed on the newly acquired offsprings if required.
6. [Check] If the specified criterion is met, return the best solution and stop.
   Else, replace the old population with the new population and go to step 2 and use the newly generated population for further.

III. LITERATURE SURVEY

We see how genetic algorithm has been used for solving traveling salesman problems for about 442 cities [4]. A genetic algorithm have been proposed by Muhlenbein that generates a near to optimal for Traveling salesman problem for 442 and 531 cities. Here they have bettered the existing approach by improving the genetic operators of the genetic algorithm. They concluded with the result that on a SUN workstation, medium sized traveling salesman

problems for up to 229 cities can be solved in less than three minutes. Furthermore we could solve Traveling salesman problems with up to 442 cities optimally in an acceptable time limit (e.g. the average runtime on a SUN workstation for the TSP (431) is about 30 minutes).

In this thesis [6], genetic algorithm solves the traveling salesman problem to find the shortest path successfully, and also it follows the self-propelled vehicle path planning validation results. These results can also be implemented for some future automated guided vehicle for the distribution and production operations. By calculating the shortest route through Traveling Salesman Problem (TSP), genetic algorithm proves its effectiveness in gaining the multi-target route planning of the autonomous line-tracking car with high accuracy. In this system, the route planning requests that each target should be made to pass through just once and the line-tracking car must return to the starting point. The route planning of the line-tracking car has already been implemented for 10 or more targets successfully.

In another paper [7], it seems that the methods that use heuristic information or encode the edges of the tour perform the best and give good indications for future work in this area. Although, genetic algorithms haven't found any better solution for the traveling salesman problem than that are already existing, but genetic algorithm method has been the basis for many of the already known best solution. The biggest problem with the genetic algorithms devised for the traveling salesman problem is that it can be difficult to maintain structure from the parent chromosomes and still end up with a legal tour in the child chromosomes. Maybe a better crossover or mutation routine which retains structure from the parent chromosomes would give a better solution that has been already found for some other traveling salesman problems.

In the work proposed by Kylie Bryant "Genetic Algorithms and the Traveling Salesman Problem" [8] Genetic algorithms use crossover and mutation operators to solve optimization problems using the theory of the survival of the fittest. They have been used in a variety of problems, which includes the traveling salesman problem. In the Traveling salesman problem a weighted graph is given in which we have to find a tour of all nodes in order to minimize the total weight of the graph. Various operators have been devised for crossover and mutation for the traveling salesman problem and each gives a different results. These results are compared and it is found that operators that use heuristic information have the best outcome.

IV. METHODOLOGY

As we know, the Traveling salesman problem is a NP-hard problem, consisting of vast search spaces and finding its exact or optimal result usually requires a large computational time. Many heuristic techniques like branch and bound, greedy method, simulated annealing, tabu search etc. have been implemented to solve the TSP and they have provided efficient results for large problems. But in this paper we see how genetic algorithms can be used to solve the Traveling salesman problem. It not only gives the more accurate results for optimization problems as compared to its counterparts, but the genetic operators that it provides exceedingly helps in speeding up the search process.

The pseudo code for genetic algorithm for implementing the Traveling salesman problem is given as follows:

```
GeneticAlgorithm( )
{
        Initialize population of routes of cities randomly with a function Random( )
        Evaluate the fitness of each individual route using function Fitness( )
        While the fitness criteria is not satisfied
        do
        {
        Selection of two routes for reproduction using select function
        (Select (parent_route1, parent_route2))
        Perform crossover on the selected parent routes with crossover function
        (child_route = Crossover (parent_route1, parent_route2))
        Perform mutation on the newly generated child routes with mutation function
```

(Mutation (child_route))

Evaluate the fitness of child_route and replace the parent population with child_route

}

}

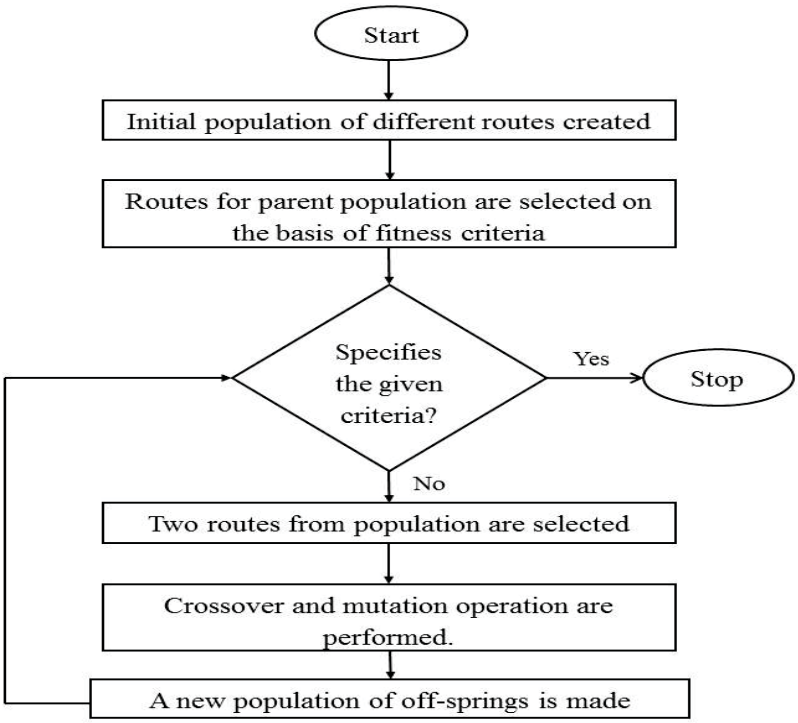


Fig. 5: Flowchart of genetic algorithm for TSP

Fig. 5 depicts the flow chart of genetic algorithm for the Traveling salesman problem.

V. ARCHITECTURE

The architecture of the genetic algorithm being implemented for solving the Traveling salesman problem has been shown through an example.[10] Consider a set of 5 cities A, B, C, D, and E as shown in fig. 6. The distances between each of these cities are given in the matrix shown below.

Table I: Distance between cities

	A	B	C	D	E
A	0	6	8	7	1
B	6	0	5	8	2
C	8	5	0	4	3
D	7	8	4	0	5
E	1	2	3	5	0

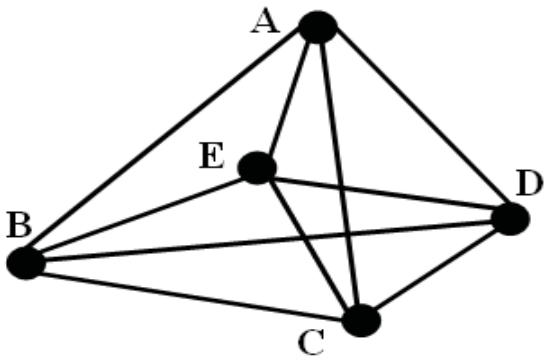


Fig. 6: 5 city TSP example

Table I gives us the distances between all the cities in the form of a matrix. Since we have 5 cities then the total number of routes that can be formed will be (5-1)!, that is 24 routes. The initial population that will be generated is as follows:

Route 1: A→E→C→D→B→A→ 22

Route 2: B→E→D→C→A→B→ 25

Route 3: E→A→C→B→D→E→ 27

Route 4: C→A→B→D→E→C→ 30

Route 5: B→C→D→A→E→B→ 19

Route 6: A→B→D→C→E→A→ 22

Route 7: C→B→D→A→E→C→ 24

Route 8: D→A→C→B→E→D→ 27

Route 9: E→C→D→A→B→E→ 22

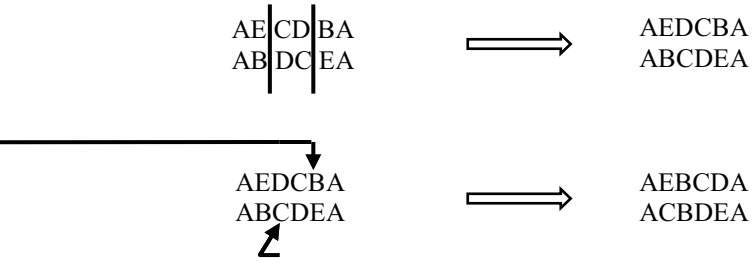
Route 10: E→C→A→D→B→E→ 28

Route 11: B→C→E→D→A→B→ 26

Route 12: E→D→B→A→C→E→ 30

and so on till we get distances of all the routes.

On the basis of the given fitness criteria we select the route 1 A→E→C→D→B→A→ 22 and route 6 A→B→D→C→E→A→ 22; and route 5 B→C→D→A→E→B→ 19 and route 10 E→C→A→D→B→E→ 28 and perform crossover and mutation operation on them. Taking parent routes as P1=AECDBA and P2=ABDCEA, we perform crossover and mutation operations over them:



We get child routes C1=AEBCDA and C2=ACBDEA. Now, taking parent routes as P1=BCDAEB and P2=ECADBE, we get child routes C1= BCAEDB and C2= EADCBE.

BC|DA|EB ⟹ BCADEB
EC|AD|BE ECDABE

The old population is replaced by the newly generated population:

A→E→B→C→D→A→ 19

A→ →A→ 27
BCADEB
B→ ECDABE →B→ 27 ⟹ BCAEDB EADCBE

E→A→D→C→B→E→ 19

We repeat the same procedure of evolution with this new population. Now taking parent routes as P1=ACBDEA and P2=BCAEDB,

ACB|DE|A
BCA|ED|B ⟹ ACBEDA BCADEB

ACBEDA
BCADEB ⟹ AEBCDA BEADCB

We get child routes C1= AEBCDA and C2=BEADCB. The length of their route if found to be 19 units.

A→E→B→C→D→A→ 19

B→E→A→D→C→B→ 19

Therefore the resultant route will be given as A→E→B→C→D→A whose distance is 19 units. Fig. 7 shows the shortest path that is selected to get the optimal solution.

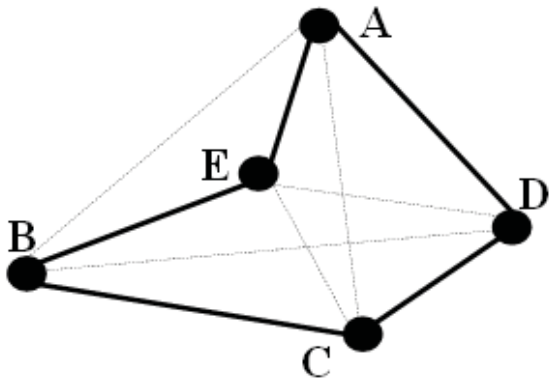


Fig. 7: Shortest path

VI. CONCLUSION

In this paper, we see how efficiently the genetic algorithm works for Traveling salesman problem. We observe how it creates solution without having any prior knowledge about the problem. Unlike other heuristic methods, it uses natural techniques like crossover, mutation and selection to make the computation easier and many times faster.

Some issues that can be faced during the algorithm are faulty child population generation which may be generated after crossover. That is, it may generate routes that are not comprehensible. Also, individuals in a population can become repetitive which needs to be avoided.

REFERENCES

[1] Richard Eglese, "Human interaction in solving hard practical optimization problems", Department of Management Science, Lancaster University Management School, Lancaster, U.K.

[2] D. B. Fogel, "An Evolutionary Approach to the Traveling Salesman Problem", Biol. Cybern. 60,139-144 (1988)

[3] Cheng, M. G. (1996). A survey of Penalty Techniques in Genetic Algorithms. *Evolutionary Computatiion 1996, Proceeding of 1EEE Internation Conference*, (pp. 804-809). Nagoya.

[4] Heinrich Braun, "On Solving Traveling Salesman Problems by Genetic Algorithms", Institut fiir Logik, Komplexitht und Deduktionssysteme, University Karlsruhe, Pg. 129-133.

[5] Solving traveling salesman problem for route planning realization of an autonomous line-tracking car by means of genetic algorithm,Lin Yong ; Lin, Yung

[6] Open Loop Traveling Salesman Problem using Genetic Algorithm, International Journal of Innovative Research in Computer and Communication Engineering Vol. 1, Issue 1, March 2013.

[7] Kylie Bryant ,Arthur Benjamin, Advisor, "*Genetic Algorithms and the Traveling Salesman Problem*", Department of Mathematics, December 2000.

[8] Rajesh Matai1, Surya Prakash Singh, Murari Lal Mittal," Traveling Salesman Problem: An Overview of Applications, Formulations, and Solution Approaches".

[9] http://www.iaindunning.com/simplexjs-tsp-demo/

[10] http://www.slideshare.net/ahlamansari/analysis-of-algorithm